UNIVERSIDAD POLITÉCNICA DE VALENCIA
DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Advances in Fully-Automatic and Interactive Phrase-Based Statistical Machine Translation

Thesis
presented by Daniel Ortiz Martínez

Supervisors:
Dr. Francisco Casacuberta Nolla
Dr. Ismael García Varea

Revised by:
Dr. Hermann Ney
Dr. Philipp Koehn
Dr. Juan Miguel Vilar Torres

Members of the committee:
Dr. Enrique Vidal Ruiz
Dr. Hermann Ney
Dr. Philipp Koehn
Dr. Marcello Federico
Dr. Juan Miguel Vilar Torres

September, 2011

Advances in Fully-Automatic and Interactive Phrase-Based Statistical Machine Translation

Daniel Ortiz Martínez

Thesis performed under the supervision of doctors
Ismael García Varea and Francisco Casacuberta Nolla
and presented at the Universidad Politécnica de Valencia
in partial fulfilment of the requirements for the degree
Doctor en Informática

Valencia, September, 2011

MIPRCV
CONSOLIDER INGENIO 2010
Multimodal Interaction in Pattern Recognition and Computer Vision

Work partially supported by the EC (FEDER/FSE), the Spanish Government (MEC, MICINN, MITyC, MAEC, “Plan E”, under grants MIPRCV “Consolider Ingenio 2010” CSD2007-00018, iTrans2 TIN2009-14511, erudito.com TSI-020110-2009-439), the Generalitat Valenciana (ALMPR grant Prometeo/2009/014, grant GV/2010/067) and the FPI fellowship.

ACKNOWLEDGEMENTS

Han transcurrido ya varios años desde que uno de mis codirectores de tesis, Ismael, me hablara por primera vez sobre la traducción automática estadística en su despacho de la Escuela Politécnica Superior de Albacete. La historia de esta tesis en cierta forma arranca ese día y se ha prolongado, al menos, hasta el momento de escribir estas líneas.

Durante todo ese tiempo, es mucha la gente que de un modo u otro ha influído en este trabajo. Quisiera dar las gracias en primer lugar a mis directores de tesis, Francisco Casacuberta e Ismael García Varea, quienes me introdujeron en el mundo de la investigación y me han enseñado tantas cosas. Voy a recordar siempre con especial cariño la asignatura de redes neuronales artificiales que impartía Paco en quinto de carrera, por transmitirme la curiosidad y la fascinación por el campo del reconocimiento de patrones. Con Isma he compartido mucho, tanto dentro como fuera del ámbito académico. Como bien dijo él en una ocasión, primero fui su alumno y después nos convertimos en amigos, y es una amistad de la que me he sentido orgulloso a lo largo de estos años.

Ha sido para mí un privilegio colaborar en diversos proyectos de investigación dentro del PRHLT. En especial, quiero agradecer a Paco por haberme permitido trabajar en el proyecto I3MEDIA perteneciente al programa Cénit, y a Enrique Vidal y a José Miguel Benedí por darme la oportunidad de participar en el proyecto MIPRCV perteneciente al programa Consolider. Esta tesis no podría haberse completado sin el apoyo económico de dichos proyectos.

También quiero agradecer al profesor Hermann Ney por haberme acogido tan amablemente en su departamento durante mi estancia en Aachen. No puedo dejar de mencionar aquí a David Vilar por toda la ayuda prestada para adaptarme a esa ciudad y al funcionamiento del departamento.

Mis compañeros del PRHLT y del ITI han sido una de las razones por las que esta etapa ha merecido la pena. Ha sido una experiencia buenísima trabajar con Vicent Alabau y Luis Leiva en el prototipo de traducción interactiva presentado en esta tesis. Otro compañero con el que la colaboración ha sido cada vez más estrecha es Jesús González, me encantaría que esa colaboración continuase por mucho tiempo. También quería dar las gracias a Jorge Civera, por lo útil que me ha resultado el material sobre el algoritmo EM que hay en su tesis, agradecimiento que me gustaría hacer extensivo a Alfons Juan como codirector de la misma. Gracias a Antonio Lagarda y Luis Rodríguez por su ayuda en relación a aspectos teóricos y prácticos de la traducción interactiva, a Alejandro Toselli por sus conocimientos y soporte técnico, a Verónica Romero por su software de cálculo de error para sistemas de post-edición, a Ricardo Sánchez por sus sugerencias sobre el título de la tesis y a Germán Sanchís y Jesús Andrés por los artículos en los que hemos colaborado.

Fuera del ámbito académico, existen tres personas a las que quisiera dar especialmente las gracias. Una de ellas es Diego Escudero, quien desde que llegué a Valencia, se convirtió para mí en un gran amigo y me ayudó en muchas cosas, seguramente mucho más de lo que él se imagina.

Tal vez la persona que más me ha apoyado durante el desarrollo de la tesis es Alicia. Su influencia quedará para siempre en todos y cada uno de los capítulos de este trabajo y también en otro tipo de capítulos, los que uno escribe al vivir su vida, pues ella está sin duda entre lo más importante que me ha pasado nunca.

Por último, no quisiera terminar sin referirme a la persona con quien más deseo compartir este instante. Esa persona es mi madre. Por todo lo que me ha enseñado, por su apoyo en los malos momentos, y por otras muchas cosas que no cabe contar aquí, creo que es ella más que nadie quien me ha brindado la oportunidad, no sólo de escribir esta tesis, sino de dedicarme a algo que me apasiona durante todos estos años.

Daniel Ortiz Martínez
Valencia, September 30, 2011

ABSTRACT

This thesis presents different contributions in the fields of fully-automatic statistical machine translation and interactive statistical machine translation.

In the field of statistical machine translation there are three problems that are to be addressed, namely, the modelling problem, the training problem and the search problem. In this thesis we present contributions regarding these three problems.

Regarding the modelling problem, an alternative derivation of phrase-based statistical translation models is proposed. Such derivation introduces a set of statistical submodels governing different aspects of the translation process. In addition to this, the resulting submodels can be introduced as components of a log-linear model.

Regarding the training problem, an alternative estimation technique for phrase-based models that tries to reduce the strong heuristic component of the standard estimation technique is proposed. The proposed estimation technique considers the phrase pairs that compose the phrase model as part of complete bisegmentations of the source and target sentences. We theoretically and empirically demonstrate that the proposed estimation technique can be efficiently executed. Experimental results obtained with the open-source THOT toolkit also presented in this thesis, show that the alternative estimation technique obtains phrase models with lower perplexity than those obtained by means of the standard estimation technique. However, the reduction in the perplexity of the model did not allow us to obtain improvements in the translation quality.

To deal with the search problem, we propose a search algorithm which is based on the branch-and-bound search paradigm. The proposed algorithm generalises different search strategies that can be accessed by modifying the input parameters. We carried out experiments to evaluate the performance of the proposed search algorithm.

Additionally, we also study an alternative formalisation of the search problem in which the best alignment at phrase-level is obtained given the source and target sentences. To solve this problem, smoothing techniques are applied over the phrase table. In addition to this, the standard search algorithm for phrase-based statistical machine translation is modified to explore the space of possible alignments. Empirical results show that the proposed techniques can be used to efficiently and robustly generate phrase-based alignments.

One disadvantage of phrase-based models is its huge size when they are estimated from very large corpora. In this thesis, we propose techniques to alleviate this problem during both the estimation and the decoding stages. For this purpose, main memory requirements are transformed into hard disk requirements. Experimental results show that the hard disk accesses do not significantly decrease the efficiency of the SMT system.

With respect to the contributions in the field of interactive statistical machine translation, on the one hand, we present alternative techniques to implement interactive machine translation systems. On the other hand, we give a proposal of an interactive machine translation system which is able to learn from user-feedback by means of online learning techniques.

We propose two alternative techniques for interactive statistical machine translation. The first one is based on the generation of partial alignments at phrase level. This approach constitutes an application of the phrase-based alignment generation techniques that are also proposed in this thesis. The second proposal tackles the interactive machine translation process by means of word graphs and stochastic error-correction models. The proposed approach differs from other existing approaches described in the literature in the introduction of error-correction techniques in the statistical framework of the interactive machine translation process. We carried out experiments to evaluate the two proposed techniques, showing that they are competitive with state-of-the-art interactive machine translation systems. In addition to this, such techniques have been used to implement an interactive machine translation prototype following a client-server architecture.

Finally, the above mentioned interactive machine translation system with online learning is based on the use of statistical models that can be incrementally updated. The main difficulty defining incremental versions of the statistical models involved in the interactive translation process appears when such models are estimated by means of the expectation-maximisation algorithm. To solve this problem, we propose the application of the incremental version of such algorithm. The proposed interactive machine translation system with online learning was empirically evaluated, demonstrating that the system is able to learn from scratch or from previously estimated models. In addition to this, the obtained results also show that the interactive machine translation system with online learning significantly outperforms other state-of-the-art systems described in the literature.

RESUMEN

Esta tesis presenta diversas contribuciones en los campos de la traducción automática estadística y la traducción interactiva desde un enfoque estadístico.

En el campo de la traducción automática estadística, se presentan contribuciones en relación a los tres problemas fundamentales a abordar en dicha disciplina: el problema del modelado, el problema del entrenamiento y el problema de la búsqueda.

Respecto al problema del modelado, se propone una derivación alternativa de los modelos de secuencias de palabras. Dicha derivación introduce un conjunto de submodelos probabilísticos que gobiernan diversos aspectos del proceso de traducción. Adicionalmente, los submodelos que se obtienen pueden introducirse como componentes de un modelo log-lineal.

Con respecto al problema del entrenamiento, se describe una técnica alternativa de estimación de modelos de secuencias de palabras que trata de reducir la fuerte componente heurística de las técnicas de entrenamiento estándar. La técnica de estimación propuesta considera los pares de secuencias de palabras que componen el modelo como parte de bisegmentaciones completas de las frases origen y destino. Se demuestra tanto teórica como empíricamente que la nueva técnica de estimación puede ejecutarse eficientemente. Resultados experimentales obtenidos con la herramienta de estimación de libre uso THOT presentada en esta tesis, demuestran que la técnica de estimación propuesta obtiene modelos con menor perplejidad que los obtenidos con la técnica de estimación estándar. Pese a ello, no se han conseguido mejoras en los resultados de traducción.

Para abordar el problema de la búsqueda se propone el uso de un algoritmo basado en el paradigma de ramificación y poda. El algoritmo propuesto generaliza distintas estrategias de búsqueda a las que se accede modificando los parámetros de entrada. El rendimiento de las distintas variantes de funcionamiento que presenta el algoritmo de búsqueda generalizado fue evaluado empíricamente.

Además de lo anterior, también se aborda una modificación del problema de la búsqueda que consiste en la obtención del mejor alineamiento a nivel de secuencias de palabras para un par de frases. Para resolver este nuevo problema se aplican técnicas de suavizado sobre los modelos de secuencias de palabras y se modifica el algoritmo de búsqueda definido inicialmente para que pueda explorar el espacio de posibles alineamientos. Resultados experimentales demuestran que las técnicas propuestas son capaces de generar alineamientos de forma eficiente y robusta.

Un inconveniente del uso de modelos de secuencias de palabras es su enorme tamaño cuando se estiman a partir de corpus muy grandes. En la tesis se proponen técnicas para aliviar este problema, tanto en la fase de estimación como en la fase de decodificación. Para ello se transforman requerimientos de memoria en requerimientos de disco duro. Resultados empíricos demuestran que los accesos a disco no degradan apreciablemente la eficiencia del sistema.

En el campo de la traducción interactiva desde un enfoque estadístico, se presentan, en

primer lugar, técnicas alternativas para implementar sistemas de traducción interactiva. En segundo lugar, también se describe una propuesta de sistema de traducción interactiva capaz de aprender de las traducciones validadas por el usuario mediante técnicas de aprendizaje online.

Con respecto a las técnicas alternativas de traducción interactiva, se proponen dos técnicas diferentes. La primera de ellas se basa en la generación de alineamientos parciales a nivel de secuencias de palabras. Este enfoque constituye una aplicación de la generación de alineamientos a nivel de secuencias de palabras también descrito en esta tesis. La segunda de las técnicas propuestas aborda el proceso de traducción interactiva con la ayuda de grafos de palabras y modelos correctores de errores estocásticos. El enfoque propuesto difiere de otros sistemas de traducción interactiva basados en grafos de palabras en que introduce el proceso de corrección de errores dentro del marco estadístico. Ambas técnicas de traducción interactiva se han evaluado mediante experimentos, demostrando ser competitivas con sistemas de traducción interactiva del estado del arte. Además, dichas técnicas se han usado para implementar un prototipo de traducción interactiva basado en la arquitectura cliente-servidor.

Finalmente, el sistema de traducción interactiva con aprendizaje online que se mencionaba anteriomente, se basa en el uso de modelos estadísticos actualizables de manera incremental. El principal obstáculo a la hora de obtener versiones incrementales de los modelos estadísticos involucrados en el proceso de traducción aparece cuando dichos modelos se estiman por medio del algoritmo *expectation-maximisation*. Para resolver este problema se propone la aplicación de la visión incremental de dicho algoritmo. El sistema de traducción interactiva con aprendizaje online fue evaluado experimentalmente, demostrándose que es capaz de aprender tanto a partir de modelos previamente estimados como de modelos vacíos. Los resultados de los experimentos también demuestran que el rendimiento del sistema que se propone es significativamente mejor que otros sistemas del estado del arte descritos en la literatura.

RESUM

Aquesta tesi presenta diverses contribucions als camps de la traducció automàtica estadística i la traducció interactiva des d'un enfocament estadístic.

Al camp de la traducció automàtica estadística, es presenten contribucions en relació als tres problemes fonamentals a abordar en aquesta disciplina: el problema del modelatge, el problema de l'entrenament i el problema de la cerca.

Respecte el problema del modelatge, es proposa una derivació alternativa dels models de seqüències de paraules. Aquesta derivació introdueix un conjunt de submodels probabilístics que governen diversos aspectes del procés de traducció. Addicionalment, els submodels que s'obtenen poden introduir-se com a components d'un model log-lineal.

Respecte al problema de l'entrenament, es descriu una tècnica alternativa d'estima-ció de models de seqüències de paraules que tracta de reduir la forta component heurística de les tècniques d'entrenament estàndard. La tècnica d'estimació proposada considera els parells de seqüències de paraules que componen el model com a part de bisegmentacions completes de les frases origen i destí. Es demostra tant teòrica com empíricament que la nova tècnica d'estimació pot executar-se eficientment. Resultats experimentals obtinguts amb la ferramenta d'estimació de lliure ús THOT presentada en aquesta tesi, demostren que la tècnica d'estimació proposada obté models amb menor perplexitat que els obtinguts amb la tècnica d'estimació estàndard. No obstant això, no s'han aconseguit millores en els resultats de traducció.

Per a abordar el problema de la cerca es proposa l'ús d'un algorisme basat en el paradigma de ramificació i poda. L'algorisme proposat generalitza distintes estratègies de cerca a les quals s'accedeix modificant els paràmetres d'entrada. El rendiment de les distintes variants de funcionament que presenta l'algorisme de cerca generalitzat s'avaluà empíricament.

A més a més, també s'aborda una modificació del problema de la cerca que consisteix en l'obtenció del millor alineament a nivell de seqüències de paraules per a un parell de frases. Per resoldre aquest nou problema s'apliquen tècniques de suavitzat sobre els models de seqüències de paraules i es modifica l'algorisme de cerca definit inicialment perquè puga explorar l'espai de possibles alineaments. Els resultats experimentals demostren que les tècniques proposades són capaces de generar alineaments de forma eficient i robusta.

Un inconvenient de l'ús de models de seqüències de paraules és el seu enorme tamany quan s'estimen a partir de corpus molt grans. En la tesi es proposen tècniques per a alleugerir aquest problema, tant en la fase d'estimació com en la fase de decodificación. Amb aquesta finalitat es transformen els requeriments de memòria en requeriments de disc dur. Els resultats empírics demostren que els accessos a disc no degraden apreciablement l'eficiència del sistema.

Al camp de la traducció interactiva des d'un enfocament estadístic, es presenten, en primer lloc, tècniques alternatives per a implementar sistemes de traducció interactiva. En segon lloc, també es descriu una proposta de sistema de traducció interactiva capa d'aprendre

de les traduccions validades per l'usuari mitjanant tècniques d'aprenentatge online.

Respecte a les tècniques alternatives de traducció interactiva, es proposen dues tècniques diferents. La primera d'elles es basa en la generació d'alineaments parcials a nivell de seqüències de paraules. Aquest enfocament constitueix una aplicació de la generació d'alineaments a nivell de seqüències de paraules també descrit en aquesta tesi. La segona de les tècniques proposades aborda el procés de traducció interactiva amb l'ajuda de grafs de paraules i models correctors d'errades estocàstics. L'enfocament proposat difereix d'altres sistemes de traducció interactiva basats en grafs de paraules en que introdueix el procés de correcció d'errades dins del marc estadístic. Ambdues tècniques de traducció interactiva s'han avaluat mitjanant experiments, demostrant ser competitives amb sistemes de traducció interactiva de l'estat de l'art. A més a més, aquestes tècniques s'han usat per a implementar un prototip de traducció interactiva basat en l'arquitectura client-servidor.

Finalment, el sistema de traducció interactiva amb aprenentatge online que es mencionava anterioment, es basa en l'ús de models estadístics actualizables de manera incremental. El principal obstacle a l'hora d'obtenir versions incrementales dels models estadístics involucrats en el procés de traducció apareix quan aquests models s'estimen per mitjà de l'algorisme expectation-maximisation. Per a resoldre aquest problema es proposa l'aplicació de la visió incremental d'aquest algorisme. El sistema de traducció interactiva amb aprenentatge online s'avaluà experimentalment, demostrant-se que és capa d'aprendre tant a partir de models prèviament estimats com de models buits. Els resultats dels experiments també demostren que el rendiment del sistema que es proposa és significativament millor que altres sistemes de l'estat de l'art descrits en la literatura.

PREFACE

Natural language processing (NLP) is the computerised approach to generating and understanding human languages, both oral or written. The goal of NLP is to accomplish human-like language processing for a range of tasks or applications. NLP is a field of artificial intelligence, and its origins can be found in the disciplines of linguistics, computer science and cognitive psychology. In the field of NLP there are two distinct focuses, namely, language processing and language generation. The first of these refers to the analysis of language for the purpose of producing a meaningful representation, while the latter refers to the production of language from a representation. Natural language processing provides both theory and implementations for a range of applications, including information retrieval, information extraction, question-answering, summarisation, machine translation, dialogue systems, etc.

This thesis explores the area of machine translation (MT), which was the first computer-based application related to natural language. The discipline of MT investigates the use of computer software to translate text or speech from one language to another. The first proposals for MT using computers dates back to the 1950's, and were based on information theory, expertise in breaking enemy codes during the second world war and speculation about the underlying principles of natural language. Even after more than 50 years of research, MT remains an open problem.

Current MT systems use different translation technologies. These translation technologies can be classified into two main approaches: rule-based systems and corpus-based systems. Rule-based systems use a set of translation rules created by human translators to generate their output. These rules determine how to translate from one language to another. Corpus-based systems make use of sets of translation examples (also called corpus or parallel texts) from one language to another. The translation examples are used to infer the translation of the source text.

This thesis approaches MT under the statistical framework. Statistical MT (SMT) systems are a kind of corpus-based MT systems that use parallel texts to estimate the parameters of the statistical models involved in the translation process. Once the statistical models have been estimated, they are used to infer the translation of new sentences. Different statistical translation models have been proposed since the beginning of the research in SMT. The IBM models were the first statistical translation models used in SMT. In these models, the fundamental unit of translation is a word in a given language. This restricted conception of the translation process does not allow to obtain good translation results due to its inability to capture context information. To solve this problem, single words were replaced as the fundamental unit of translation by multiple words in a new family of statistical translation models. Among the different multi-word statistical translation models that have been proposed so far, the so-called phrase-based models currently constitute the state-of-the-art in SMT. Phrase-based models work by translating sequences of words or phrases. These phrases are not linguistically motivated; instead, they are extracted from corpora using statistical methods.

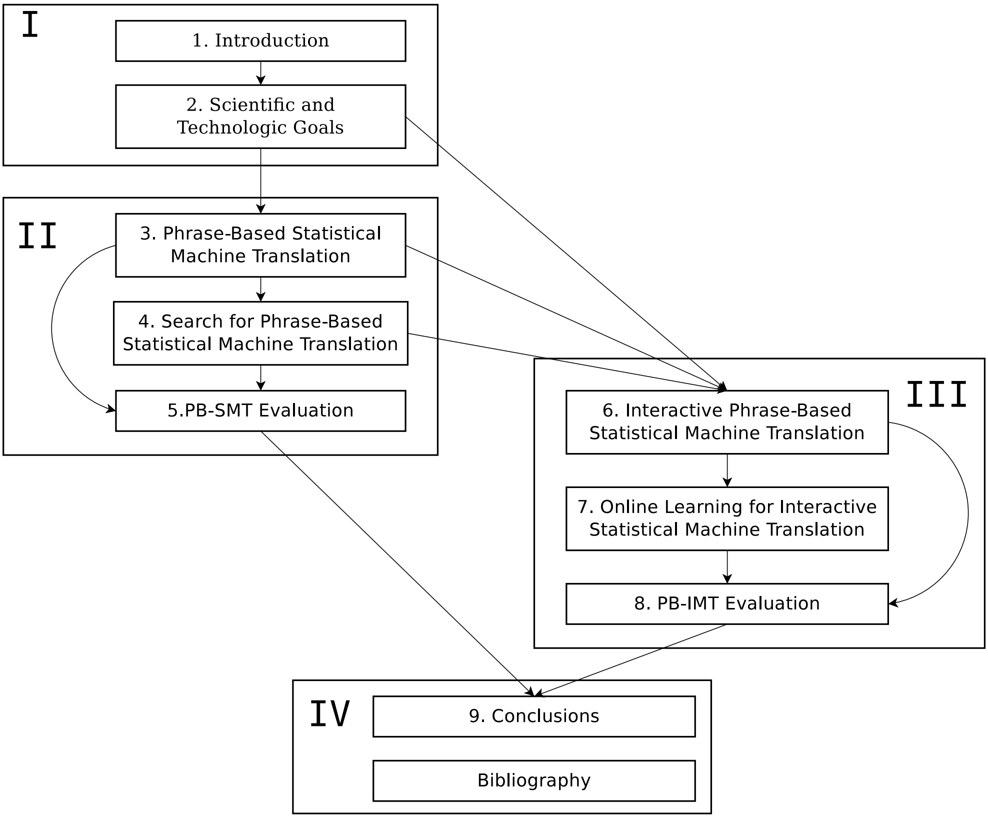
Current MT systems are not able to produce ready-to-use texts. Indeed, MT systems usually require human post-editing in order to achieve high-quality translations. This motivates an alternative application of MT in which the MT system collaborates with the user to generate the output translations. This alternative application receives the name of computer-assisted translation (CAT). The canonical example of CAT system is represented by the so-called memory-based machine translation (MBMT) systems. MBMT systems store user-validated translations (translation memories) for its reuse in the translation of similar texts. These reused translations are post-edited by the user so as to generate the target text. However, CAT is a broad and imprecise term covering a range of tools. In this thesis, we will focus on a specific instantiation of the CAT paradigm which receives the name of interactive machine translation (IMT). In the IMT framework, the user obtains her desired translations in a series of interactions with the MT system. IMT differs from post-editing CAT techniques in its capability to take advantage of the knowledge of the human translator (it should be noticed that when applying post-editing CAT techniques, the CAT system and its user work in separated serial processes).

The scientific goals of this thesis can be divided into two groups as follows:

1. **Fully automatic phrase-based SMT**. We develop contributions regarding the three problems that are to be addressed in SMT, namely, the modelling, the estimation and the search problems. With respect to the modelling problem, we propose an alternative phrase-based model derivation that allows us to obtain a set of probabilistic models governing different aspects of the translation process. Regarding the estimation problem, we describe a new estimation procedure that tries to reduce the strong heuristic component of the standard estimation algorithm. Both the new and the standard estimation techniques were implemented in a publicly available toolkit called THOT which is also presented in this thesis. With respect to the search problem, we describe a search algorithm that is based in the branch-and-bound paradigm. The proposed search algorithm generalises a set of search strategies that can be accessed by only modifying the input parameters of the algorithm. In addition to this, we also study a modification of the search problem that consists in the generation of alignments at phrase level. Finally, one important disadvantage of phrase-based models is their huge size when estimated from very large corpora. We propose techniques to alleviate this problem during both the estimation and the decoding stages.

2. **Interactive phrase-based SMT**. We propose two novel IMT techniques. The first one constitutes an application of the phrase-level alignment generation techniques that were studied for fully automatic phrase-based SMT. The second IMT technique combines phrase-based translation models and stochastic error-correction models in a unified statistical framework. In addition to this, we describe an IMT system able to learn from user feedback by means of online learning techniques.

This thesis is structured in four parts, containing a total of nine chapters plus the bibliography section. In the following figure we show the dependencies between chapters:


I
1. Introduction
2. Scientific and Technologic Goals
II
3. Phrase-Based Statistical Machine Translation
4. Search for Phrase-Based Statistical Machine Translation
5.PB-SMT Evaluation
III
6. Interactive Phrase-Based Statistical Machine Translation
7. Online Learning for Interactive Statistical Machine Translation
8. PB-IMT Evaluation
IV
9. Conclusions
Bibliography

The content of each chapter is as follows:

Chapter 1 introduces the disciplines of NLP and MT. Among the different translation technologies used in MT, this chapter is focused on the SMT framework, briefly introducing the main SMT techniques that have been described in the literature. In addition to this, the discipline of IMT is also introduced. Finally, we present the automatic evaluation measures and the main features of the bilingual corpora that were used to empirically evaluate the proposals presented in this thesis.

Chapter 2 presents the list of scientific and technologic goals of this thesis. These goals are classified into fully automatic phrase-based SMT goals and interactive phrase-based SMT goals.

Chapter 3 describes different proposals regarding both the modelling and the estimation problem in SMT. First, a novel estimation technique for phrase-based models that tries to reduce the strong heuristic component of the standard estimation technique is presented. Second, we describe techniques to deal with very large corpora during the estimation of phrase-based models. Finally, we show a specific derivation for phrase-based models.

Chapter 4 presents different proposals regarding the search problem in SMT. First, a generalised search algorithm based on the branch-and-bound paradigm is defined. Second we describe techniques to deal with huge phrase-based models during the search stage. Finally, we study a modification of the search problem that consists in the generation of alignments at phrase level.

Chapter 5 presents the empirical results of the evaluation of the fully automatic phrase-based SMT techniques presented in chapters 3 and 4.

Chapter 6 describes two novel IMT techniques. The first one is based on the phrase-level alignment generation techniques described in Chapter 4. The second one combines phrase-based models and stochastic error-correction models in a unified statistical framework.

Chapter 7 describes an IMT system able to learn from user feedback by means of online learning techniques.

Chapter 8 presents the empirical results of the evaluation of the phrase-based IMT techniques presented in chapters 6 and 7.

Chapter 9 presents a summary of the work presented in this thesis, including a list of scientific publications, followed by a list of future directions for further developments of the work presented here.

The previous content is complemented by a set of appendices:

Appendix A shows a detailed derivation of the incremental expectation-maximisation algorithm for the HMM-based word alignment model. The results of this derivation are used in Chapter 7.

Appendix B presents the open-source THOT toolkit for statistical machine translation. The THOT toolkit has been used to carry out the experiments presented in this thesis.

Appendix C describes the main features of a web-based IMT prototype that has been developed following the techniques proposed in this thesis.

Appendix D presents the list of mathematical symbols and acronyms used in this thesis.

Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

Part I

Introduction and Goals

CHAPTER 1
INTRODUCTION

1.1 Natural Language Processing

Natural language processing (NLP) is the computerised approach to generating and understanding human languages, both oral or written. The goal of NLP is to accomplish human-like language processing for a range of tasks or applications. NLP is a field of artificial intelligence, and its origins can be found in the disciplines of linguistics, computer science and cognitive psychology. In the field of NLP there are two distinct focuses, namely, language processing and language generation. The first of these refers to the analysis of language for the purpose of producing a meaningful representation, while the latter refers to the production of language from a representation. Natural language processing provides both theory and implementations for a range of applications, including information retrieval, information extraction question-answering, summarisation, machine translation, dialogue systems, etc.

This thesis explores the area of machine translation (MT), which was the first computer-based application related to natural language. The discipline of MT investigates the use of computer software to translate text or speech from one language to another.

1.2 Machine Translation

Multiplicity of languages is inherent to modern society. Phenomena such as the globalisation and technological development have dramatically increased the need of translating information from one language to another. This necessity can be found in different fields including political institutions, industry, education or entertainment. A good example of multilingualism can be found in the European Union (EU) political institutions. The EU has 27 Member States and 23 official languages. Translation in the European institutions concerns legislative, policy and administrative documents. According to [Com08], in 2008 the EU employed 1750 translators working full time on translating documents and on other language-related tasks. To cope with a level of demand that fluctuates in response to political imperatives, the EU used external translation providers which generated approximately the fourth part of the EU translation output. The EU also maintained a web translation unit specialised in the translation of web pages. As a result, in 2008 the EU translation services translated more than 1 800 000 pages and spent about 1000 million Euros on translation and interpreting.

The high demand of translations cannot be completely satisfied by human translators, motivating a great interest in the development of machine translation (MT) techniques. The aim of MT is to carry out the translation process from one language to another by means of a computer. MT techniques are specially useful to translate formal documents such as manuals, official reports, financial reports, etc. where the strict adherence to layout and stylistic rules is considered important to produce high quality translations. MT has gained more and more importance in the last years and is already being used by companies and political institutions.

The first proposals for MT using computers dates back to the 1950's, and were based on information theory, expertise in breaking enemy codes during the second world war and speculation about the underlying principles of natural language. Early work in MT took the simplistic view that the only differences between languages resided in their vocabularies and the permitted word orders. The results obtained following these principles were poor, since the proposed MT systems simply used word dictionaries to select the appropriate words for translation and reordered the resulting words following the word-order rules of the target language. Even after more than 50 years of research, MT remains an open problem.

In the following sections we will briefly describe the main strategies that have been historically applied to tackle the problem of MT.

1.2.1 MT Systems Taxonomy

The different approximations to the MT problem can be classified using different criteria:

1. Depending on the type input: text or speech.

2. Depending on the type of the application which uses the translations. These applications can be divided into four different groups: applications that translate the input into a database query; applications that produce an approximated translation of the input for its correction in a post-edition stage by the user; applications that interactively generate the output in collaboration with the user; and finally, fully automated translation systems. Currently, fully automated translation systems can only work on restricted domains.

3. Depending on the translation technology. We can identify two main approaches: rule-based systems and corpus-based systems. In spite of the fact that these systems use opposite technologies, a number of proposals combining both approaches can be found in the literature.

The vast majority of the introductory works on MT [Hob92, HS92, Som98, Tru99, Lop08] classify MT systems depending on the translation technology that these systems use. In the following sections, we will focus on the different translation technologies that have been proposed so far.

1.2.2 Rule-Based Systems

Rule-based systems use a set of translation rules created by human translators to generate their output. These rules determine how to translate from one language to another. The process of creating the translation rules is very costly and requires the knowledge provided by expert linguists in both the source and the target languages. Regular rule-based systems execute two different steps to generate their translations, namely, the analysis step and the generation step. The analysis step extracts information from the source text. Once the analysis step has been executed, the generation step produces the target text. One extra step can be introduced between the analysis and the generation steps: the transference step. The transference step transforms the result of the analysis step into an abstract, less language-specific representation. A particular case of the transfer step uses an *interlingua*, i.e., an abstract language representation. The target sentence is then generated from the interlingua. The use of an interlingua requires a deep analysis of the source text, in addition to this, there is no language that is globally accepted for its use as an interlingua; by these reasons not all rule-based systems use it. To avoid the necessity of an interlingua, the transference step is executed. The transference step allows to reduce the complexity of the analysis step. The previous considerations can be depicted as a diagram by means of the so-called *Vauquois triangle* [Vau75] (see Figure 1.1).

Rule-based systems can be classified according to the importance assigned to the analysis and transference steps. Under this criterion, we find three different rule-based approaches, namely, the direct approach, the transfer approach and the interlingual approach.

Direct Approach

The direct approach is the translation strategy adopted by the first machine translation systems that were proposed. The direct approach uses a word-to-word translation strategy including a morpho-syntactic analysis of the source text. The morpho-syntactic analysis tries to capture grammar categories and other morphological information, but excludes relationships between words or groups of words.

Transfer Approach

This approach first generates a logical representation of the source text. Once this logical representation has been generated, a set of transfer rules is applied to obtain its equivalent representation in the target language. Finally, the target text is generated from the target language logical representation.

Interlingual Approach

The Interlingual approach first performs a deep analysis of the source text. As a result of this analysis, an abstract language representation is obtained. This representation is called interlingua and is independent of both the source and the target languages. After obtaining this conceptual representation, the translation is generated; in other words, the source text is first understood and then translated. The interlingual approach has one advantage with respect to the previous approaches, specifically, we only need to define a correspondence between each language and the interlingua instead of defining correspondences between each language pair.

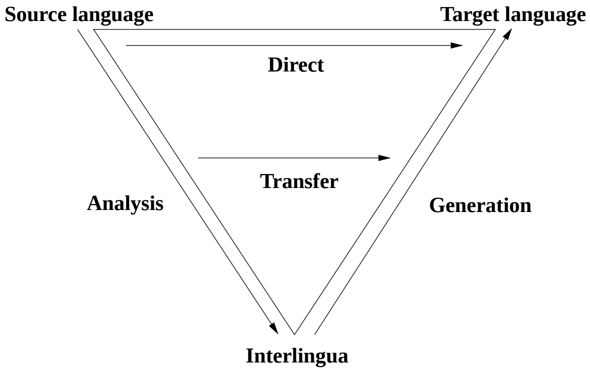


Figure 1.1: Vauquois triangle.

1.2.3 Corpus-Based Systems

Corpus-based systems make use of the so-called empirical approaches to MT. The main feature of corpus-based systems is that they use sets of translation examples (also called corpus or parallel texts) from one language to another. The translation examples are used to infer the translation of the source text. Once a corpus-based system has been implemented, the software can be quickly adapted for its use with different language pairs or different domains, as opposed to rule-based systems, which are specific for a given language pair.

Corpus-based systems can be classified into two groups: the example-based machine translation systems and the statistical machine translation systems. Additionally, there exist other approaches that are different to the previous ones and will also be mentioned in later sections.

Example-Based Machine Translation (EBMT) Systems

The example-based approach to machine translation uses a set of translation examples as its main knowledge base. EBMT systems execute two steps to generate their translations, namely, the comparison and the recombination steps: first, a set of hypotheses that are similar to the source text is extracted from the whole corpus (comparison). Second, the hypotheses are recombined to generate the final translation of the source text (recombination).

One important translation technology derived from the example-based approach to MT is the so-called memory-based machine translation (MBMT). MBMT allows to assist human translators in the translation of texts. MBMT stores user-validated translations (translation memories) for its reuse in the translation of similar texts.

Statistical Machine Translation (SMT) Systems

The statistical approach to MT requires the availability of a great amount of parallel text containing relevant information for the translation process. This parallel text is used to estimate the parameters of a set of statistical models involved in the translation process. Once the statistical models have been estimated, they are used to infer the translation of new source sentences.

Some authors [Som98] classify the statistical approach into the example-based approach because of the necessity of parallel text to estimate the statistical models. However, the statistical approach differs from the example-based approach because of the different way in which the comparison-recombination steps described above are implemented. Specifically, the statistical approach is focused on statistical parameter estimation. By this reason, other authors consider that the statistical approach can be classified as a separate approach.

In the first works on SMT, the statistical models had to be simplified. Specifically, a simplified grammar was used instead of a complete grammar of the target language. The transfer rules were replaced by two different statistical models, namely, a model that captures lexical relationships between source and target words; and a model that captures the relationships between the positions of the words of the source and target sentences.

More recently, and due to the great increase in the linguistic resources, better and more complex statistical models have been obtained. This will be explained with more detail in section 1.3.

Other Corpus-Based Systems

There exit alternative approaches that can be followed to implement corpus-based systems. Examples of these alternative approaches are the connectionist approach, the finite state approach and the synchronous context free grammars approach. The connectionist approach uses artificial neural networks to tackle the problem of MT, and some authors consider that it is a subclass of the statistical approach. The finite state approach to machine translation uses the mathematical tools provided by the automata theory. Finally, the synchronous context free grammars approach applies context free grammars to MT. Both the finite state approach and the synchronous context free grammars approach can also be classified into the statistical approach.

1.3 Statistical Machine Translation

The statistical approach to MT formalises the problem of generating translations under a statistical point of view. This approach is classified into the corpus-based approaches, as was explained in section 1.2.3. The availability of corpora, specifically parallel-texts, is required to estimate the parameters of the statistical models involved in the translation process. Such statistical models can be described as a *mathematical theory* about how a sentence in the source language is translated into its equivalent in the target language. It is worthy of note that one important advantage of the SMT systems is their ability to work with different language pairs if the corresponding parallel texts to estimate the parameters of the statistical models are available.

More formally, given a source sentence $f_1^J \equiv f_1...f_j...f_J$ in the source language $\mathcal{F}$, we want to find its equivalent target sentence $e_1^I \equiv e_1...e_i...e_I$[a] in the target language $\mathcal{E}$. From the set of all possible sentences of the target language, we are interested in that with the highest probability according to the following equation:

$$\hat{e}_1^{\hat{I}} = \arg\max_{I,e_1^I}\{Pr(e_1^I|f_1^J)\} \tag{1.1}$$

The early works on SMT were based on the use of *generative models*. A generative model is a full probability model of all statistical variables that are required to randomly generating observable data. Generative models decompose $Pr(e_1^I|f_1^J)$ applying the Bayes decision rule. Taking into account that $Pr(f_1^J)$ does not depend on e_1^I we arrive to the following expression:

$$\hat{e}_1^{\hat{I}} = \arg\max_{I,e_1^I}\{Pr(e_1^I) \cdot Pr(f_1^J|e_1^I)\} \tag{1.2}$$

This equation can be seen as a representation of the process by which a linguist translate a source sentence into its equivalent target sentence. Obtaining the final translation requires the exploration of all target language sentences, calculating the probability $Pr(e_1^I)$ for each sentence e_1^I and the conditional probability $Pr(f_1^J|e_1^I)$. After the exploration is completed, we return the translation $\hat{e}_1^{\hat{I}}$ of highest probability. This corresponds to the so-called *noisy channel model*. In the noisy channel model, the sentence of $\mathcal{E}$ is obtained by transmitting the source sentence of $\mathcal{F}$ through a *noisy channel*. This noisy channel has the property of transforming the sentences of $\mathcal{F}$ into their equivalent in the language $\mathcal{E}$.

Equation (1.2) is the so-called *fundamental equation of machine translation* [BDDM93], where: $Pr(e_1^I)$ represents the probability of generating the target sentence, and $Pr(f_1^J|e_1^I)$ is the probability of generating e_1^I given f_1^J . Since the real probability distributions $Pr(e_1^I)$ and $Pr(f_1^J|e_1^I)$ are not known, they are approximated by means of parametric statistical models. Typically, the values of the parameters of such statistical models are obtained by means of the well-known *maximum-likelihood* estimation method.

Statistical parametric models have a set of parameters Θ associated with either a known probability density function or a probability mass function, denoted as $p(\cdot|\Theta)$. Given a set of training samples $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$, the *log-likelihood function* is defined as the logarithm of the probability density associated with the given observed data:

$$\mathcal{L}(\Theta, \mathbf{x}) = \log p(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N|\Theta) = \sum_{i=1}^{N} \log p(\mathbf{x}_i|\Theta) \tag{1.3}$$

The method of maximum-likelihood estimates Θ by finding the value of Θ that maximises $\mathcal{L}(\Theta, \mathbf{x})$. This is the maximum-likelihood (ML) estimator of Θ:

$$\hat{\Theta} = \arg\max_{\Theta}\{\mathcal{L}(\Theta, \mathbf{x})\} \tag{1.4}$$

Typically, the two distributions that appear in Equation (1.2) are modelled separately. Specifically, the probability distribution $Pr(e_1^I)$ is modelled by means of a *language*

[a] f_j and e_i note the i'th word and the j'th word of the sentences f_1^J and e_1^I respectively.

model and $Pr(f_1^J|e_1^I)$ is modelled by means of a *translation model*. Therefore, we have to find two sets of parameters Θ_{LM} and Θ_{TM} corresponding to the language and the translation models respectively. Given the training set composed of sentence pairs $\mathcal{X} = \{(\mathbf{f}_1, \mathbf{e}_1), (\mathbf{f}_2, \mathbf{e}_2), ..., (\mathbf{f}_N, \mathbf{e}_N)\}$, and following the ML criterion we arrive to the following expressions:

$$\hat{\Theta}_{LM} = \arg\max_{\Theta_{LM}} \left\{ \sum_{n=1}^{N} \log p(\mathbf{e}_n|\Theta_{LM}) \right\} \tag{1.5}$$

$$\hat{\Theta}_{TM} = \arg\max_{\Theta_{TM}} \left\{ \sum_{n=1}^{N} \log p(\mathbf{f}_n|\mathbf{e}_n; \Theta_{TM}) \right\} \tag{1.6}$$

It is worth pointing out that the translation process can also be carried out by directly modelling the posterior probability $Pr(e_1^I|f_1^J)$. This is explained with more detail in section 1.4.4.

SMT can be viewed as a specific instance of a classification problem where the object to be classified is the source sentence f_1^J to be translated and the set of possible classes are the set of possible sentences in the target language e_1^I. Therefore, under this point of view the decision rule stated in Equation (1.2) is optimal under the assumption of a zero-one loss function. In SMT, the zero-one loss function is better known as sentence error rate (SER) and considers that there is an error if the translation given by the system is not identical to the reference translation. Therefore, by applying Equation (1.2) we are minimising the probability of error using SER as a loss function. Although SER has been commonly used as loss function in several works on SMT, alternative proposals of loss functions can be found in the literature (for more details see [AFOMGVC08]).

Figure 1.2 shows the architecture of the translation process using the Bayes rule [NNO$^+$00]. As it is shown in the figure, the translation process requires four different modules:

- Translation model ($p(f_1^J|e_1^I)$): the translation model measures how good f_1^J is as a translation of e_1^I. Regular translation models include a lexical submodel (also called statistical dictionary) which assigns probabilities to the translation of target words e_j by source words f_i. Translation models also include an alignment submodel which assigns probabilities to the relationships between the word positions of the source and the target sentences.

- Language model ($p(e_1^I)$): This model measures the well-formedness of the sentence e_1^I as a sentence of the language $\mathcal{E}$.

- Global search: this module executes the translation process. For this purpose, Equation (1.2) is solved, obtaining the target sentence $\hat{e}_1^{\hat{I}}$ of highest probability given by both the translation and the language models.

- Pre/postprocess: the pre/postprocessing stages comprise a series of input/output transformations that are useful to increase the performance of the translation system.

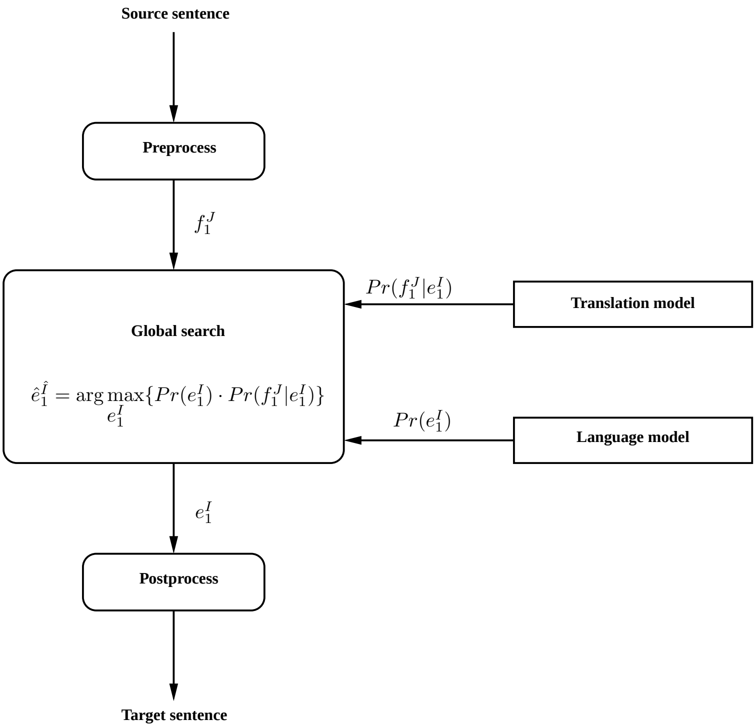


Figure 1.2: Architecture of the translation process using the Bayes rule.

The building process of an SMT system following the Bayes decision rule involves addressing three problems [Ney01]:

1. the *modelling problem*, that is, how to structure the dependencies of source and target language sentences.

2. the *training problem*, that is, how to estimate the model parameters given the training data.

3. the *search problem*, that is, how to find the best translation candidate among all possible target language sentences.

1.4 Statistical Models for Machine Translation

The above mentioned modelling problem involves finding good approximations for the two probability distributions that are shown in Equation (1.2).

To approximate the probability distribution $Pr(e_1^I)$ which is shown in Equation (1.2), the vast majority of the works on SMT in the literature use the so-called *statistical n-gram* *language models*. Regarding the probability distribution $Pr(f_1^J|e_1^I)$, it is approximated by a statistical translation model. In the following sections we will briefly explain the statistical models that are commonly used in SMT.

1.4.1 n-gram Language Models

Statistical language models are formulated as a probability distribution $p(\mathbf{x})$ for strings $\mathbf{x}$. This probability distribution tries to reflect how frequently does the string $\mathbf{x}$ appear.

The most widely used statistical language models are, by far, the n-gram language models. We will introduce the n-gram language models considering that $n = 2$; these models are the so-called *bigram* language models. Let us consider the sentence $\mathbf{x}$ composed of the words $x_1 x_2 ... x_{|\mathbf{x}|}$[b], we can express $Pr(\mathbf{x})$ without loss of generality as follows:

$$Pr(\mathbf{x}) = p(x_1) \cdot p(x_2|x_1) \cdot p(x_3|x_1 x_2) \cdot ... \cdot p(x_{|\mathbf{x}|}|x_1 ... x_{|\mathbf{x}|-1}) = \prod_{i=1}^{|\mathbf{x}|} p(x_i|x_1 ... x_{i-1}) \quad (1.7)$$

Bigram models assume that the probability of a given word depends only on the immediately preceding word:

$$Pr(\mathbf{x}) = \prod_{i=1}^{|\mathbf{x}|} p(x_i|x_1 ... x_{i-1}) \approx \prod_{i=1}^{|\mathbf{x}|} p(x_i|x_{i-1}) \quad (1.8)$$

The special token BOS which denotes the begin of a sentence is introduced so that x_0 is BOS. Additionally, to make $\sum_w p(w) = 1$ is necessary to add another special token EOS which denotes the end of a sentence.

To estimate $p(x_i|x_{i-1})$, the frequency of the word x_i given the previous word x_{i-1}, we count the number of occurrences of the bigram $x_{i-1} x_i$ in the training text and normalise, which corresponds to a ML estimation:

$$p(x_i|x_{i-1}) = \frac{c(x_{i-1} x_i)}{\sum_{x_i} c(x_{i-1} x_i)} \quad (1.9)$$

Where $c(x_i)$ is the count of the number of occurrences of the word x_i in the source text.

For n-grams with $n > 2$, instead of conditioning the probability of a word on the identity of just the preceding word, we condition this probability on the identity of the last $n - 1$ words. Here we take x_{-n+2} through x_0 to be BOS and $x_{|\mathbf{x}|+1}$ to be EOS. Thus, the sentence probability is calculated as follows:

$$p(\mathbf{x}) = \prod_{i=1}^{|\mathbf{x}|+1} p(x_i|x_{i-n+1} ... x_{i-2} x_{i-1}) \quad (1.10)$$

Regarding the estimation of the n-gram probabilities, the corresponding equation is very similar to Equation (1.9) for bigram models:

$$p(x_i|x_{i-n+1}^{i-1}) = \frac{c(x_{i-n+1}^{i-1})}{\sum_{x_i} c(x_{i-n+1}^{i})} \quad (1.11)$$

[b] $|\mathbf{x}|$ represents the length of the string $\mathbf{x}$.

Where x_{i-n+1}^{i-1} denotes the segment of the source sentence which starts at the $(i-n+1)$'th word and finishes at the $(i-1)$'th word.

The words x_{i-n+1}^{i-1} are usually called *history* of the n-gram. n is called *order* of the n-gram. In the literature on SMT, the value of n has typically been set to 3. These are the so-called *trigram* language models. More recently, and due to the availability of larger and larger training corpora, n is set to 5.

A wide variety of the n-gram language model estimation techniques described in the literature are implemented in the SRILM toolkit [Sto02]. The SRILM toolkit also provides tools and code to access the parameters of previously estimated language models.

Complexity Measures

Some measures to judge the performance of a language model have been proposed in the literature (see [Ros00]). The simplest one of this measures is the *average log-likelihood* of the test samples. The average log-likelihood for the test set is given by the following expression:

$$\mathrm{AL}(\mathcal{X}|\Theta) = \frac{1}{N}\sum_{i=1}^{N}\log p(\mathbf{x}_i|\Theta) \tag{1.12}$$

where $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ are the test samples and Θ is the set of language model parameters. This measure can be seen as an empirical estimation of the *cross entropy* of the real probability distribution (but unknown) Pr with respect to the model distribution given by Θ:

$$\mathrm{H}(Pr; p(\cdot|\Theta)) = -\sum_{i=1}^{N} Pr(\mathbf{x}_i)\cdot\log p(\mathbf{x}_i|\Theta) \tag{1.13}$$

The most widely used performance measure for statistical language models is the so-called *perplexity*:

$$\mathrm{PP}(Pr; p(\cdot|\Theta)) = 2^{\mathrm{H}(Pr; p(\cdot|\Theta))} \tag{1.14}$$

The perplexity of a language model given a test set can be interpreted as the geometric average of the branching factor of the given language with respect to the model. The perplexity measures both the model performance and the language complexity.

An alternative way to measure the language model performance is to measure its impact in the specific application in which this language model is used. Typically, the lower the perplexity the lower the error rates of the application. As an informal rule, [Ros00] states that a 5% perplexity reduction does not produce significant improvements; a reduction between 10% and 20% may produce an appreciable improvement and finally, a reduction above 30% usually produces a great improvement in the error rates.

n-gram Model Smoothing

The above described n-gram language models cannot assign probabilities greater than zero to events that have not been seen during the training process. To solve this problem, the n-gram language model probabilities are modified using smoothing techniques. The term "smoothing" comes from the aim of these techniques. Specifically, the smoothing techniques try to obtain more uniform probability distributions, increasing the probabilities equal or almost equal to zero and decreasing the probabilities equal or almost equal to one. For this purpose, a certain probability mass is *discounted* from certain events and added to others. The smoothing techniques not only allow to avoid events with null probability, but also significantly improve the performance of n-gram language models.

Several smoothing techniques have been proposed in the literature such as the Jelinek-Mercer smoothing, the Katz smoothing, the Witten-Bell smoothing, etc. (see [CG96] for more details). Although we will not describe the smoothing techniques in detail, the vast majority of them can be summarised by means of the following expression [KN95]:

$$p_s(x_i|x_{i-n+1}^{i-1}) = \begin{cases} \alpha(x_i|x_{i-n+1}^{i-1}) & if \ c(x_{i-n+1}^{i}) > 0 \\ \gamma(x_{i-n+1}^{i-1}) \cdot p_s(x_i|x_{i-n+2}^{i-1}) & if \ c(x_{i-n+1}^{i}) = 0 \end{cases} \tag{1.15}$$

According to Equation (1.15) if a given n-gram has been seen during the training stage, we use the distribution $\alpha(x_i|x_{i-n+1}^{i-1})$; otherwise, we use the lower order *backoff* distribution $p_s(x_i|x_{i-n+2}^{i-1})$, where the scale factor $\gamma(x_{i-n+1}^{i-1})$ is introduced to make the conditional distribution sum up to one. All the models that can be described in this way are called *backoff models*, and the scale factor γ is usually called *backoff weight*. The canonical example of backoff smoothing is the so-called Katz smoothing.

Finally, there is another kind of smoothing algorithms that can be expressed as a linear interpolation of higher and lower order n-grams:

$$p_s(x_i|x_{i-n+1}^{i-1}) = \lambda_{x_{i-n+1}^{i-1}} \cdot p_{ML}(x_i|x_{i-n+1}^{i-1}) + (1 - \lambda_{x_{i-n+1}^{i-1}}) \cdot p_s(x_i|x_{i-n+2}^{i-1}) \tag{1.16}$$

Where $p_{ML}(\cdot)$ is the n-gram probability estimated by means of the ML criterion and $\lambda_{x_{i-n+1}^{i-1}}$ and $(1 - \lambda_{x_{i-n+1}^{i-1}})$ are the interpolation weights. This kind of language models receives the name of *interpolated language models*.

1.4.2 Single-Word Alignment Models

The IBM models [BDDM93] were the first alignment models used in SMT. The IBM models were developed at the IBM T.J. Watson research institution. In spite of the fact that the IBM models no longer constitute the state of the art in SMT, they are still used in statistical machine translation for different purposes. There are five different types of IBM models, ranging from the IBM 1 Model to the IBM 5 Model. The IBM models are based on the concept of alignment between the words of the source and the target sentences (f_1^J, e_1^I).

Formally, an alignment is a correspondence between word positions of the source and the target sentences f_1^J and e_1^I: $a \subset \{1 \cdots J\} \times \{1 \cdots I\}$. However, in [BDDM93], the alignments are restricted to be functions $a : \{1 \cdots J\} \rightarrow \{0 \cdots I\}$, where $a_j = i$ if the j'th source position is aligned with the i'th target position. Additionally, $a_j = 0$ notes that the word position j of f_1^J has not been aligned with any word position e_1^I (or that it has been aligned with the *null word* e_0). Let $\mathcal{A}(f_1^J, e_1^I)$ be the set of all possible alignments between e_1^I and f_1^J, and let $Pr(f_1^J, a_1^J|e_1^I)$ be the probability of translating f_1^J by e_1^I given the alignment hidden variable a_1^J. We formulate $Pr(f_1^J|e_1^I)$ as follows:

$$Pr(f_1^J|e_1^I) = \sum_{a_1^J \in \mathcal{A}(f_1^J, e_1^I)} Pr(f_1^J, a_1^J|e_1^I) \tag{1.17}$$

Under a generative point of view, $Pr(f_1^J, a_1^J|e_1^I)$ can be decomposed without loss of generality as follows:

$$Pr(f_1^J, a_1^J|e_1^I) = Pr(J|e_1^I) \cdot \prod_{j=1}^{J} Pr(f_j|f_1^{j-1}, a_1^j, e_1^I) \cdot Pr(a_j|f_1^{j-1}, a_1^{j-1}, e_1^I) \tag{1.18}$$

Given this general equation and depending on the specific type of IBM model, we make different assumptions. The first assumption is common to the five model types. Specifically, the probability distribution $Pr(f_j|f_1^{j-1}, a_1^j, e_1^I)$ is approximated by $p(f_j|e_{a_j})$ (which constitutes an statistical dictionary of words). The different IBM models differ in the assumptions that are made over the alignment probabilities $Pr(a_j|f_1^{j-1}, a_1^{j-1}, e_1^I)$. These differences are briefly described here:

- **IBM 1 Model**: uniform alignment probabilities are assumed.
- **IBM 2 Model**: $Pr(a_j|f_1^{j-1}, a_1^{j-1}, e_1^I)$ is approximated by $p(a_j|i, J, I)$, a zero-order model which establishes dependencies between absolute word positions of the source and the target sentences.
- **IBM 3 Model**: a *fertility model* $p(\phi|e)$ is added (representing the probability that the word e generates ϕ source words). The alignment probability for the IBM 3 Model is approximated by a zero-order model called *distortion model* $p(i|a_j, J, I)$, which establishes dependencies between absolute word positions of the source and the target sentences.
- **IBM 4 and IBM 5 models**: they use a distortion model with first-order dependencies between the relative word positions of the source and the target sentences.

IBM model parameter estimation is carried out by means of the *expectation-maximisation* (EM) algorithm (refer to section 1.5.1 for more details). The specific details of the application of the EM algorithm to estimate the IBM model parameters can be found in [BDDM93]. In addition to this, there exists a publicly-available software tool that allows to estimate IBM model parameters. This package is the so-called GIZA++ toolkit [Och00].

The search problem using IBM models is formalised following Equation (1.2); alternatively, the so-called *maximum-approximation* can also be used:

$$\hat{e}_1^{\hat{I}} = \arg\max_{e_1^I} \left\{Pr(e_1^I) \cdot \sum_{a_1^J} Pr(f_1^J, a_1^J|e_1^I)\right\} \approx \arg\max_{e_1^I}\{Pr(e_1^I) \cdot \max_{a_1^J} Pr(f_1^J, a_1^J|e_1^I)\} \tag{1.19}$$

In the maximum-approximation, the maximisation process is carried out obtaining the probability of the best alignment for the source and the target sentences. The best alignment for a given sentence pair is often referred to as the *Viterbi alignment*.

Apart from the IBM models, other single-word alignment models have been proposed. The most important of those single-word alignment models are the hidden Markov model (HMM) based alignment models [VNT96]. The HMM-based alignment models are similar to IBM models, specifically, they use a statistical dictionary of words and a first order alignment model $p(a_j|a_{j-1}, I)$. HMM-based alignment models have been extended in different works [TIM02, DB05].

1.4.3 Multi-Word Alignment Models

The main disadvantage of the single-word alignment models is their inability to capture context information. Because of this, when single-word models are used, the responsibility of capturing context information exclusively lies on the language model. One possible solution to this problem consists in the definition of translation models that establish relationships between groups of words of the source and the target words instead of single words. This solution has been tested in the literature in different ways: multi-word joint probability models are described in [MW02]; the so-called alignment templates approach is defined in [OTN99]; Finally, the estimation and application of the so-called Phrase-based models is discussed in [TC01, MW02, ZON02, Tom03]. Phrase-based models are the standard translation models used in regular SMT systems.

A key concept used by these models is the concept of *phrase*. Specifically, a phrase is a set of one or more consecutive words of the source or the target sentences. For example, given the source sentence $f_1^4 \equiv f_1 f_2 f_3 f_4$ composed of four words, the following are examples of valid phrases of f_1^4: f_1, $f_1^2 \equiv f_1 f_2$, $f_2^3 \equiv f_2 f_3$, $f_1^4 \equiv f_1 f_2 f_3 f_4$, etc. We will use the symbols $\tilde{f}$ and $\tilde{e}$, to denote an unspecified source or target phrase, respectively. It should be noted that in this context, phrases are not linguistically motivated.

Joint Probability Models

Joint probability alignment models were proposed in [MW02]. Joint probability alignment models assume that lexical correspondences can be established at phrase-level (the concept of phrase has been explained above). This assumption is the basis of a model which is able to capture sets of equivalent phrase-pairs.

The joint probability model does not assume that the target sentences are generated from the source sentences. Instead, it is assumed that the source and the target sentences are generated simultaneously. This allows to estimate a joint probability model. Once the joint probability model has been estimated, it can be easily converted into a conditional probability model.

The model is based on the so-called *bag of concepts* C, where each concept $c_i \in C$ determines a phrase pair $(\tilde{f}_i, \tilde{e}_i)$ of the source and the target sentences. This bag of concepts defines a phrase partition of the source and the target sentences. Only those bags of concepts that allow to obtain both the source and the target sentences after applying the corresponding reordering operations will be considered. Since a bag of concepts C may or may not be successfully used to generate the sentence pair (f_1^J, e_1^I), the predicate $L(f_1^J, e_1^I, C)$ is defined to test this property. Once the bag of concepts has been generated, every concept c_i contained in the bag of concepts is analysed, determining its associated phrase pair $(\tilde{f}_i, \tilde{e}_i)$ which is generated according to the probability distribution $p(\tilde{f}_i, \tilde{e}_i)$ ($\tilde{f}_i$ and $\tilde{e}_i$ have at least one word). Finally, the phrase pairs are reordered in order to obtain both the source and the target sentences. Under this model, the probability of a given sentence pair (f_1^J, e_1^I) is obtained by summing over all possible bags of concepts $C \in \mathcal{C}$:

$$p(f_1^J, e_1^I) = \sum_{C \in \mathcal{C} | L(f_1^J, e_1^I, C)} \prod_{c_i \in C} p(\tilde{f}_i, \tilde{e}_i) \tag{1.20}$$

The joint probability model that we have described above presents a major drawback due to its inability to impose constraints on the reorderings. To solve this problem, an absolute-position distortion model is included [MW02].

The estimation of the model parameters is carried out by means of the EM algorithm. To simplify the estimation process, certain heuristic prunings are introduced. The details of the estimation process can be found in [MW02].

Alignment Templates

The key concept of this approach is the concept of alignment template [OTN99, Och02]. An alignment template is a phrase pair of the categorised source and target sentences plus an alignment between the words contained in this phrase pair.

Alignment templates model decomposes translation probability by the introduction of two hidden variables: the alignment templates sequence, $\tilde{a}_1^K$, and the word alignments z_1^K between the templates.

$$Pr(f_1^J | e_1^I) = \sum_{z_1^K, \tilde{a}_1^K} Pr(\tilde{a}_1^K | e_1^I) \cdot Pr(z_1^K | \tilde{a}_1^K, e_1^I) \cdot Pr(f_1^J | z_1^K, \tilde{a}_1^K, e_1^I) \tag{1.21}$$

It should be noted that the vector $\tilde{a}_1^K$ determines a partition of the source and the target sentences into K phrases. This allows us to define the phrase vectors $\tilde{f}_1^K \equiv f_1^J$ and $\tilde{e}_1^K \equiv e_1^I$. Taking this into account, the probability distribution $Pr(f_1^J | e_1^I)$ can be approximated as follows:

$$p(f_1^J | e_1^I) = \sum_{z_1^K, \tilde{a}_1^K} \prod_{k=1}^{K} p(\tilde{a}_k | \tilde{a}_{k-1}) \cdot p(z_k | \tilde{e}_k) \cdot p(\tilde{f}_k | z_k, \tilde{e}_k) \tag{1.22}$$

Thus, we have three different statistical models that are to be estimated:

- Phrase alignment model $p(\tilde{a}_k | \tilde{a}_{k-1})$
- Alignment template model $p(z_k | \tilde{e}_k)$
- Statistical dictionary of phrase pairs $p(\tilde{f}_k | z_k, \tilde{e}_k)$

The estimation of alignment templates models has the following steps: first, single-word alignment matrices for the sentence pairs contained in the training corpora are generated; second, bilingual word classes are trained and third, a set of phrase pairs that are consistent with the previously obtained word alignment matrices is collected. The exact details of alignment templates model estimation can be found in [Och02].

Phrase-Based Models

Phrase-based models constitute another alternative to overcome the limitations that the single-word models present. Phrase-based models also use statistical dictionaries of phrase pairs, as well as the alignment templates model.

The translation of the source sentence f_1^J into its equivalent target sentence e_1^I using phrase-based models can be explained under a generative point of view as follows:

1. The source sentence is divided into K phrases $f_1^J \equiv \tilde{f}_1^K$
2. We choose the target phrase translations for each source phrase
3. The target phrase translations are reordered to compose the target sentence $e_1^I \equiv \tilde{e}_1^K$

Similarly to IBM translation models (see section 1.4.2), phrase-based models assume that the relationships between the source and the target phrases are explained by means of a hidden alignment variable $\tilde{a}_1^K \equiv \tilde{a}_1\tilde{a}_2...\tilde{a}_K$. This hidden alignment variable summarises all the decisions made during the generative process.

According to the generative process explained above, the translation using phrase-based models implies the generation of a complete *bisegmentation* of the source and the target sentences. A bilingual segmentation or bisegmentation of length K of a sentence pair (f_1^J, e_1^I), $\tilde{A}(f_1^J, e_1^I)$, is defined as a triple $(\tilde{f}_1^K, \tilde{e}_1^K, \tilde{a}_1^K)$, where the hidden variable $\tilde{a}_1^K$ can be seen as a specific one-to-one mapping between the K segments/phrases of both sentences $(1 \leq K \leq \min(I, J))$. A bisegmentation can be seen as a phrase-level alignment or *phrase-based alignment* between two sentence pairs.

The hidden variable $\tilde{a}_1^K$ allows us to reexpress the probability distribution $Pr(f_1^J|e_1^I)$ without loss of generality as follows:

$$Pr(f_1^J|e_1^I) = \sum_{\tilde{a}_1^K} Pr(\tilde{a}_1^K, \tilde{f}_1^K|\tilde{e}_1^K) = \sum_{\tilde{a}_1^K} Pr(\tilde{a}_1^K|\tilde{e}_1^K) \cdot Pr(\tilde{f}_1^K|\tilde{a}_1^K, \tilde{e}_1^K) \tag{1.23}$$

Different assumptions can be made to model the previous probability distributions. Monotonic alignments and uniform segmentation probabilities are assumed in [ZON02], obtaining the following expression:

$$p(f_1^J|e_1^I) = \alpha(e_1^I) \sum_{\tilde{a}_1^K} p(\tilde{f}_1^K|\tilde{e}_1^K) \tag{1.24}$$

where:

$$p(\tilde{f}_1^K|\tilde{e}_1^K) = \prod_{k=1}^{K} p(\tilde{f}_k|\tilde{e}_k) \tag{1.25}$$

Jesús Tomás [Tom03] does not assume monotonic alignments:

$$p(f_1^J|e_1^I) = \sum_{\tilde{a}_1^K} \prod_{k=1}^{K} p(\tilde{a}_k|\tilde{a}_1^{k-1}) \cdot p(\tilde{f}_k|\tilde{e}_{\tilde{a}_k}) \tag{1.26}$$

A simple distortion model is proposed in [KOM03], this distortion model replaces $p(\tilde{a}_k|\tilde{a}_1^{k-1})$ with $d(a_k - b_{k-1})$, where a_k notes the starting position of the source phrase which was translated by the k-th target phrase and b_{k-1} notes the ending position of the source phrase which was translated by the $(k-1)$-th target phrase. The distortion model can be estimated by means of a joint probability model [MW02] or implemented using the formula: $d(a_k - b_{k-1}) = \alpha^{|a_k - b_{k-1} - 1|}$.

The search problem using phrase-based models is formalised using the maximum-approximation, where the segmentation length K should also be maximised:

$$\begin{aligned} \hat{e}_1^{\hat{I}} &= \arg\max_{I,e_1^I} \left\{ Pr(e_1^I) \cdot \sum_{\tilde{a}_1^K} Pr(\tilde{a}_1^K, \tilde{f}_1^K | \tilde{e}_1^K) \right\} \\ &\approx \arg\max_{I,e_1^I} \{ Pr(e_1^I) \cdot \max_{K,\tilde{a}_1^K} Pr(\tilde{a}_1^K, \tilde{f}_1^K | \tilde{e}_1^K) \} \qquad (1.27) \end{aligned}$$

where it should be noted that $e_1^I \equiv \tilde{e}_1^K$.

The main disadvantages of these models are their poor generalisation capability and the high space complexity of the phrase translation tables [Tom03]. Different solutions to deal with the problem of the space complexity have been proposed in the literature [CBBS05, ZV05, OMGVC08].

Regarding the estimation of phrase-based model parameters, different proposals can be found in the literature. The most commonly used phrase-based model estimation technique is based on the relative frequencies of the phrase pairs that are extracted from word alignment matrices [Och02], the details of this estimation technique are similar to that of the Alignment Templates model described in section 1.4.3. This estimation technique has been implemented in the publicly-available open-source THOT [OGVC05] toolkit. Additionally, different techniques that try to reduce the heuristic component that the standard estimation technique present have been defined in [TC01, OGVC05, BCBOK06, AFJC07]. In addition to this, there are proposals that try to combine phrase-based models with linguistic information, such as the factored models described in [KH07].

1.4.4 Log-Linear Models

In the early days of SMT, the translation process was formalised as a maximisation of a function with two terms, namely, the statistical language model and the statistical translation model (this is the fundamental equation of statistical machine translation, see section 1.3). The use of a language and a translation model is beneficial because our estimates for each model are errorful. By applying them together we hope to counterbalance their errors. More recently, alternative formalisations have been proposed. Such formalisations are based on the direct modelling of the posterior probability $Pr(e_1^I|f_1^J)$, replacing the generative models by *discriminative models*. Discriminative models are a class of models used in machine learning for modelling the dependence of an unobserved variable on an observed variable and they differ from the generative models in that the former ones do not allow to randomly generate samples from the joint distribution of the unobserved and observed variables. The so-called log-linear models [PRW98, ON02] constitute an example of discriminative models:

$$Pr(e_1^I|f_1^J) = \frac{exp\left(\sum_{m=1}^{M} \lambda_m h_m(f_1^J, e_1^I)\right)}{\sum_{e'^{I'}_1} exp\left(\sum_{m=1}^{M} \lambda_m h_m(f_1^J, e'^{I'}_1)\right)} \tag{1.28}$$

Log-linear models use a set of feature functions $h_m(f_1^J, e_1^I)$ each one with its corresponding weight λ_m. In the previous equation, the denominator depends only on the source sentence f_1^J, so it can be omitted during the search process. As a result of the previous considerations, we arrive at a new expression which consists in a log-linear combination of individual models $h(\cdot,\cdot)$:

$$\hat{e}_1^{\hat{I}} = \arg\max_{I, e_1^I} \left\{ \sum_{m=1}^{M} \lambda_m h_m(f_1^J, e_1^I) \right\} \tag{1.29}$$

The direct optimisation of the posterior probability in the Bayes decision rule is referred to as *discriminative training* [Ney95]. Since the features of regular SMT log-linear models are usually implemented by means of generative models, discriminative training is applied here only to estimate the weights involved in the log-linear combination. Given the training set $\mathcal{X} = \{(\mathbf{f}_1, \mathbf{e}_1), (\mathbf{f}_2, \mathbf{e}_2), ..., (\mathbf{f}_N, \mathbf{e}_N)\}$, and following the ML criterion:

$$\hat{\lambda}_1^M = \arg\max_{\lambda_1^M} \left\{ \prod_{n=1}^{N} p(\mathbf{e}_n|\mathbf{f}_n; \lambda_1^M) \right\} \tag{1.30}$$

To solve the maximisation problem shown in Equation (1.30) the so-called generalised iterative scaling algorithm is used (see section 1.5.2 for more details). Alternatively, the ML criterion can be replaced by a criterion based on automatic evaluation methods. In this case, we assume that the best model is the one that produces the smallest overall error with respect to a given error function. This new optimisation problem can be solved by means of the minimum error rate training algorithm (see section 1.5.3 for more details). The so-called Moses toolkit [KHB+07] (as well as its predecessor, the Pharaoh decoder [KOM03]) implements a SMT system based on log-linear models. The toolkit provides the functionality of training the log-linear combination weights by means of the minimum error rate training algorithm and phrase-based model estimation using standard estimation techniques.

Discriminative modelling is useful because it frees us from the generative modelling requirement that each term in our translation model must have an associated event in the translation process. Generative models are often chosen for computational reasons rather than for their accuracy. By contrast, discriminative models allow us to define a set of features that may help to improve translation. One crucial aspect in discriminative modelling is defining useful features.

1.5 Parameter Estimation Techniques

Once the statistical models have been completely defined, the next step is to estimate the set of parameters of these statistical models. This is the so-called training problem (see section 1.3). In previous sections we have mentioned three well-known parameter estimation techniques, namely, the expectation-maximisation algorithm, the generalised iterative scaling algorithm and the minimum error rate training algorithm. In the following sections we will briefly describe these three estimation techniques.

1.5.1 The Expectation-Maximisation Algorithm

The expectation-maximisation (EM) algorithm [DLR77, Wu83] is used for finding ML estimates of parameters in statistical models (please refer to section 1.3 for an explanation of the ML criterion), where the model depends on unobserved hidden variables.

The EM algorithm has two different applications. The first occurs when the data has missing values, due to problems with or limitations of the observation process. The second occurs when optimising the log-likelihood function is analytically intractable but when the log-likelihood function can be simplified by assuming the existence of additional but missing (or hidden) parameters.

We assume that data $\mathcal{X}$ is observed and generated by some distribution governed by the the set of parameters Θ. In addition to this, we assume that a *complete data* set exists $\mathcal{Z} = (\mathcal{X}, \mathcal{Y})$. The complete data comprises the incomplete data and also a *missing* or *hidden data* set $\mathcal{Y}$. Finally we assume a probability density function for the complete data:

$$p(\mathbf{z}|\Theta) = p(\mathbf{x}, \mathbf{y}|\Theta) = p(\mathbf{y}|\mathbf{x}, \Theta)p(\mathbf{x}|\Theta) \tag{1.31}$$

We use the previous equation to define a new expression of the log-likelihood, $\mathcal{L}(\Theta, \mathbf{z}) = \mathcal{L}(\Theta, \mathbf{x}, \mathbf{y})$. This is the so-called *complete data log-likelihood function*. It is worth mentioning that this new log-likelihood is a random variable since $\mathcal{Y}$ is also randomly distributed. The original log-likelihood $\mathcal{L}(\Theta, \mathbf{x})$ is referred to as the incomplete-data likelihood function.

EM algorithm first finds the expected value of the complete data log-likelihood, $\log p(\mathbf{x}, \mathbf{y}|\Theta)$, with respect to the hidden data given the incomplete data and a previous estimation of the model parameters:

$$Q(\Theta, \Theta^{t-1}) = E[\log p(\mathbf{x}, \mathbf{y}|\Theta)|\mathbf{x}, \Theta^{(t-1)}] \tag{1.32}$$

Where $\Theta^{(t-1)}$ are the currently estimated model parameters and Θ are the new parameters that are being optimised to increase Q. The evaluation of the Q function is the so-called E step of the EM algorithm.

The M step finds the set of parameters Θ that maximises the Q function computed at the E step.

$$\Theta^{(t)} = \arg\max_{\Theta} Q(\Theta, \Theta^{(t-1)}) \tag{1.33}$$

The EM algorithm estimates the set of parameters Θ of a model iteratively, starting from some initial guess. Each iteration executes the E and the M steps. As shown in [DLR77], each EM iteration improves the log-likelihood of the incomplete data $\mathcal{L}(\Theta, \mathbf{x})$ or leaves it unchanged. Indeed for most models the algorithm will converge to a local maximum of $\mathcal{L}(\Theta, \mathbf{x})$.

The M step of the EM algorithm may be only partially implemented, with the new estimate for the parameters improving the likelihood but not necessarily maximising it. Such a partial M step always improves the likelihood as well. Dempster et al. [DLR77] refer to such variants as generalised EM (GEM) algorithms.

1.5.2 Generalised Iterative Scaling

The generalised iterative scaling (GIS) algorithm [DR72] can be used to find a ML estimate of the log-linear combination weights λ_1^M shown in Equation (1.29). The application of the GIS algorithm is very costly due to the necessity of computing the normalisation factor that is shown in Equation (1.28). Och and Ney [ON02] greatly reduce the computation costs by constraining the calculation of the normalisation factor to the set of N-best translations generated by the SMT system.

1.5.3 Minimum Error Rate Training

Log-linear model weights can also be adjusted by means of the minimum error rate training (MERT) algorithm [Och03]. The MERT algorithm uses a given translation quality measure to estimate the above mentioned log-linear weights. The MERT algorithm can be implemented by means of different optimisation algorithms. Och [Och03] proposes the use of the Powell's conjugate gradient descent method [Pow64]. Alternatively, the so-called downhill-simplex algorithm [NM65] can also be used.

1.6 Search Algorithms

Once the statistical models involved in the translation process have been estimated, the remaining step consists in defining how the target sentence is generated from the source sentence. This is the so-called search problem (see section 1.3). In the purely statistical approach to MT (see section 1.3), the search problem is expressed formally by means of the fundamental equation of machine translation (see Equation (1.2)) or, alternatively, by means of Equation (1.29) corresponding to the use of log-linear models. The search problem as presented in equations 1.2 and 1.29 was demonstrated to be an NP-complete problem [Kni99, UM06].

The vast majority of the search algorithms that have been proposed so far share the same basic idea. Specifically, the search process starts from a *null-hypothesis* (that is, a hypothesis that does not contain any words) and works by iteratively extending partial hypotheses. The extension of a partial hypothesis adds new words to this hypothesis. In typical search algorithms, the translations are built from left to right. This iterative process is repeated until a *complete* hypothesis has been generated. The hypothesis extension procedure is driven by the statistical models involved in the translation process. Each partial hypothesis has an associated score. The score associated to a hypothesis gives a measure of how good this hypothesis is as a partial translation of the source sentence.

Different search algorithms have been proposed in the literature. These search algorithms can be classified into four groups: the branch-and-bound search algorithms, the dynamic programming based algorithms, the search algorithms based on greedy techniques and the search algorithms based on linear programming.

Branch-and-bound algorithms for SMT include $A^\star$ search [OUN01] and the so-called stack-decoding algorithms [BBD+96, WW97, GJK+01, OUN01]. $A^\star$ search and stack decoding algorithms use a stack data structure to incrementally extend partial hypotheses. The stack orders the hypothesis by ascending order of the score assigned to the hypotheses by the statistical models involved in the translation process. $A^\star$ search algorithms are optimal, since the search problem has been demonstrated to be NP-complete, we cannot expect to obtain an efficient search. Branch-and-bound algorithms typically follow a best-first search strategy. A depth-first search strategy has also been adopted in [BBD+96].

The DP-based algorithms [Til01, ZON02, GV03] decompose the problem of translating the source sentence into a set of sub-problems that are solved separately. The final solution is computed as a combination of the sub-problems. This procedure is based on the Bellman optimality principle [Bel57]. The DP-based algorithms for MT use a breadth-first search strategy.

The greedy algorithms for MT were described in [BBD+94, Wan98, GJK+01]. These algorithms differ from the previous ones in that they do not work by incrementally extending an initial null-hypothesis. By contrast, the greedy algorithm first heuristically generates a complete hypothesis that is iteratively improved by the application of different operators. The greedy algorithms are not commonly used in MT due to the quality of the results that they obtain, which are worse than those obtained by other search techniques. The main and only advantage of the greedy decoding algorithms is their low time complexity.

Finally, a different search strategy based on linear programming, and more specifically in integer programming has been described in [GJK+01]. This search strategy obtains high quality results but has a high computational complexity.

1.7 Alternative Techniques for Corpus-Based MT

The corpus-based techniques mentioned in section 1.2.3 not only includes the purely statistical approach described above, but also a number of alternative approaches. The most commonly used of these alternative approaches are the finite state approach and the context free grammar approach. These approaches can also be classified into the statistical approach to MT.

1.7.1 MT based on Stochastic Finite State Transducers

One alternative corpus-based approach to MT is based on the use of stochastic finite state transducers (SFSTs) for MT [VP92, CGV94, LJS+95, Als96b, Als96a, AX97, Vid97, KAO98, ABC+00].

SFSTs can be trained from bilingual corpora, obtaining a joint probability model. An SFST is a finite-state automaton which accepts sentences given in the source language and returns sentences given in the target language.

A particular kind of SFST is the so-called subsequential transducer [OGV93]. Subsequential transducers are deterministic SFST's. The main advantage of the subsequential transducers consists in their capability to *delay* their output until a sufficient number of source symbols has been seen. This is done to ensure the correctness of the output. The search problem when using SFSTs is commonly solved by means of the well-known Viterbi algorithm [Vit67].

The OSTIA [OGV93] (onward subsequential transducer inference algorithm) and the OMEGA [Vil00] (from spanish: OSTIA mejorado empleando garantías y alineamientos) algorithms allow to automatically generate SFST's. The OSTIA algorithm exclusively uses finite-state automaton techniques and the OMEGA algorithm combines finite-state automaton techniques with additional information extracted by means of statistical methods. Finally, an additional technique that allows SFSTs inference is the so-called GIATI technique [CV04] (grammatical inference and alignments for transducer inference). The GIATI technique has been implemented in the publicly-available GREAT [GSC08] toolkit.

1.7.2 MT based on synchronous context-free grammars

Context-free grammars (CFG) applied to MT confers two advantages with respect to the MT techniques described in previous sections. First, they are closely tied to some linguistic representations of syntax. Second, in their synchronous form (synchronous CFG, or SCFG), they can easily represent long-distance reordering without the exponential complexity of permutation. However, these advantages comes with new modelling challenges that are to be solved. Because of this, the context free grammar approach to MT is currently an area of active research.

Different approaches to SMT can be expressed in the SCFG formalism. One important advantage of this is that the search problem with SCFG models is equivalent to CFG parsing [Mel04]. In the following sections we will briefly describe three applications of SCFGs that are representative of their use in SMT, namely bracketing grammars, syntax-based translation and hierarchical phrase-based translation. For a more detailed review of the literature on SCFGs applied to MT, please refer to [Lop08].

Bracketing Grammars

One reason to use SCFGs is efficient expression of reordering. In the previously described techniques, long-distance reordering is difficult to model. The most permissive approach (arbitrary permutation) is exponential in sentence length. By contrast, SCFGs can represent long-distance reordering while remaining polynomial in sentence length. This motivates the use of bracketing grammars. They represent all possible reorderings consistent with a binary bracketing of the input string [Wu96].

Stochastic inversion transduction grammars (SITGs) are described in [Wu97]. A recursive statistical translation model with some similarities with the SITGs is proposed in [VV05]. A lexicalised bracketing grammar, in which non-terminal symbols are annotated with words is described in [ZG05]. A related formalism is the head transduction grammar [ABD00]. Additionally, Xiong et al. [XLL06] adapted bracketing grammars to phrase translation.

Syntax-Based Translation

Syntax-based translation [WW98, YK01] tries to capture syntactic information from both the source and the target languages. Typically, a translation system using syntax-based models works as follows: first, the system is given a source sentence containing syntactic labels and hierarchically represented by means of a tree data structure. From this tree, a set of node reordering operations is applied obtaining a new labelled tree. The structure of the resulting tree is equivalent to that of the source sentence, but it has been built according to the syntax rules of the target language. Finally, the source words contained in the tree are replaced by their corresponding translations in the target language.

Additional works on syntax-based translation can be found in the literature [GHKM04, KG05, GGK$^+$06]. A slightly different proposal of syntax-based translation system can be found in [Ima02], where a method to hierarchically extract equivalent phrases pairs from an aligned bilingual corpus is described. According to this approach, two phrases are equivalent if they can be directly translated by means of an EBMT system.

Hierarchical Phrase-Based Translation

SCFG models, since they enable only word-to-word translation, are not able to capture context information. As it was explained above, one way to alleviate this problem is to use multi-word translation models such as the phrase-based models. Ideally, we would like to benefit from the insights behind both hierarchical models and phrase-based models. This is accomplished in hierarchical phrase-based translation [Chi05, Chi07].

1.8 Interactive Machine Translation

Current MT systems are not able to produce ready-to-use texts [NIS06, CBFK$^+$07]. Indeed, MT systems usually require human post-editing in order to achieve high-quality translations.

One way of taking advantage of MT systems is to interactively combine them with the knowledge of a human translator, constituting the Interactive Machine Translation (IMT) paradigm. This IMT paradigm can be considered a special type of the so-called computer-assisted translation (CAT) paradigm [IC97].

An important contribution to IMT technology was carried out within the TransType (TT) project [FIP97, LFL00, LLL02, FLL02, Fos02]. This project entailed a focus shift in which interaction is directly aimed at the production of the target text, rather than at the disambiguation of the source text, as in former interactive systems. The idea proposed in that work was to embed data driven MT techniques within the interactive translation environment.

Following these TT ideas, Barrachina et al. [BBC$^+$09] proposed a new approach to IMT. In this approach, fully-fledged SMT systems are used to produce full target sentence hypotheses, or portions thereof, which can be partially or completely accepted and amended by a human translator. Each partially corrected text segment, or prefix, is then used by the SMT system as additional information to achieve improved suggestions. Figure 1.3 illustrates a typical IMT session. In interaction-0, the system suggests a translation ($\mathbf{e}_s$). In interaction-1, the user moves the mouse to accept the prefix composed of the first eight characters "To view " ($\mathbf{e}_p$) and presses the $\boxed{\mathbf{a}}$ key (k), then the system suggests completing the sentence with "*list of resources*" (a new $\mathbf{e}_s$). Interactions 2 and 3 are similar. In the final interaction, the user completely accepts the current suggestion.

Figure 1.4 (inspired from [VRCGV07]) shows a schematic view of these ideas. Here, f_1^J is the input sentence and e_1^I is the output derived by the IMT system from f_1^J. By observing

Figure 1.3: IMT session to translate a Spanish sentence into English.

source(f_1^J): Para ver la lista de recursos
reference($\hat{e}_1^{\hat{I}}$): To view a listing of resources

interaction-0	$\mathbf{e}_p$						
	$\mathbf{e}_s$	*To*	*view*	*the*	*resources*	*list*	
interaction-1	$\mathbf{e}_p$	To	view				
	k			$\boxed{\textbf{a}}$			
	$\mathbf{e}_s$				*list*	*of*	*resources*
interaction-2	$\mathbf{e}_p$	To	view	a	list		
	k				$\boxed{\textbf{i}}$		
	$\mathbf{e}_s$				*ng*	*resources*	
interaction-3	$\mathbf{e}_p$	To	view	a	listing		
	k					$\boxed{\textbf{o}}$	
	$\mathbf{e}_s$					*f*	*resources*
acceptance	$\mathbf{e}_p$	To	view	a	listing	of	resources

f_1^J and e_1^I, the user interacts with the IMT system, validating prefixes and/or pressing keys (k) corresponding to the next correct character, until the desired output $\hat{e}_1^{\hat{I}}$ is produced. The models used by the IMT system are obtained through a classical batch training process from a previously given training sequence of pairs $(\mathbf{f}_n, \mathbf{e}_n)$ from the task being considered.

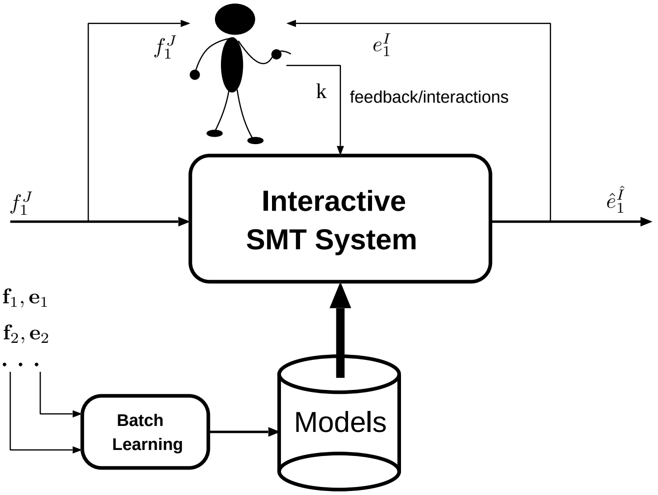


Figure 1.4: An Interactive SMT system.

In the IMT scenario we have to find an extension $\mathbf{e}_s$ for a given prefix $\mathbf{e}_p$:

$$\hat{\mathbf{e}}_s = \arg\max_{\mathbf{e}_s} \left\{ p(\mathbf{e}_s \mid f_1^J, \mathbf{e}_p) \right\} \tag{1.34}$$

Applying the Bayes rule, we arrive at the following expression:

$$\hat{\mathbf{e}}_s = \arg\max_{\mathbf{e}_s} \left\{ p(\mathbf{e}_s \,|\, \mathbf{e}_p) \cdot p(f_1^J \,|\, \mathbf{e}_p, \mathbf{e}_s) \right\} \tag{1.35}$$

where the term $p(\mathbf{e}_p)$ has been dropped since it does not depend on $\mathbf{e}_s$.

Thus, the search is restricted to those sentences e_1^I which contain $\mathbf{e}_p$ as prefix. It is also worth mentioning that the similarities between Equation (1.35) and Equation (1.2) (note that $\mathbf{e}_p\mathbf{e}_s \equiv e_1^I$) allow us to use the same models if the search procedures are adequately modified [BHV+05, BBC+09].

It should be noted that the statistical models are defined at word level while the IMT interface described in Figure 1.3 works at character level. This is not an important issue since the transformations that are required in the statistical models for their use at character level are trivial.

1.9 Evaluation

In this section we will review the automatic evaluation measures that are commonly used in three different tasks: MT, word-alignment generation and IMT. Among all the automatic measures that have been described, we will pay special attention to those that will be used to evaluate the techniques proposed in this thesis.

1.9.1 MT evaluation

In recent years, various methods have been proposed to automatically evaluate machine translation quality by comparing hypothesis translations with reference translations. Examples of such methods are word error rate (WER), position-independent word error rate (PER) [ABC+00, CNO+04], generation string accuracy [BRW00], multi-reference word error rate [NOL00], BLEU score [PRWZ01], NIST score [Dod02], METEO [BL05] and TER [SDS+06]. All these criteria try to approximate human assessment and some works report a high degree of correlation to human evaluation [PRWZ01, Dod02]. By contrast, other authors such as [CBFK+07] report substantial differences between human and automatic evaluations. Because of this, automatic MT evaluation still remains an open problem.

In this thesis, the BLEU score will be used to measure the translation quality. The BLEU (bilingual evaluation understudy) score computes the geometric mean of the precision of n-grams of various lengths between a hypothesis and a set of reference translations multiplied by a factor BP$(\cdot)$ that penalises short sentences:

$$\text{BLEU} = \text{BP}(\cdot) \exp\left(\sum_{n=1}^{N} \frac{\log p_n}{N} \right)$$

Here p_n denotes the precision of n-grams in the hypothesis translation. Typically, a value of $N = 4$ is used.

1.9.2 Word Alignment Evaluation

The generation of word and phrase alignments is strongly related to the SMT framework, since statistical alignment models constitute one key component of regular SMT systems. The performance of statistical alignment models can be measured by means of automatic methods, such as precision (P), recall (R), F-measure (F) and alignment error rate (AER) (see [ON00]). In a word alignment task we are given a parallel text and a reference alignment G which is compared with the system alignment A. Both A and G are sets whose elements are alignments between the words of the parallel text. Both A and G can be split into two subsets A_S,A_P and G_S, G_P, respectively representing *Sure* and *Probable* alignments. Precision, recall, F-measure and alignment error rate are computed as follows:

$$\begin{aligned}
P_T &= \frac{|A_T \cap G_T|}{|A_T|} \\
R_T &= \frac{|A_T \cap G_T|}{|G_T|} \\
F_T &= \frac{2P_T \cdot R_T}{|P_T + R_T|} \\
\text{AER} &= 1 - \frac{|A_S \cap G_S| + |A_P \cap G_P|}{|A_P| + |G_S|}
\end{aligned}$$

where T is the alignment type, and can be set to either S or P. In this thesis all the previously described alignment quality measures will be used.

1.9.3 IMT Evaluation

The IMT framework has its own evaluation measures since in this case, the main goal of automatic assessment is to estimate the effort of the human translator. For this purpose the following measures have been proposed: key-stroke ratio (KSR), mouse-action ratio (MAR), key-stroke and mouse-action ratio (KSMR) (these three measures are described in [BBC$^+$09]) and word-stroke ratio (WSR) [TC06].

The above mentioned IMT evaluation measures are based on the concept of *longest common character prefix* (LCP). The LCP is obtained by comparing the translation given by the IMT system with the reference sentence that the user has in mind. Specifically, the first non-matching character of the system translation marks the end of the LCP. The user moves the mouse pointer to this first non-matching character and then replaces it with the corresponding reference character. After this, a new system hypothesis is produced. This process is iterated until a full match with the reference is obtained. Each computation of the LCP would correspond to the user looking for the next error and moving the pointer to the corresponding position of the translation hypothesis, increasing the number of mouse-actions. Each character replacement, on the other hand, would correspond to a key-stroke of the user. If the first non-matching character is the first character of the new system hypothesis in a given iteration, no LCP computation is needed; that is, no mouse-actions are made by the user.

In this thesis, the following three IMT evaluation techniques will be used:

- **Key-stroke ratio** (KSR): Number of key-strokes divided by the total number of reference characters.

- **Mouse-action ratio** (MAR): Number of pointer movements plus one more count per sentence (aimed at simulating the user action needed to accept the final translation), divided by the total number of reference characters.

- **Key-stroke and mouse-action ratio** (KSMR): Number of key-strokes plus number of mouse-actions divided by the total number of reference characters.

When we evaluate an IMT system, it is also important to estimate the human effort reduction that can be obtained using this IMT system with respect to using a conventional SMT system followed by human post-editing. For this purpose, the KSR measure defined above can be compared with the *character error rate* (CER) measure. The CER measure is defined as the minimum number of characters that are to be inserted, deleted or substituted to transform the sentences generated by the translation system into the reference translations. However, the CER measure constitutes a rough estimation of the post-editing effort, since professional translators typically use text editors with autocompletion capabilities to generate the target translations. To solve this problem, we can use the *post-editing key stroke ratio* (PKSR) measure defined in [RTV10]. This measure has been applied in the field of computer-assisted transcription of text images, but can also be used here without any modification. The PKSR measure is calculated as the number of keystrokes that the user of the post-editing system must enter to achieve the reference translation, divided by the total number of reference characters. When the user enters a character to correct some incorrect word, the post-editing system automatically completes such word with the most probable word contained in the task vocabulary.

1.10 Corpus

This section describes the different parallel corpora that will be used to test the techniques proposed in this thesis. Available corpora is usually divided in two parts: the training and the test parts. The training part is used to train statistical models and the test part to obtain quality measures like those defined in section 1.9. Additionally, some corpora also include a development part which is used to adjust specific parameters of the statistical models such as the log-linear combination weights described in section 1.4.4.

1.10.1 EuTrans-I Corpus

The EuTrans-I task [Vid97, ABC+00] comes from a limited-domain Spanish-English machine translation application for human-to-human communication situations in the front-desk of a hotel. It was semi-automatically built from a small dataset of sentence pairs collected from traveller-oriented booklets.

Table 1.1 shows the main figures of the EuTrans-I corpus. As can be seen, the EuTrans-I corpus is a very simple corpus with small vocabularies and very low perplexities. By this reason, the EuTrans-I corpus is no longer used in the SMT field. However, it is described here because it will be used to test certain techniques proposed in this thesis.

Table 1.1: EuTrans-I corpus statistics.

		Spanish	English
Training	Sentences	10 000	
	Running words	97 131	99 292
	Vocabulary	686	513
Test	Sentences	2 996	
	Running words	35 023	35 590
	Perplexity (trigrams)	4.9	3.6

Table 1.2: Europarl corpus statistics for three different language pairs.

		Spanish	English	French	English	German	English
Training	Sentences	730 740		688 031		751 088	
	Running words	15.6M	15.2M	13.8M	15.3M	15.2M	16.0M
	Vocabulary	113 886	72 742	69 034	86 803	205 378	74 711
Dev	Sentences	2 000		2 000		2 000	
	Running words	60 276	57 945	65 029	57 945	54 247	57 945
	Perplexity (trigrams)	66.5	62.8	45.0	62.8	113.7	62.8
Test	Sentences	3 064 (2 000+1 064)		3 064 (2 000+1 064)		3 064 (2 000+1 064)	
	Running words	91 650	85 226	98 720	85 226	82 351	85 226
	Perplexity (trigrams)	91.9	103.3	61.5	103.3	177.8	103.3

1.10.2 Europarl Corpus

Europarl corpus [Koe05] is extracted from the proceedings of the European Parliament, which are written in the different languages of the European Union. Specifically, this is the version which was used in the shared task of the NAACL 2006 Workshop on Statistical Machine Translation [KM06]. Table 1.2 shows some statistics of this corpus, which includes parallel texts for three European language pairs, specifically Spanish-English, French-English and German-English. As it can be observed, the Europarl corpus contains a great number of sentences and great vocabulary sizes. These features are common to other well-known corpora described in the literature.

It is worthy of note that the test data is not only composed of sentence pairs extracted from the European Union government records, but also from the editorials of the Project Syndicate Website[c] which are published in all the four languages of the shared task. According to [KM06], this new test data differs from the Europarl data in various ways. The text type are editorials instead of speech transcripts. The domain is general politics, economics and science. However, it is also mostly political content and opinion.

In summary, the test corpus is composed of 2 000 *in-domain* sentence pairs and 1 064 *out-domain* sentence pairs. This feature makes this corpus specially attractive for testing statistical model adaptation techniques.

The NAACL 2006 version of the Europarl corpus also provides specific data to train the language models. Table 1.3 shows the main figures of the monolingual texts. There is twice

[c] http://www.project-syndicate.com

as much language modelling data, since training data for the machine translation system is filtered against sentences of length larger than 40 words.

Table 1.3: Europarl language model data.

	English	Spanish	French	German
Sentences	1 003 349	1 070 305	1 066 974	1 078 141
Running words	27.4M	29.1M	31.6M	26.5M
Vocabulary	86 698	151 111	109 597	277 197

1.10.3 Hansards Corpus

The Canadian Hansards corpus [Ger01] consists of a set of aligned texts in the French and the English languages extracted from the official records of the Canadian Parliament. A subset of this corpus was used in the HLT/NAACL 2003 workshop on “Building and Using Parallel Texts: Data Driven Machine Translation and Beyond” to carry out alignment experiments (see [MP03] for more details). The main figures of this corpus are shown in Table 1.4. The corpus has a development set consisting of 37 manually aligned sentence pairs and a development set of 447 manually aligned sentence pairs. The manual alignments assign two different confidence degrees: Sure (S) or Probable (P), allowing to obtain the automatic alignment quality measures described in section 1.9.

1.10.4 Xerox Corpus

The Xerox corpus [SdIfIV^{+}01] consists of translation of Xerox printer manuals involving three different pairs of languages: French-English, Spanish-English, and German-English. The main features of these corpora are shown in Table 1.5. Partitions into training, development, and test were performed by randomly selecting (without replacement) a specific number of development and test sentences and leaving the remaining ones for training.

The Xerox corpus has been typically used to test IMT techniques (see [BBC^{+}09]) and is also used in this thesis for the same purpose.

Table 1.4: Hansards corpus statistics.

		French	English
Training	Sentences	1 130 104	
	Running words	22.9M	19.9M
	Vocabulary	86 591	68 019
Dev	Sentences	37	
	Running words	704	661
	Perplexity (trigrams)	66.2	83.4
Test	Sentences	447	
	Running words	7 559	7 011
	Perplexity (trigrams)	52.1	71.6

Table 1.5: Xerox corpus statistics for three different language pairs.

		Spanish	English	French	English	German	English
Training	Sentences	55 761		52 844		49 376	
	Running words	657 172	571 960	573 170	542 762	440 682	506 877
	Vocabulary	29 565	25 627	27 399	24 958	37 338	24 899
Dev	Sentences	1 012		994		964	
	Running words	13 808	12 111	9 801	9 480	8 283	9 162
	Perplexity (trigrams)	34.0	46.2	74.1	96.2	124.3	68.4
Test	Sentences	1125		984		996	
	Running words	9 358	7 634	9 805	9 572	9 823	10 792
	Perplexity (trigrams)	59.6	107.0	135.4	192.6	169.2	92.8

Table 1.6: EU corpus statistics for three different language pairs.

		Spanish	English	French	English	German	English
Training	Sentences	214 473		215 216		222 644	
	Running words	5.8M	5.2M	6.5M	5.9M	6.1M	6.4M
	Vocabulary	97 444	83 738	91 307	83 746	152 696	86 185
Dev	Sentences	400		400		400	
	Running words	11 471	10 080	12 250	11 106	10 730	11 106
	Perplexity (trigrams)	46.1	59.4	34.3	42.6	60.8	41.7
Test	Sentences	800		800		800	
	Running words	22 631	19 944	23 945	21 906	20 791	21 906
	Perplexity (trigrams)	45.2	60.8	36.2	44.7	63.6	43.9

1.10.5 EU Corpus

The EU corpora was extracted from the Bulletin of the European Union, which exists in all official languages of the European Union [KG03] and is publicly available on the Internet. Table 1.6 shows the main figures of this corpus, which includes parallel texts in the language pairs Spanish-English, French-English and German-English. The main features of these corpora are shown in Table 2. The training, the development and the test sets were obtained in a similar way as with the Xerox corpus (see section 1.10.4). As well as the Xerox corpus, the EU corpus has also been typically used to carry out IMT experiments (see [SdIfIV+01, BBC+09]).

1.11 Summary

In this chapter we have introduced the field of MT, classifying the main MT systems that have been proposed so far according to the specific translation technology that these MT systems use. We have paid special attention to the statistical approach to MT since it is the approach in which this thesis is focused. Three different problems are to be solved when building SMT systems, namely, the modelling problem, the training problem and the search problem. We have described the main approaches to define the statistical models involved in the translation process, including the statistical n-gram language models, the single-word alignment models and the phrase-based alignment models, which are of special importance in this thesis. Regarding the training problems we have described the ML criterion and the well-known EM training algorithm, as well as other training algorithms, such as the GIS algorithm and the MERT algorithm. Finally, we have provided a brief overview of the search problem, describing the main search algorithms that have been applied to SMT.

The output of fully automatic MT systems is not error free. Because of this, an alternative framework in which the MT system and its user collaborate to generate correct translations was proposed. This is the so-called IMT framework. In this chapter we have provided a brief introduction to the IMT framework since this thesis also presents contributions on this topic.

Finally, we have also described the parallel corpora as well as the evaluation measures used to test the techniques proposed in this thesis.

CHAPTER 2
SCIENTIFIC AND TECHNOLOGIC GOALS

This thesis is focused on the statistical approach to MT, and more specifically on statistical phrase-based machine translation. The scientific ([SC]) and technologic ([TC]) goals of this thesis can be classified into two groups, namely, fully-automatic phrase-based SMT goals and interactive phrase-based SMT goals:

1. **Fully-automatic phrase-based SMT goals**

   - **Improved phrase-based model estimation** [SC]

     The standard phrase-based model estimation techniques have a strong heuristic component. The generative process of phrase-based models explicitly involves the bisegmentation of the source and the target sentences. Nevertheless, phrase-based models are commonly estimated without taking into account any information about bisegmentations. We propose an alternative phrase-based model estimation technique which considers phrase pairs as part of complete bisegmentations of the source and the target sentences.

   - **Phrase-based model estimation from very large corpora** [TC]

     Phrase-based models have huge memory requirements when estimated from large corpora. These high memory requirements often make the estimation unfeasible. To solve this problem we propose a specific estimation technique that allows us to transform main memory requirements into disk space requirements.

   - **Development of open-source software for phrase-based SMT** [TC]

     Open-source software constitutes a valuable resource for the research community. In this thesis we present the THOT toolkit for SMT. The THOT toolkit allows to estimate phrase-based models using two different estimation techniques, namely, the well-known, standard phrase-based model estimation technique and the improved phrase-based model estimation technique proposed in this thesis.

- **Specific phrase-based model derivation** [SC]

  Standard phrase-based models are composed of a set of translation probabilities between phrase pairs. Typically, these statistical dictionaries of phrase pairs constitute the key features of the log-linear combinations used by current SMT systems. In addition to this, some extra features are added to help improving the translation quality. These extra features usually lack of a formal justification. We give a specific phrase-based model derivation that, after making the appropriate assumptions, allows us to obtain a set of statistical submodels governing different aspects of the translation process. These submodels can also be added as individual features of a log-linear combination. The specific phrase-based model derivation proposed here is used as a key component of other proposals presented in this thesis.

- **Branch-and-bound search for phrase-based SMT** [SC]

  The branch-and-bound search paradigm constitutes one possible way to tackle the search problem in SMT. SMT search algorithms make a trade-off between efficiency and translation quality. Here we propose a branch-and-bound algorithm for phrase-based SMT. This search algorithm offers several ways to make such a trade-off by modifying its parameters.

- **Efficient decoding with large phrase-based models** [TC]

  Since phrase-based models basically consists in statistical dictionaries of phrase pairs, their estimation from very large corpora yields a huge number of parameters which are to be stored in memory. The handling of millions of model parameters has become a bottleneck in the field of SMT. We propose a way to solve this problem which is strongly inspired by a classic concept of computer architecture: cache memory. The proposed technique allows us to transform main memory requirements into disk requirements. In addition to this, we also propose a specific data structure with very low memory requirements to represent the phrase pairs that compose the phrase models.

- **Generation of phrase-based alignments** [SC]

  The problem of finding the best alignment at phrase level has not been extensively addressed in the literature. This problem is interesting since a range of different applications from phrase-based SMT systems to machine-aided NLP tools can benefit from the availability of phrase-based alignments. We propose the use of smoothed phrase-based statistical alignment models together with a specific search algorithm to compute the best phrase-to-phrase alignment for a pair of sentences.

2. **Interactive phrase-based SMT goals**

   - **Alternative IMT techniques** [SC]

     Common IMT techniques rely on a word graph data structure that represents possible translations of the given source sentence. During the IMT process for the source sentence, the system makes use of the word graph generated for that sentence in order to complete the prefixes accepted by the human translator. A common problem in IMT arises when the user sets a prefix which cannot be found in the word graph. The common procedure to face this problem is to perform a tolerant search in the word graph based on the well known concept of Levenshtein distance, allowing us to obtain the most similar string for the given prefix. This tolerant search is not included in the statistical formalisation of the IMT system. In this thesis we propose an alternative formalisation of the IMT framework in which the tolerant search is conducted by a stochastic error-correction model. This new IMT framework can also be applied to other machine-aided NLP tools. Alternatively, we propose a new IMT technique which is not based on the use of word graphs. This new IMT technique modifies the phrase-based alignment generation techniques also proposed in this thesis to obtain the suffixes required in the IMT framework.

- **Online learning for IMT** [SC]

  The vast majority of the existing work on IMT makes use of the well-known batch learning paradigm. In the batch learning paradigm, the training of the IMT system and the interactive translation process are carried out in separate stages. This paradigm is not able to take advantage of the new knowledge produced by the user of the IMT system. In this thesis, we propose the application of the online learning paradigm to the IMT framework. In the online learning paradigm, the training and prediction stages are no longer separated. This feature is particularly useful in IMT since it allows the user feedback to be taken into account.

Part II

Fully-Automatic Phrase-Based Statistical Machine Translation

CHAPTER 3

PHRASE-BASED STATISTICAL MACHINE TRANSLATION

3.1 Introduction

Since they were proposed, phrase-based models have received increasing attention from the SMT community and they currently constitute the state-of-the-art in this discipline. One of the most interesting properties of phrase-based models is their ability to capture context information in contrast with single-word alignment models (represented by the IBM and the HMM-based alignment models). Typically, very strong assumptions are made during the derivation of standard phrase-based models, reducing them to mere statistical dictionaries of phrase pairs (also called bilingual phrases). In spite of these strong assumptions, the estimation of phrase-based models is a challenging issue due to its computational complexity both in terms of time and space. In the last years, this problem has become even worse due to the availability of larger and larger corpora.

This chapter is devoted to the study of different issues regarding the modelling and the training problems in phrase-based SMT (PB-SMT): as an introduction, the standard estimation technique for phrase-based models is described in section 3.2. We propose an alternative way to carry out the estimation of phrase-based models in section 3.3. A specific phrase-based model estimation technique which is able to work with very large corpora is proposed in section 3.4. We give an alternative phrase-based model derivation in section 3.5. Finally, we provide a summary of the chapter in section 3.6.

In addition to the previously described content, the majority of the techniques proposed in this chapter have been implemented into the open-source THOT toolkit for phrase-based statistical machine translation. The THOT toolkit is described in Appendix B.

3.2 Standard Estimation of Phrase-Based Models

As mentioned above, PB-SMT systems are based on statistical dictionaries of phrase pairs, also called phrase tables, that must be previously estimated in order to perform the translation.

Different techniques have been proposed in the literature to carry out this estimation process (we will give more details on this in section 3.3). Among these techniques, there is one that has become the standard technique implemented in regular PB-SMT systems due to its efficiency and good performance. This standard estimation technique first extracts a set of bilingual phrases from a training corpus according to a predefined extraction criterion. After this extraction process, a probability distribution is estimated for the resulting bilingual phrase dictionary following standard ML estimation techniques.

According to [KOM03], there are three ways of obtaining the bilingual phrases from a parallel training corpus:

1. From word-based alignments.
2. From syntactic phrases (see [YK01] for more details).
3. From sentence alignments, by means of a joint probability model (see [MW02]).

The standard phrase-based model estimation technique uses the first method, in which the bilingual phrases are extracted from a bilingual, word-aligned training corpus. The extraction process is driven by an additional constraint: each bilingual phrase must be consistent with its corresponding word alignment matrix, A, as shown in equation (3.1) (which is the same given in [Och02] for the alignment template approach).

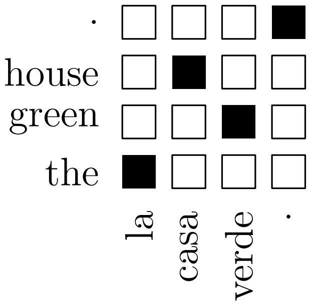
$$\begin{aligned} \mathcal{BP}(f_1^J, e_1^I, A) = \{(f_j^{j+m}, e_i^{i+n} | \forall (i', j') \in A : \\ j \leq j' \leq j+m \iff i \leq i' \leq i+n\} \end{aligned} \tag{3.1}$$

Hence, the set of consistent bilingual phrases is constituted by those bilingual phrases where all the words within the source phrase are only aligned to the words of the target phrase and viceversa. Figure 3.1 shows a word alignment matrix example along with its corresponding set of consistent bilingual phrases. The phrase pair (casa – house) is an example of a consistent phrase pair, since the word “casa” is aligned with the word “house”. By contrast, the phrase pair (casa – green) is not consistent with the alignment matrix, since the word “casa” is aligned with the word “house”, which is not included in the phrase pair.

The word alignment matrices are supposed to be manually generated by linguistic experts; however, due to the cost of such generation, in practise they are obtained using single-word alignment models. This can be done by means of the GIZA++ toolkit, which generates word alignments for the training data as a by-product of the estimation of IBM models.

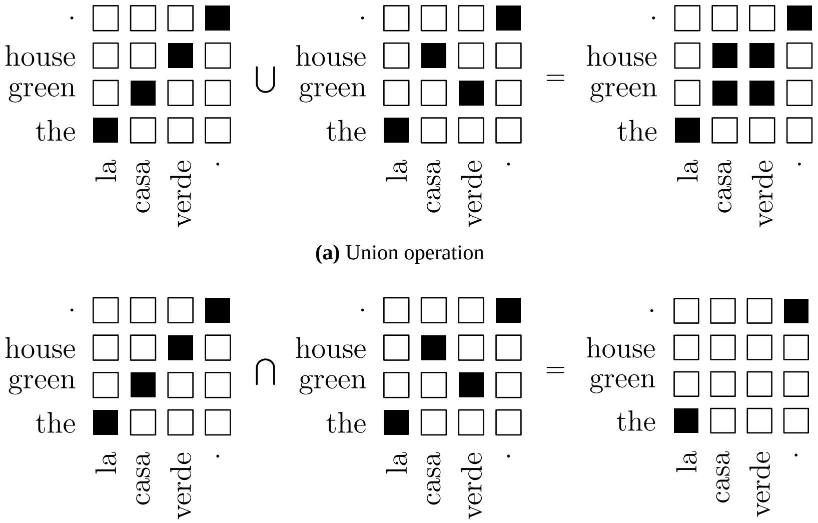
Since word alignment matrices obtained via the estimation of IBM models are restricted to being functions (each word of the source sentence can be aligned with one and only one word of the target sentence), some authors [Och02] have proposed performing operations between alignment matrices in order to obtain better alignments. The common procedure consists of estimating IBM models in both directions and performing different operations with the resulting alignment matrices such as union or intersection. Figures 3.2a and 3.2b show the result of the union operation and the intersection operation, respectively, executed on two alignment matrices.

One issue that may arise during the estimation process based on IBM-generated word-alignment matrices is the occurrence of words that are not aligned into the matrices (the so-called *spurious* and *zero fertility* words, see [BDDM93]). These special words are not



Source phrase	Target phrase
la	the
casa	house
verde	green
casa verde	green house
la casa verde	the green house
.	.
casa verde .	green house .
la casa verde .	the green house .

Figure 3.1: Set of consistent bilingual phrases (right) given a word alignment matrix (left).



(a) Union operation

(b) Intersection operation

Figure 3.2: Example of the execution of different operations between two alignment matrices.

taken into account by equation (3.1) and must be considered separately. A simple way to solve this problem consists in modifying the consistence condition given by equation (3.1) to exclude those words that are not aligned at all (see [Och02]). In addition to this, only those phrase pairs with at least one aligned word may form a consistent phrase pair. For instance, let us consider the alignment matrix that is obtained by means of the intersection operation in Figure 3.2b, where the source words “casa” and “verde”, and the target words “green” and “house” are not aligned. Under these circumstances, some extra phrase pairs would be consistent with the alignment matrix, including the following: (La casa – the), (la casa – the green), (la casa – the green), (casa verde – green), etc. By contrast, the phrase pair (casa – house) would not be in $\mathcal{BP}$, since it would not contain any aligned words.

Once the phrase pairs are collected, the phrase translation probability distribution is calculated via relative frequency (RF) estimation as follows:

$$p(\tilde{f}|\tilde{e}) = \frac{c(\tilde{f},\tilde{e})}{\sum_{\tilde{f}'} c(\tilde{f}',\tilde{e})} \tag{3.2}$$

We will refer to this standard estimation method for phrase-based models as RF estimation.

3.2.1 Implementation Details

In this section we will show the implementation details of the algorithm implemented by the standard estimation technique to extract the set of consistent bilingual phrases. This will be useful to compare the algorithmic complexity of the standard estimation technique and that of the alternative estimation technique presented in this thesis.

Algorithm 3.1 shows the `phrase_extract` algorithm. The `phrase_extract` algorithm allows to extract the set of consistent bilingual phrases given the source and the target sentences and their corresponding alignment matrix. This algorithm is identical to that proposed in [Och02]. The first two *while* loops (lines 3 and 5) iterates over the set of possible source phrases. The `quasi_consecutive` predicate is introduced here to appropriately handle unaligned words. Specifically, this predicate is evaluated to true if the aligned source positions contained in the set $\mathcal{SP}$ are consecutive, with the possible exception of words that are not aligned at all. At line 9, the source phrase $f_{j'}^{j''}$ is completely determined. After that, two inner *while* loops (lines 13 and 15) are used to iterate over the target phrases, $e_{i_1}^{i_2}$, that can be used to obtain consistent phrase pairs of the form $(f_{j'}^{j''}, e_{i_1}^{i_2})$.

The time complexity of the `phrase_extract` algorithm given the input parameters (f_1^J, e_1^I, A) is in $\mathcal{O}(I^2J^2)$ (the first two *while* loops are executed in $\mathcal{O}(I^2)$ and the two inner *while* loops have a time complexity in $\mathcal{O}(J^2)$).

3.3 Bisegmentation-based RF Estimation

The standard estimation method presented in the previous section is heuristic for two reasons. First, the bilingual phrases are obtained from a given single-word alignment matrix, which forces us to impose a heuristic consistence restriction in order to extract them. Second, as was previously explained, the translation process using phrase-based models involves the generation of a bisegmentation[a] between the source and target sentences; however, the extracted bilingual phrases are not considered as part of complete bisegmentations during the estimation of the model. The first problem cannot be solved without changing the whole estimation method. By contrast, an alternative estimation technique that tries to solve the second

[a] The concept of bisegmentation was explained in section 1.4.3, a bisegmentation is basically defined as a one to one mapping between the source and the target phrases that compose the sentences e_1^I and f_1^J.

```
input   : $e_1^I$,$f_1^J$,$A$
output  : $\mathcal{BP}$
auxiliar: $\mathcal{SP}$ (set of source positions), $\mathcal{TP}$ (set of target positions)
begin
    $i_1 := 1$
    while $i_1 \leq I$ do
        $i_2 := i_1$
        while $i_2 \leq I$ do
            $\mathcal{SP} := \{j|\exists i : i_1 \leq i \leq i_2 \wedge A(i,j)\}$
            if quasi_consecutive ($\mathcal{SP}$) then
                $j_1 := min(\mathcal{SP})$
                $j_2 := max(\mathcal{SP})$
                $\mathcal{TP} := \{i|\exists j : j_1 \leq j \leq j_2 \wedge A(i,j)\}$
                if $\mathcal{TP} \subseteq \{i_1, i_1 + 1, ..., i_2\}$ then
                    $j' := j_1$
                    while $j' = j_1 \vee (j' > 0 \wedge \forall i : A(i,j') = 0)$ do
                        $j'' := j_2$
                        while $j'' = j_2 \vee (j'' \leq J \wedge \forall i : A(i,j'') = 0)$ do
                            $\mathcal{BP} := \mathcal{BP} \cup \{(f_{j'}^{j''}, e_{i_1}^{i_2})\}$
                            $j'' := j'' + 1$
                        $j' := j' - 1$
            $i_2 := i_2 + 1$
        $i_1 := i_1 + 1$
end
```

Algorithm 3.1: Pseudocode for the `phrase_extract` algorithm.

problem while maintaining the use of single-word alignment matrices can be proposed. We will refer to this new estimation technique as *bisegmentation-based RF* (BRF) estimation.

In the following sections we describe the details of our proposed alternative estimation technique. Specifically, in section 3.3.1 some complexity issues regarding the estimation of phrase-based models are reviewed. The algorithm that implements our proposed technique is described in section 3.3.2. Some implementation details are given in section 3.3.3. Finally, some possible extensions and applications of the proposed algorithms are given in section 3.3.4.

3.3.1 Complexity Issues

The translation of a given sentence into another sentence in the target language using phrase-based models is a complex task. According to the generative process of phrase-based models, a complete bisegmentation of both the source and the target sentences has to be generated. To illustrate the complexity of this task, we will calculate the total number of bisegmentations for a given sentence pair.

Before calculating the number of bisegmentations for a sentence pair, it is illustrative to

Table 3.1: Set of all monolingual segmentations for a source sentence of 4 words and their representation as a binary number of 3 bits.

Segmentation	Code
$f_1f_2f_3f_4$	000
$f_1f_2f_3 - f_4$	001
$f_1f_2 - f_3f_4$	010
$f_1f_2 - f_3 - f_4$	011
$f_1 - f_2f_3f_4$	100
$f_1 - f_2f_3 - f_4$	101
$f_1 - f_2 - f_3f_4$	110
$f_1 - f_2 - f_3 - f_4$	111

calculate the number of monolingual segmentations for a single sentence. For this purpose, we will show that, given a sentence composed of n words, any monolingual segmentation of this sentence can be represented by a binary number of $n-1$ bits.

Let us consider that we are given a sentence in the source language composed of four words: $f_1^4 = f_1f_2f_3f_4$. The monolingual segmentation of length 2 where the first word f_1 is in the first phrase and the rest of the words $f_2f_3f_4$ are in the second phrase, $(f_1 - f_2f_3f_4)$, can be represented as a binary number of 3 bits: 100. In this binary number, the n'th bit[b] is set to 1 if the n'th word and the $(n+1)$'th word of the sentence belong to different phrases. Table 3.1 shows a complete example of the set of all possible monolingual segmentations for the sentence f_1^4 and how these segmentations are represented by means of binary numbers. As can be seen in the table, the number of monolingual segmentations is equal to the number of possible bit combinations. Thus, we conclude that the number of monolingual segmentations of a sentence composed of n words is 2^{n-1}.

Since a bisegmentation of two sentences is a one-to-one mapping between K source and target phrases, we are interested in the total number of monolingual segmentations of a given length, because only those segmentations with the same length can be combined.

If we represent the monolingual segmentations with binary numbers as was explained above, the segmentation length will be given by the number of bits that are set to 1. The number of monolingual segmentations of length K will be given by all those binary numbers with $K-1$ bits set to 1 and $J-K$ bits set to 0. This number is given by the number of permutations of 2 elements where the first one is repeated $K-1$ times and the second is repeated $J-K$ times, and we will note it as $P^{J-1}_{(K-1)(J-K)}$.

Let $\mathcal{MS}_{f_1^J, e_1^I, K}$ be the set of all possible monotonic segmentations of length K given the source sentence f_1^J and the target sentence e_1^I. The total number of monotonic bisegmentations of length K (with $1 \leq K \leq \min(I, J)$), $|\mathcal{MS}_{f_1^J, e_1^I, K}|$, is given by the following expression:

$$|\mathcal{MS}_{f_1^J, e_1^I, K}| = P^{J-1}_{(K-1)(J-K)} \cdot P^{I-1}_{(K-1)(I-K)} \tag{3.3}$$

Hence, the total number of monotonic bisegmentations for each possible value of K,

[b] Here we assume that the first bit is the leftmost bit.

$|\mathcal{MS}_{e_1^I,f_1^J}|$, is given by:

$$|\mathcal{MS}_{f_1^J,e_1^I}| = \sum_{k=1}^{\min(I,J)} |\mathcal{MS}_{f_1^J,e_1^I,k}| \tag{3.4}$$

Let $\mathcal{S}_{f_1^J,e_1^I,K}$ be the set of all possible non-monotonic bisegmentations of length K given f_1^J and e_1^I. The total number of non-monotonic bisegmentations of length K, $|\mathcal{S}_{f_1^J,e_1^I,K}|$, is given by:

$$|\mathcal{S}_{f_1^J,e_1^I,K}| = |\mathcal{MS}_{f_1^J,e_1^I,K}| \cdot K! \tag{3.5}$$

where the factorial term comes from the fact that a monolingual segmentation of length K can be permuted in $K!$ different ways.

Let $\mathcal{S}_{f_1^J,e_1^I}$ be the set of all possible non-monotonic bisegmentations. The total number of non-monotonic bisegmentations for each possible value of K, $|\mathcal{S}_{f_1^J,e_1^I}|$, is calculated in the same way as it was shown for the monotonic case:

$$|\mathcal{S}_{f_1^J,e_1^I}| = \sum_{k=1}^{\min(I,J)} |\mathcal{S}_{f_1^J,e_1^I,k}| \tag{3.6}$$

The huge number of non-monotonic bisegmentations for a sentence pair constitutes a serious challenge if phrase-based models are intended to be estimated by means of the EM algorithm, since the E-step has to compute sufficient statistics for each possible bisegmentation. According to our previous calculations, a brute force algorithm implementing the E-step would have factorial complexity.

The calculation of the expectation during the estimation of phrase-based models using the EM algorithm has long been suspected of being NP-hard. Marcu and Wong [MW02] proposed an approximated E-step based on the Viterbi alignments of the training pairs. An exponential-time dynamic program to systematically explore the set of possible bisegmentations was proposed in [DGZK06]; in practise, however, the space of alignments has to be pruned severely using word alignments to control the running time of EM. More recently, the computation of the E-step for phrase-based models has been demonstrated to be an NP-hard problem [DK08]. Finally, a proposal based on sampling is applied to estimate a Bayesian translation model in [DBCK08].

We propose an alternative estimation technique for phrase-based models which is also based on word alignments as well as the standard estimation technique described in section 3.2. In contrast to the standard estimation technique, our proposal extracts the phrase counts from the space of possible bisegmentations. The word alignments are used here to constrain this space of possible bisegmentations, making the estimation process feasible. Our proposal does not use the EM algorithm, but the techniques proposed here can be applied to greatly simplify the time requirements of the E-step. In this sense, our proposal can be seen as the predecessor of the work presented in [DGZK06], where the EM algorithm is used to estimate phrase-based models and the set of possible bisegmentations is constrained to those that can be obtained using consistent phrase pairs. However, in that work, the authors claim that even if this constraint is imposed, the problem is still intractable. To reduce the computational cost, De Nero et al. [DGZK06] introduce a maximum phrase length of three words during the training. In addition to this, different factors cause their estimation algorithm to rule out more than one half of the training set. As will be demonstrated in section 3.3.3, The proposal presented here does not have such disadvantages.

3.3.2 Algorithm

In this section we give a new proposal for model estimation. The estimation procedure has three steps that are repeated for each sentence pair and its corresponding alignment matrix (f_1^J, e_1^I, A):

1. Obtain the set $\mathcal{BP}(f_1^J, e_1^I, A)$ of all consistent bilingual phrases.
2. Obtain the set $\mathcal{S}_{\mathcal{BP}(f_1^J, e_1^I, A)}$ of all possible bilingual segmentations of the pair (f_1^J, e_1^I) that can be composed using the extracted bilingual phrases.
3. Update the counts (actually fractional counts) for every different phrase pair $(\tilde{f}, \tilde{e})$ in the set $\mathcal{S}_{\mathcal{BP}(f_1^J, e_1^I, A)}$, as:

   $$c(\tilde{f}, \tilde{e}) = c'(\tilde{f}, \tilde{e}) + \frac{c(\tilde{f}, \tilde{e} | \mathcal{S}_{\mathcal{BP}(f_1^J, e_1^I, A)})}{|\mathcal{S}_{\mathcal{BP}(f_1^J, e_1^I, A)}|}$$

   where $c'(\tilde{f}, \tilde{e})$ is the previous count of $(\tilde{f}, \tilde{e})$, $c(\tilde{f}, \tilde{e} | \mathcal{S}_{\mathcal{BP}(f_1^J, e_1^I, A)})$ is the number of times that the pair $(\tilde{f}, \tilde{e})$ occurs in $\mathcal{S}_{\mathcal{BP}(f_1^J, e_1^I, A)}$, and $| \cdot |$ denotes the cardinality of the set $\mathcal{S}_{\mathcal{BP}(f_1^J, e_1^I, A)}$.

Afterwards the probability of every phrase pair $(\tilde{f}, \tilde{e})$ is computed again by relative frequency estimation as follows:

$$p(\tilde{f} | \tilde{e}) = \frac{c(\tilde{f}, \tilde{e})}{\sum_{\tilde{f}'} c(\tilde{f}', \tilde{e})}$$

Step 2 implies that if a bilingual phrase cannot be part of any bisegmentation for a given sentence pair, this bilingual phrase will not be extracted. For this reason, BRF estimation extracts fewer bilingual phrases than the RF estimation.

Figure 3.3 shows all possible bisegmentations for the word alignment matrix given in Figure 3.1. Phrase alignments are represented here using boxes. The counts and fractional counts for each extracted bilingual phrase will differ for each estimation method, as shown in Table 3.2 for the RF and the BRF estimation methods, respectively.

3.3.3 Implementation Details

The key aspect of the BRF estimation algorithm described in the previous section is the generation of the set $\mathcal{S}_{\mathcal{BP}(f_1^J, e_1^I, A)}$ containing all those bisegmentations for a sentence pair that can be composed using consistent phrase pairs.

Algorithm 3.2 shows the pseudocode for the `brf_phr_extract` algorithm. Such algorithm calculates the phrase counts for a given sentence pair from the set of bisegmentations composed of consistent phrase pairs. The algorithm takes as input the sentence pair and

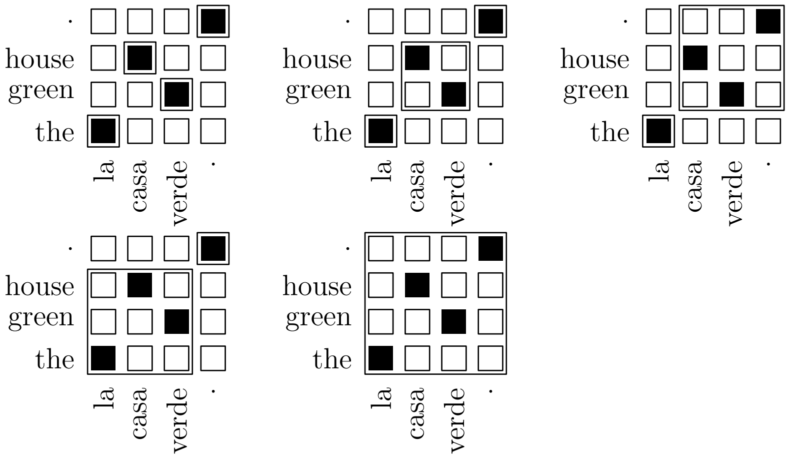


Figure 3.3: Possible bisegmentations for a given word-alignment matrix.

Table 3.2: Bilingual phrase counts and fractional counts for RF and BRF estimation, respectively, for the sentence pair shown in Figure 3.1.

$\tilde{f}$ – $\tilde{e}$	**RF**	**BRF**
la – the	1	3/5
casa – house	1	1/5
verde – green	1	1/5
casa verde – green house	1	1/5
la casa verde – the green house	1	1/5
. – .	1	3/5
casa verde . – green house .	1	1/5
la casa verde . – the green house .	1	1/5

its word alignment matrix (f_1^J, e_1^I, A), the set of consistent phrase pairs $\mathcal{BP}$ and the set of source positions that are to be aligned ($\mathcal{SP}$). The set $\mathcal{SP}$ initially contains every position of the source sentence, that is, $\mathcal{SP} = \{j|1 \leq j \leq J\}$. Similarly, the set $\mathcal{TP}$ initially contains every position of the target sentence, that is, $\mathcal{TP} = \{i|1 \leq i \leq I\}$. Given these input parameters, the algorithm recursively obtains phrase counts from the constrained set of bisegmentations $\mathcal{S}_{\mathcal{BP}(f_1^J, e_1^I, A)}$. The algorithm works by extending partial bisegmentations using consistent phrase pairs from the set $\mathcal{BP}$, only those extensions that do not cause overlapping alignments are considered (the target positions to be aligned must be contained in $\mathcal{TP}$). The recursion ends when all the source and the target sentence positions have been aligned (the sets $\mathcal{SP}$ and $\mathcal{TP}$ are empty).

It is worthy of note that Algorithm 3.2 works by exploring the tree of possible bisegmentations. Considering the example depicted in Figure 3.3, the tree containing the set of

```
input  : $f_1^J$,$e_1^I$,$A$,$\mathcal{BP}$,$\mathcal{SP}$ (set of source positions that are to be aligned),
         $\mathcal{TP}$ (set of target positions that are to be aligned)
output : $\mathcal{C}$ (set of phrase counts in $\mathcal{S}_{\mathcal{BP}(f_1^J,e_1^I,A)}$),
         S (bisegmentation size $|\mathcal{S}_{\mathcal{BP}(f_1^J,e_1^I,A)}|$)
begin
    if $\mathcal{SP} \neq \emptyset$ then
        $j_1 := min(\mathcal{SP})$
        $j_2 := max(\mathcal{SP})$
        $j := j_1$
        while $j \leq j_2$ do
            forall $(f_{j_1}^{j}, e_{i_1}^{i_2}) \in \mathcal{BP} \wedge \{i_1, i_1+1, ..., i_2\} \in \mathcal{TP}$ do
                $\mathcal{SP}' := \mathcal{SP} - \{j_1, j_1+1, ..., j\}$
                $\mathcal{TP}' := \mathcal{TP} - \{i_1, i_1+1, ..., i_2\}$
                $(\mathcal{C}', S')$ :=brf_phr_extract ($f_1^J$,$e_1^I$,$A$,$\mathcal{BP}$,$\mathcal{SP}'$,$\mathcal{TP}'$)
                $\mathcal{C} := \mathcal{C} \cup \mathcal{C}' \cup \{(f_{j_1}^{j}, e_{i_1}^{i_2}) \times S'\}$
                $S := S + S'$
            $j := j + 1$
    else
        $\mathcal{C} := \emptyset$
        if $\mathcal{TP} := \emptyset$ then
            $S := 1$
        else
            $S := 0$
end
```

Algorithm 3.2: Pseudocode for the `brf_phr_extract` algorithm.

possible bisegmentations has the form given in Figure 3.4. In the tree of possible bisegmentations, each bisegmentation is given by a path from the root of the tree to one of its leafs. The node numbers show the order in which the nodes are visited by the `brf_phr_extract` algorithm. Each edge of the tree represents a bisegmentation decision and is labelled with the corresponding phrase pair implied by this bisegmentation decision. The set of possible labels for each edge of the tree is the set $\mathcal{BP}$ of consistent phrase pairs.

The computational complexity of the `brf_phr_extract` algorithm is given by the number of nodes of the bisegmentation tree. This size may vary depending on the given word alignment matrix A. If the word alignment matrices do not contain unaligned target words, then there is at most one consistent pair $(f_{j_1}^{j_2}, e_{i_1}^{i_2}) \in \mathcal{BP}$ for a given source phrase $f_{j_1}^{j_2}$. Under these circumstances, in the best case scenario, where all the words of the source and the target sentences are mutually aligned, the complexity of the algorithm is linear with the length of the source sentence, J. By contrast, in the worst case scenario, where each source word is aligned with only one target word and the word alignments are located in the main diagonal of the alignment matrix A, the complexity is in $\mathcal{O}(2^{J-1})$. This exponential complexity comes from the fact that, in the above described conditions, each monolingual segmentation of the source sentence f_1^J is contained in $\mathcal{S}_{\mathcal{BP}(f_1^J,e_1^I,A)}$, resulting in a bisegmentation tree containing

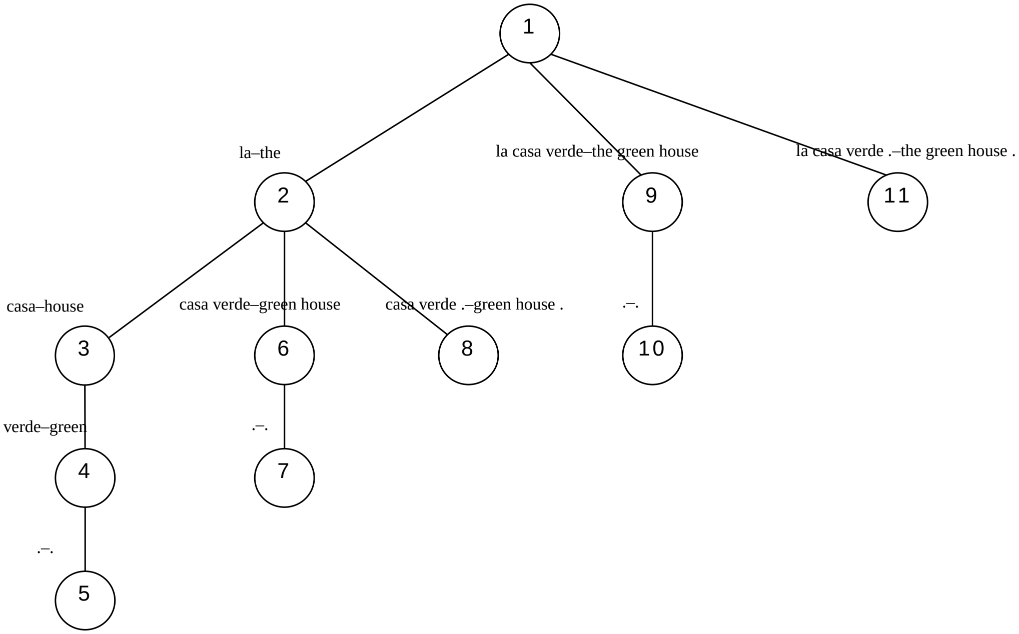

1
la–the
la casa verde–the green house
la casa verde .–the green house .
2
9
11
casa–house
casa verde–green house
casa verde .–green house .
.–.
3
6
8
10
verde–green
.–.
4
7
.–.
5

Figure 3.4: Tree of possible bisegmentations for a sentence pair.

2^{J-1} leafs (each leaf corresponds to one complete bisegmentation). There are a great number of possible situations between the best and the worst case scenarios. One example of these situations is the word alignment matrix given in Figure 3.1, which presents a non-monotonic alignment. This non-monotonic alignment reduces the number of possible bisegmentations to only five (see Figure 3.3). The existence of unaligned words in the word alignment matrices may vary the number of bisegmentations, but the upper bound of the complexity is still exponential.

Efficient Estimation Algorithm

The exponential complexity of Algorithm 3.2, although it is far from the factorial complexity of the brute force algorithm to enumerate the set of non-monotonic bisegmentations, is not acceptable when the algorithm is to be applied to long sentences. However, the time complexity can be greatly improved if certain results obtained during the exploration of the tree of possible bisegmentations are reutilised. Specifically, those partial bisegmentations with the same set of aligned source and target positions can be completed in the same way. As a result of this, we can establish a set of equivalence classes for the partial bisegmentations. For each equivalence class representing a set of partial bisegmentations, the number of possible completions, S, and the set of phrase counts, $\mathcal{C}$, that corresponds to these completions are computed. For example, in the example given in Figure 3.4, nodes 4, 6 and 9 represent three partial bisegmentations belonging to the same equivalence class.

```
input    : $f_1^J$,$e_1^I$,$A$,$\mathcal{BP}$,$\mathcal{SP}$ (set of source positions that are to be aligned),
           $\mathcal{TP}$ (set of target positions that are to be aligned)
output   : $\mathcal{C}$ (set of phrase counts in $\mathcal{S}_{\mathcal{BP}(f_1^J,e_1^I,A)}$),
           S (bisegmentation size $|\mathcal{S}_{\mathcal{BP}(f_1^J,e_1^I,A)}|$)
auxiliar : $\mathcal{EC}$ (Set of equivalence classes),
           $\mathcal{C}_{(\mathcal{SP},\mathcal{TP})}$ (set of phrase counts for the equivalence class $(\mathcal{SP},\mathcal{TP})$),
           $S_{(\mathcal{SP},\mathcal{TP})}$ (bisegmentation size for the equivalence class $(\mathcal{SP},\mathcal{TP})$)
begin
    if $\mathcal{SP} \neq \emptyset$ then
        $j_1 := min(\mathcal{SP})$
        $j_2 := max(\mathcal{SP})$
        $j := j_1$
        while $j \leq j_2$ do
            forall $(f_{j_1}^{j}, e_{i_1}^{i_2}) \in \mathcal{BP} \wedge \{i_1, i_1+1, ..., i_2\} \in \mathcal{TP}$ do
                $\mathcal{SP}' := \mathcal{SP} - \{j_1, j_1+1, ..., j\}$
                $\mathcal{TP}' := \mathcal{TP} - \{i_1, i_1+1, ..., i_2\}$
                if $(\mathcal{SP}', \mathcal{TP}') \notin \mathcal{EC}$ then
                    $(\mathcal{C}', S')$ :=brf_phr_extract ($f_1^J$,$e_1^I$,$A$,$\mathcal{BP}$,$\mathcal{SP}'$,$\mathcal{TP}'$)
                    $\mathcal{C}_{(\mathcal{SP}',\mathcal{TP}')} := \mathcal{C}'$
                    $S_{(\mathcal{SP}',\mathcal{TP}')} := S'$
                    $\mathcal{EC} := \mathcal{EC} \cup \{(\mathcal{SP}', \mathcal{TP}')\}$
                $\mathcal{C} := \mathcal{C} \cup \mathcal{C}_{(\mathcal{SP}',\mathcal{TP}')} \cup \{(f_{j_1}^{j}, e_{i_1}^{i_2}) \times S_{(\mathcal{SP}',\mathcal{TP}')}\}$
                $S := S + S_{(\mathcal{SP}',\mathcal{TP}')}$
            $j := j + 1$
    else
        $\mathcal{C} := \emptyset$
        if $\mathcal{TP} := \emptyset$ then
            $S := 1$
        else
            $S := 0$
end
```

Algorithm 3.3: Pseudocode for the `brf_phr_extract_dp` algorithm.

The previous considerations allow us to propose a recursive dynamic programming algorithm to extract phrase counts using BRF estimation. Algorithm 3.3 shows the implementation details of the new proposal. The proposed algorithm stores the number of bisegmentations and the phrase counts associated to each equivalence class and reuses it when possible. The equivalence class for a given partial bisegmentation is given by the sets $(\mathcal{SP}, \mathcal{TP})$ of source and target positions to be aligned.

Again, the complexity of Algorithm 3.3 is given by the number of nodes contained in the bisegmentation tree. Since now the equivalent nodes of the bisegmentation tree are processed only once, we have to calculate the number of equivalence classes. Each equivalence class may be reached from different father nodes. Thus, the complexity will be given by the number of equivalence classes multiplied by a constant representing the maximum number of father nodes that reach to an equivalence class. Since in the bisegmentation tree, two nodes can only be connected with edges labelled with phrase pairs of the set $\mathcal{BP}$, an upper bound for the maximum number of father nodes is $|\mathcal{BP}|$.

The number of equivalence classes is given by the number of possible combinations of the values for the sets $\mathcal{SP}$ and $\mathcal{TP}$. Given a source sentence f_1^J, there are only $J+1$ possible values for $\mathcal{SP}$, from $\mathcal{SP}_0$ to $\mathcal{SP}_J$: $\mathcal{SP}_0 = \{1, 2, ..., J\}$, $\mathcal{SP}_1 = \{2, 3, ..., J\}$, $\mathcal{SP}_2 = \{3, 4, ..., J\}$,..., $\mathcal{SP}_{J-1} = \{J\}$ and $\mathcal{SP}_J = \{\}$. Let $\mathcal{EC}$ be the set of equivalence classes and $\mathcal{EC}_{\mathcal{SP}}$ be the set of equivalence classes for a given set $\mathcal{SP}$. Then, the total number of equivalence classes is given by the expression:

$$|\mathcal{EC}| = \sum_{j=0}^{J} |\mathcal{EC}_{\mathcal{SP}_j}| \tag{3.7}$$

To calculate the number of equivalence classes for the set $\mathcal{SP}_j$, $|\mathcal{EC}_{\mathcal{SP}_j}|$, first we define two auxiliary subsets of the set $\mathcal{BP}$: $\mathcal{BP}_{j_1,j_2}$ and $\mathcal{BP}_{min}$. The set $\mathcal{BP}_{j_1,j_2}$ is composed of those phrase pairs $(f_{j_1}^{j_2}, e_{i_1}^{i_2}) \in \mathcal{BP}$ where the words $\{e_{i_1}, ..., e_{i_2}\}$ do not appear in phrase pairs $(f_{j'}^{j''}, e_{i'}^{i''}) \in \mathcal{BP}$ where $j'' < j_1$:

$$\begin{aligned} \mathcal{BP}_{j_1,j_2} = \{(f_{j_1}^{j_2}, e_{i_1}^{i_2}) \in \mathcal{BP} \,|\, \neg\exists(f_{j'}^{j''}, e_{i'}^{i''}) \in \mathcal{BP} \;:\; (j'' < j_1 \wedge \\ \{e_{i_1}, ..., e_{i_2}\} \cap \{e_{i'}, ..., e_{i''}\} \neq \emptyset)\} \end{aligned} \tag{3.8}$$

$\mathcal{BP}_{min}$ is defined as the set of phrase pairs $(f_{j_1}^{j_2}, e_{i_1}^{i_2}) \in \mathcal{BP}$ where $f_{j_1}^{j_2}$ is the shortest source phrase ending in position j_2 for which there exists (if $j_1 > 1$) at least one phrase pair in $\mathcal{BP}$ of the form $(f_{j'}^{j_1-1}, e_{i'}^{i''})$:

$$\begin{aligned} \mathcal{BP}_{min} = \{(f_{j_1}^{j_2}, e_{i_1}^{i_2}) \in \mathcal{BP} \,| \\ ((\neg\exists(f_{j'}^{j_2}, \cdot) \in \mathcal{BP} \;:\; j' > j_1 \wedge (j_1 = 1 \vee (f_{j''}^{j_1-1}, \cdot) \in \mathcal{BP})) \vee \\ (\exists(f_{j'}^{j_2}, \cdot) \in \mathcal{BP} \;:\; j' > j_1 \wedge (f_{j''}^{j'-1}, \cdot) \notin \mathcal{BP}))\} \end{aligned} \tag{3.9}$$

Once we have defined the sets $\mathcal{BP}_{j_1,j_2}$ and $\mathcal{BP}_{min}$, $|\mathcal{EC}_{\mathcal{SP}_j}|$ is given by the following recurrence:

$$|\mathcal{EC}_{\mathcal{SP}_j}| = \begin{cases} 1 & if\ j = 0 \\ 0 & if\ j > 0 \wedge \neg\exists(f_{j'}^{j}, e_{i_1}^{i_2}) \in \mathcal{BP}_{min} \\ |\mathcal{EC}_{\mathcal{SP}_{j'-1}}| \times |\mathcal{BP}_{j',j}| & if\ j > 0 \wedge \exists(f_{j'}^{j}, e_{i_1}^{i_2}) \in \mathcal{BP}_{min} \end{cases} \tag{3.10}$$

In the previous recurrence, $|\mathcal{EC}_{\mathcal{SP}_0}|$ is equal to 1 since given $\mathcal{SP}_0 = \{1, 2, ..., J\}$, there is only one possible value for $\mathcal{TP}$: $\{1, 2, ..., I\}$ (these values are set during the initialisation of Algorithm 3.3). Regarding the values of $|\mathcal{EC}_{\mathcal{SP}_j}|$, with $j > 1$, the set $\mathcal{BP}_{min}$ is used to determine if there are consistent phrase pairs whose source phrases end in position j. If there are no phrase pairs in $\mathcal{BP}_{min}$ whose source phrases end in position j, then $|\mathcal{EC}_{\mathcal{SP}_j}| = 0$. Otherwise, $\mathcal{BP}_{min}$ also provides the starting position j' of the shortest source phrase ending in position j. Under this circumstances, $|\mathcal{EC}_{\mathcal{SP}_j}|$ is given by the number of equivalence classes for $\mathcal{SP}_{j'-1}$, $|\mathcal{EC}_{\mathcal{SP}_{j'-1}}|$, multiplied by $|\mathcal{BP}_{j',j}|$; where $|\mathcal{BP}_{j',j}|$ represents the number of values that the set $\mathcal{TP}$ can take when we align the source positions $\{j', ..., j\}$. Note that the definition of $|\mathcal{BP}_{j',j}|$ given by Equation (3.8) allows us to avoid counting overlapping alignments for the target positions when we align the set of source positions $\{j', ..., j\}$ after having aligned the set of source positions $\{1, ..., j' - 1\}$.

According to the previous considerations, Algorithm 3.3 is able to achieve great savings in the computational cost with respect to the implementation given by Algorithm 3.2, specially if the word alignment matrices do not contain unaligned words. Let us consider the case in which we are given a source sentence composed of J words, where each source word is aligned with one and only one target word, and the word alignments are located in the main diagonal of the word alignment matrix A. Under these circumstances, there are a total of 2^{J-1} possible bisegmentations and only $J + 1$ equivalence classes, since according to Equation (3.10), $|\mathcal{EC}_{\mathcal{SP}_j}| = 1, \forall j \,|\, 0 \leq j \leq J$ (note that $|\mathcal{BP}_{j',j}| = 1$ for all possible values of j' and j). Since this is the worst case scenario, we can conclude that, if the word alignment matrices do not contain unaligned words, the time complexity of the algorithm is in $\mathcal{O}(J \cdot |\mathcal{BP}|)$ (as was explained above, $|\mathcal{BP}|$ is an upper bound for the maximum number of father nodes that reach to an equivalence class). If there are unaligned words in the word alignment matrices, the situation is completely different, since now $|\mathcal{BP}_{j',j}|$ may be greater than one. Under these circumstances and according to Equation (3.10), $|\mathcal{EC}_{\mathcal{SP}_j}|$ may grow geometrically with respect to $|\mathcal{EC}_{\mathcal{SP}_{j'-1}}|$. In spite of the fact that this may substantially increase the time complexity of the algorithm, a low rate of unaligned words in the word alignment matrices can be expected (this will be empirically demonstrated in section 5.5).

3.3.4 Possible Extensions and Applications of the Proposed Algorithms

The `brf_phr_extract_dp` algorithm allows to efficiently obtain the number of possible bisegmentations and the set of phrase counts extracted from $\mathcal{S}_{\mathcal{BP}(f_1^J, e_1^I, A)}$ for a given sentence pair (f_1^J, e_1^I) and its corresponding alignment matrix A. The algorithm can be straightforwardly modified to obtain more detailed information about the bisegmentation process, including information about phrase lengths or about reorderings. In addition to this, given a previously estimated phrase model, the proposed algorithm can also be modified to obtain the Viterbi alignment or the sum of the probability for each possible phrase alignment contained in $\mathcal{S}_{\mathcal{BP}(f_1^J, e_1^I, A)}$. These modified versions of the initial algorithm can be used to partially compute the E step of the EM algorithm. This partial computation can be justified by means of the sparse version of the EM algorithm proposed in [NH98].

Typically, the estimation techniques for phrase-based models that can be found in the literature rely on word alignments. This results in phrase translation tables composed of phrase pairs which are contained in the set $\mathcal{BP}$ of consistent phrase pairs. This may constitute an important limitation, since desirable phrases can be eliminated due to errors in the word alignments. Breaking this limitation is not trivial due to the huge size of the set $\mathcal{S}_{f_1^J, e_1^I}$. One possible solution to this problem would consist in obtaining a uniform random sample of the restricted set of bisegmentations $\mathcal{S}_{\mathcal{BP}(f_1^J, e_1^I, A)}$ combined in some way with the unrestricted set of bisegmentations $\mathcal{S}_{f_1^J, e_1^I}$. From the resulting set of random samples, a phrase-based model could be built by collecting counts or using a better theoretically founded estimation technique such as the Monte-Carlo EM algorithm described in [WT90]. It is easy to obtain

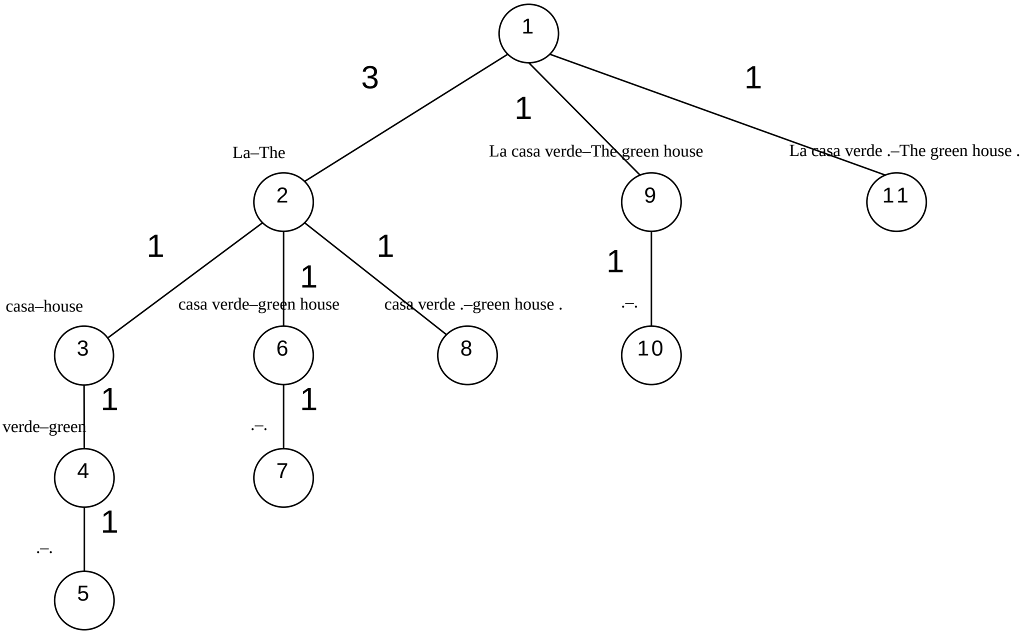

1
3
1
1
La–The
La casa verde–The green house
La casa verde .–The green house .
2
9
11
1
1
1
1
casa–house
casa verde–green house
casa verde .–green house .
.–.
3
6
8
10
1
1
verde–green
.–.
4
7
1
.–.
5

Figure 3.5: Tree of possible bisegmentations including the required information to generate random walks: each edge of the tree is labeled with the number of reachable leafs.

random samples from the set $\mathcal{S}_{f_1^J,e_1^I}$ due to the regularities that this set presents. By contrast, the generation of random samples from the set $\mathcal{S}_{\mathcal{BP}(f_1^J,e_1^I,A)}$ is not easy, since this set does not present such regularities.

The `brf_phr_extract_dp` algorithm obtains the number of bisegmentations for each equivalence class (or, in other words, the number of bisegmentations that can be reached following a given edge of the tree) as a by-product. As it will be shown, this information can be exploited to obtain random samples from the set $\mathcal{S}_{\mathcal{BP}(f_1^J,e_1^I,A)}$.

Since the set of possible bisegmentations presents a tree structure, our problem of generating random samples is equivalent to the problem of generating random paths in the bisegmentation tree. One way to generate such random paths consists in, starting from the root of the tree, randomly select which one of the edges of the current node is added to the path. This is often referred to as a *random walk*. Unfortunately, this straightforward technique priorizes the random walks with smallest lengths (or, in other words, the bisegmentations with smallest lengths). To solve this problem, we can assign weights to each edge depending on the number of bisegmentations that can be reached from it. For this purpose, we would need a labeled bisegmentation tree as the one presented in Figure 3.5, where each edge is labeled with the number of leafs that can be accessed from it. These labels can be efficiently obtained by means of the `brf_phr_extract_dp` algorithm.

After labeling the bisegmentation tree, Algorithm 3.4 generates random walks with uni-

```
input  : $f_1^J$,$e_1^I$,$A$,$\mathcal{BP}$,
         $S_{\forall(\mathcal{SP},\mathcal{TP})}$ (bisegmentation size for each equivalence class $(\mathcal{SP},\mathcal{TP})$),
         $\mathcal{SP}$ (set of source positions that are to be aligned),
         $\mathcal{TP}$ (set of target positions that are to be aligned),
         $\pi_{f_1^J}$ (partial partition of the source sentence),
         $\pi_{e_1^I}$ (partial partition of the target sentence)
output : $\Pi_{f_1^J}$ (partition of the source sentence),
         $\Pi_{e_1^I}$ (partition of the target sentence)
auxiliar: $x$ (random number in $(0,1]$)
begin
    if $\mathcal{SP} \neq \emptyset$ then
        $j_1 := min(\mathcal{SP})$
        $j_2 := max(\mathcal{SP})$
        $j := j_1$
        $n := 0$
        while $j \leq j_2$ do
            forall $(f_{j_1}^j, e_{i_1}^{i_2}) \in \mathcal{BP} \wedge \{i_1, i_1+1, ..., i_2\} \in \mathcal{TP}$ do
                $n := n+1$
                $\mathcal{SP}^n := \mathcal{SP} - \{j_1, j_1+1, ..., j\}$
                $\mathcal{TP}^n := \mathcal{TP} - \{i_1, i_1+1, ..., i_2\}$
                $\pi_{f_1^J}^n := (\pi_{f_1^J}; f_{j_1}^j)$
                $\pi_{e_1^I}^n := (\pi_{e_1^I}; e_{i_1}^{i_2})$
            $j := j+1$
        $x :=$rand()
        $N := n$
        $k \,|\, 1 \leq k \leq N \wedge \frac{\sum_{n=1}^{n=k-1} S_{(\mathcal{SP}^n,\mathcal{TP}^n)}}{\sum_{n=1}^{n=N} S_{(\mathcal{SP}^n,\mathcal{TP}^n)}} < x \leq \frac{\sum_{n=1}^{n=k} S_{(\mathcal{SP}^n,\mathcal{TP}^n)}}{\sum_{n=1}^{n=N} S_{(\mathcal{SP}^n,\mathcal{TP}^n)}}$
        bisegm_random_walk($f_1^J$,$e_1^I$,$A$,$\mathcal{BP}$,$S_{\forall(\mathcal{SP},\mathcal{TP})}$,$\mathcal{SP}^k$,$\mathcal{TP}^k$,$\pi_{f_1^J}^k$,$\pi_{e_1^I}^k$)
    else
        $\Pi_{f_1^J} := \pi_{f_1^J}$
        $\Pi_{e_1^I} := \pi_{e_1^I}$
end
```

Algorithm 3.4: Pseudocode for the `bisegm_random_walk` algorithm.

form probability for the set of possible bisegmentations. For this purpose, at each step of the generation of the random walk, if there are a total of N candidate edges to be extended, the k'th edge of the tree is chosen by generating a random number between zero and one. Each one of the N candidate edges is assigned a probability equal to the number of leafs that can be reached from it divided by the number of leafs that can be reached from the N edges.

3.4 Phrase-based Model Estimation from Very Large Corpora

The translation of a given source sentence using phrase-based models requires that all its constituent phrases have been seen during the training process. This requirement is hard to achieve in practise, since the number of phrases to be learnt for a given language is huge. As a result of this, phrase-based models have a poor generalisation capability.

A common strategy to overcome the lack of generality that the phrase-based models present consists in the acquisition of larger and larger training corpora. Recently, very large corpora have been made available, as for example, the Europarl corpus, or the Arabic-English corpus used in the annual NIST machine translation evaluation campaigns. These corpora are composed of millions of sentence pairs and tens of millions of running words.

However, the use of larger corpora extraordinarily increases the number of parameters to be learnt and the memory size required to store them. For example, Callison Burch et al. [CBBS05] report that the storage of the parameters of a phrase-based model for the above mentioned NIST Arabic-English corpus may require up to 30 GBytes of memory if standard lookup tables are used. Therefore, dealing with very large corpora has become a bottleneck in SMT, as has happened in other well-known pattern recognition tasks.

One typical way to reduce the number of parameters contained in a phrase model is to impose a constraint over the length of the phrases. Such a constraint does not affect negatively the translation quality if the maximum phrase length allowed is sufficiently high. However, this constraint is not enough to solve the above-mentioned scalability problems of the phrase-based models, as it will be shown in section 3.4.1.

Most of the authors of works on PB-SMT have shown the importance of dealing with larger corpora and longer sentences in order to build statistical machine translation systems. For example, [KOM03] concludes that longer phrases results in better translation quality but at the cost of managing huge translation lookup tables. Also, in [Til03, VVW03, VZH$^+$03] this problem is also apparent.

3.4.1 Some Model Statistics

To illustrate the huge size of the phrase tables we have computed some statistics for different phrase models estimated from the Europarl corpus[c]. Table 3.3 shows the number of phrase pairs, Spanish words and English words contained in eight phrase-models ranging over the maximum phrase length for both languages. As it can be observed, the size of the models may become huge if the maximum phrase length is increased.

3.4.2 Training Procedure

Even if very efficient data structures in terms of space complexity are used, important problems arise when phrase models are to be estimated from very large corpora. In order to overcome this limitation, we propose an algorithm which trains phrase models from corpora of an arbitrary size.

[c] We used the standard estimation technique described in section 3.2 to obtain the phrase-based models

Table 3.3: Statistics of different phrase models estimated from the Europarl corpus ranging over the maximum phrase size (denoted by **m**).

m	**Phrases**	**English words**	**Spanish words**
1	937 433	2 085 329	937 433
2	4 313 219	10 840 699	7 689 005
3	9 969 817	29 078 625	24 658 799
4	16 500 370	55 378 535	50 781 011
5	22 935 135	87 051 762	82 954 836
6	28 928 890	122 136 760	118 917 366
7	31 227 305	159 525 055	157 425 528
8	39 456 451	198 444 311	197 636 688

One possible way to solve this problem, which allows to train corpora of an arbitrary size, is given by the algorithm `frag_by_frag_training` given in Algorithm 3.5. This algorithm is based on the use of phrase counts instead of probabilities and works as follows:

1. First, it splits the corpus (given in the file $align\_file$) into fragments of a fixed number of aligned sentence pairs ($fragment\_size$). The splitting technique that is used here is very simple and corpus independent. In addition to this, its functionality is implemented by standard tools as for example the ”split” command which can be found in UNIX-like operating systems.

2. Then, a phrase model estimation process is carried out for each fragment. The *submodel* for each fragment will be composed of a series of phrase counts labelled with an identifier associated to the corpus fragment from which the counts were extracted (fid); by this reason, such identifier is required by the training algorithm.

3. Finally, once the submodels have been generated, they are merged into a single file. This file is lexicographically ordered and the phrase counts that compose the model are then merged. The lexicographical ordering of the input file allows us to merge the counts without having to store the whole model in memory.

The labelling process which has been mentioned above is done to allow a correct merging of the phrase counts. Figure 3.6 shows an example of a file containing sorted counts with their corresponding labels.

The details of the merging process can be found in the algorithm `merge_counts` given by Algorithm 3.6. Such an algorithm takes a file with sorted counts and returns the final phrase-based model. The `merge_counts` algorithm takes advantage of the lexicographical ordering of the input file to merge the counts without having to store the whole model in memory. Specifically, the algorithm reads a block of model entries sharing the same target phrase (line 4). After that, the contribution of each entry of the block to the target (line 10) and joint counts (line 12) are processed. Finally, the counts for each phrase pair are written to file (line 15).

The above mentioned process yields a phrase model composed of a set of phrase counts that is identical to the one obtained from the whole corpus. It is worth noticing that the

```
input  : $fragment\_size$, $align\_file$ (alignment file)
output : $phr\_model$ (phrase-based model)
begin
    $fragments$ := split($align\_file$, $fragment\_size$)
    open($counts\_file$,'a') // open for appending
    $fid := 1$
    forall $f \in fragments$ do
        train($f, fid$) >> $counts\_file$ // ''>>'' means append to file
        $fid := fid + 1$
    $sorted\_counts\_file$ := sort_counts($counts\_file$)
    $phr\_model$ := merge_counts($sorted\_counts\_file$)
end
```

Algorithm 3.5: Pseudocode for the `frag_by_frag_training` algorithm.

```
$trg\_phr1$ ||| $src\_phr1$ ||| $c(trg\_phr1, fid1)$ $c((trg\_phr1, src\_phr1), fid1)$ $fid1$
$trg\_phr1$ ||| $src\_phr2$ ||| $c(trg\_phr1, fid1)$ $c((trg\_phr1, src\_phr2), fid1)$ $fid1$
$trg\_phr1$ ||| $src\_phr2$ ||| $c(trg\_phr1, fid2)$ $c((trg\_phr1, src\_phr2), fid2)$ $fid2$
$trg\_phr1$ ||| $src\_phr3$ ||| $c(trg\_phr1, fid3)$ $c((trg\_phr1, src\_phr3), fid3)$ $fid3$
$trg\_phr2$ ||| $src\_phr3$ ||| $c(trg\_phr2, fid2)$ $c((trg\_phr2, src\_phr3), fid2)$ $fid2$
$trg\_phr2$ ||| $src\_phr3$ ||| $c(trg\_phr2, fid3)$ $c((trg\_phr2, src\_phr3), fid3)$ $fid3$
$trg\_phr3$ ||| $src\_phr3$ ||| $c(trg\_phr3, fid1)$ $c((trg\_phr3, src\_phr3), fid1)$ $fid1$
...
```

Figure 3.6: Example of a file containing sorted counts.

obtained set of counts allows us to efficiently generate direct and inverse phrase model probabilities if appropriate data structures are used. One example of the data structures that allow us to efficiently calculate direct and inverse phrase probabilities from phrase counts will be presented in section 4.3.

The proposed algorithm introduces time overhead because of the necessity of sorting and merging the phrase counts. This overhead will be empirically measured in section 5.1. However, it is important to stress that the training and sorting steps executed by the algorithm can be parallelised, resulting in a very efficient method to train phrase models.

It is worth pointing out that our proposed estimation algorithm constitutes an application of the *MapReduce* software framework [DG04]. MapReduce builds on the observation that many tasks have the same basic structure: a computation is applied over a large number of records (e.g., parallel sentences) to generate partial results (map step), which are then aggregated in some fashion (reduce step). The MapReduce software framework has been applied to estimate phrase-based models and word alignment models in other works, such as [DCML08, GV08]. The estimation methods described in these works are similar to the one described above.

```
input    : $sorted\_counts\_file$ (sorted counts file)
output   : $phr\_model$ (phrase-based model)
auxiliar: $BE$ (block of entries of the file with sorted counts)
          $\mathcal{F}$ (set of fragment identifiers)
          $JC_s$ (joint count for source phrase $s$)
          $(t, s, ftc, fjc, fid)$ (entry of the file with sorted counts)
          $\mathcal{S}$ (set of source phrases)
begin
    open(sorted_counts_file,'r'); open(phr_model,'w')
    while not end_of(sorted_counts_file) do
        $BE :=$ read_block_with_same_trg_phr$(sorted\_counts\_file)$
        // The $BE$ variable stores the next block
        // of entries which shares the same target phrase
        $tc := 0$
        $\mathcal{F} := \emptyset$
        $JC_s := 0, \forall s$
        forall $(t, s, ftc, fjc, fid) \in BE$ do
            // An entry of $BE$ includes:  target phrase ($t$),
            // source phrase ($s$), target count ($ftc$),
            // joint count ($fjc$), fragment identifier ($fid$)
            if $fid \notin \mathcal{F}$ then
                $tc := tc + ftc$
                $\mathcal{F} := \mathcal{F} \cup fid$
            $JC_s := JC_s + fjc$
            if $s \notin \mathcal{S}$ then $\mathcal{S} := \mathcal{S} \cup s$
        forall $s \in \mathcal{S}$ do
            write(phr_model,(t,s,tc,$JC_s$))
end
```

Algorithm 3.6: Pseudocode for the `merge_counts` algorithm.

3.5 Specific Phrase-Based Model Derivation

The assumptions that are typically made during the derivation of phrase based models reduce them to mere statistical dictionaries of phrase pairs plus a simple distortion model. These are very strong assumptions, since the translation process using phrase-based models requires the generation of bisegmentations of the source and the target sentences, and the bisegmentation process can be carried out in a huge number of ways, as was explained in section 3.3.1.

In this section we propose a phrase-based model derivation that allows us to incorporate specific probability distributions governing the basic aspects of the bisegmentation process. As will be shown in section 3.5.3, the resulting submodels that are obtained by means of our proposed model derivation can also be added as individual components of a log-linear translation model. In this sense, our model derivation provides a well-founded criterion to add certain components to log-linear translation models.

Additionally, the model derivation presented here is specifically designed for its use by standard search algorithms. As it was explained in section 1.6, search algorithms for SMT typically generate their translations by adding words to the translation hypotheses from left to right. Informally, the generative process followed by these search algorithms has three steps:

1. Choose the next word or group of words of the source sentence to be translated.
2. Choose the target translation of the untranslated source words, and append it at the right of the hypothesis.
3. If there still are untranslated source words, goto 1.

This generative process contrasts with the standard generative process for phrase-based models described in section 1.4.3. In such generative process, the source sentence is first divided into phrases, then, the target phrases that translate each source phrase are chosen, and finally, the target phrases are reordered to compose the target sentence. As will be shown, our proposed generative process reflects the way in which standard search algorithms work, allowing a more natural implementation of such algorithms.

3.5.1 Generative Process

Given the probability distribution $Pr(f_1^J|e_1^I)$, we define the generative process of our proposed phrase model as follows:

1. Choose K as the length of the bisegmentation, where $1 \leq K \leq \min(I,J)$
2. For $k = 1$ to K do:
   (a) Choose the ending position of the k'th phrase in e_1^I, a_k, determining the k'th target phrase, $e_{a_{k-1}+1}^{a_k}$, where $a_0 = 0$ and $a_k > a_{k-1}$
   (b) Choose the number of *skipped* words for the k'th phrase in f_1^J, b_k, with respect to the ending position β_{k-1} of the previous source phrase (b_k is an integer number)
   (c) Choose the length of the k'th phrase of f_1^J, c_k, determining the k'th source phrase, $f_{\alpha_k}^{\beta_k}$, where c_k should be greater than zero and:
   $$\begin{array}{rcl} \alpha_k & = & \beta_k - c_k + 1 \\ \beta_k & = & \beta_{k-1} + b_k + c_k \\ \beta_0 & = & 0 \end{array}$$
   (d) Choose the k'th phrase of f_1^J, $f_{\alpha_k}^{\beta_k}$, as translation of the k'th phrase of e_1^I, $e_{a_{k-1}+1}^{a_k}$

The bisegmentation decisions made during the generative process are summarised by the hidden variables K, a_1^K, b_1^K and c_1^K, which store the bisegmentation length, the ending positions of the target phrases, the number of skipped source words with respect to the previous source phrase and the length of the source phrases for the k'th bisegmentation step respectively.

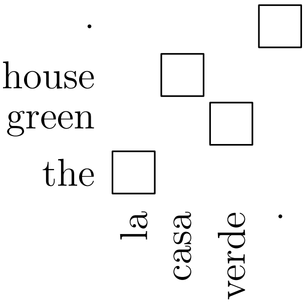
At each step of the *for* loop of the generative process, the hidden variables a_1^K, b_1^K and c_1^K are extended with a new value. The newly added values have to be chosen so as to generate correct bisegmentations of the source and the target sentences. Correct bisegmentations are composed of a set of phrases containing all the words of the source and the target sentences without overlappings. This can be formally stated as follows:

$$e_1^{a_1} e_{a_1+1}^{a_2} ... e_{a_{K-1}+1}^{a_K} \equiv e_1^I \tag{3.11}$$

$$L\big(f_{\alpha_1}^{\beta_1} f_{\alpha_2}^{\beta_2} ... f_{\alpha_K}^{\beta_K}, [(\alpha_1, \beta_1), (\alpha_2, \beta_2), ..., (\alpha_K, \beta_K)]\big) \equiv f_1^J \tag{3.12}$$

where the function $L(\cdot)$ takes the concatenation of the source phrases obtained during the generative process and their associated indices as input, and returns a linearisation of such concatenation of source phrases (for instance, given the source sentence $f_1 f_2 f_3$, the concatenation $f_2^3 f_1$ and the indices $[(2,3)(1,1)]$, $L(f_2^3 f_1, [(2,3), (1,1)]) = f_1 f_2 f_3$).

As an example of how the above explained generative process works, Figure 3.7 shows a bisegmentation example for a sentence pair and the corresponding values of the hidden variables according to the generative process.



$$\boxed{\begin{array}{l} K = 4 \\ a_1^4 = \{1, 2, 3, 4\} \\ b_1^4 = \{0, 1, -2, 1\} \\ c_1^4 = \{1, 1, 1, 1\} \end{array}}$$

Figure 3.7: Bisegmentation example for a sentence pair (left) and the set of values for the hidden variables according to our proposed generative process (right).

3.5.2 Model Derivation

The generative process described in the previous section can be more formally expressed as follows:

$$Pr(f_1^J|e_1^I) = Pr(J|e_1^I) \cdot \sum_{K, a_1^K, b_1^K, c_1^K} Pr(f_1^J, K, a_1^K, b_1^K, c_1^K | e_1^I, J) \tag{3.13}$$

$$= Pr(J|e_1^I) \cdot \sum_K Pr(K|e_1^I, J) \cdot \sum_{a_1^K, b_1^K, c_1^K} Pr(f_1^J, a_1^K, b_1^K, c_1^K | e_1^I, J, K) \tag{3.14}$$

In the previous equation, the probability distribution $Pr(J|e_1^I)$ is introduced for completeness and the term $Pr(f_1^J, a_1^K, b_1^K, c_1^K | e_1^I, J, K)$ can be decomposed without loss of generality as a product for each step of the bisegmentation process:

$$
\begin{aligned}
Pr(f_1^J, a_1^K, b_1^K, c_1^K | e_1^I, J, K) = \prod_{k=1}^{K} \Big[& Pr(a_k | e_1^I, J, K, a_1^{k-1}, b_1^{k-1}, c_1^{k-1}, f_{\alpha_1}^{\beta_1}, ..., f_{\alpha_{k-1}}^{\beta_{k-1}}) \\
& Pr(b_k | e_1^I, J, K, a_1^{k}, b_1^{k-1}, c_1^{k-1}, f_{\alpha_1}^{\beta_1}, ..., f_{\alpha_{k-1}}^{\beta_{k-1}}) \\
& Pr(c_k | e_1^I, J, K, a_1^{k}, b_1^{k}, c_1^{k-1}, f_{\alpha_1}^{\beta_1}, ..., f_{\alpha_{k-1}}^{\beta_{k-1}}) \\
& Pr(f_{\alpha_k}^{\beta_k} | e_1^I, J, K, a_1^{k}, b_1^{k}, c_1^{k}, f_{\alpha_1}^{\beta_1}, ..., f_{\alpha_{k-1}}^{\beta_{k-1}}) \Big]
\end{aligned}
\quad (3.15)
$$

The previous decomposition of $Pr(f_1^J|e_1^I)$ exactly reflects the generative process given in section 3.5.1. From this point, we are forced to make a series of assumptions to obtain a tractable expression of the model:

$$Pr(J|e_1^I) \approx p(J|I) \quad (3.16)$$

$$Pr(K|e_1^I, J) \approx p(K|I, J) \quad (3.17)$$

$$Pr(a_k | e_1^I, J, K, a_1^{k-1}, b_1^{k-1}, c_1^{k-1}, f_{\alpha_1}^{\beta_1}, ..., f_{\alpha_{k-1}}^{\beta_{k-1}}) \approx p(a_k|a_{k-1}) \quad (3.18)$$

$$Pr(b_k | e_1^I, J, K, a_1^{k}, b_1^{k-1}, c_1^{k-1}, f_{\alpha_1}^{\beta_1}, ..., f_{\alpha_{k-1}}^{\beta_{k-1}}) \approx p(b_k) \quad (3.19)$$

$$Pr(c_k | e_1^I, J, K, a_1^{k}, b_1^{k}, c_1^{k-1}, f_{\alpha_1}^{\beta_1}, ..., f_{\alpha_{k-1}}^{\beta_{k-1}}) \approx p(c_k|a_k, a_{k-1}) \quad (3.20)$$

$$Pr(f_{\alpha_k}^{\beta_k} | e_1^I, J, K, a_1^{k}, b_1^{k}, c_1^{k}, f_{\alpha_1}^{\beta_1}, ..., f_{\alpha_{k-1}}^{\beta_{k-1}}) \approx p(f_{\alpha_k}^{\beta_k} | e_{a_{k-1}+1}^{a_k}) \quad (3.21)$$

The resulting model is composed of the following submodels:

- **Source length submodel** ($p(J|I)$): generates probabilities for the length of the source sentence given the length of the target sentence.

- **Bisegmentation length submodel** ($p(K|I, J)$): assigns probabilities to each segmentation length given the lengths of the source and the target sentences.

- **Target phrase length submodel** ($p(a_k|a_{k-1})$): assigns probabilities to the ending position of a target phrase given the ending position of the previous target phrase. This submodel can be seen as a submodel for the length of the target phrases.

- **Distortion submodel** ($p(b_k)$): this submodel assigns probabilities to the number of skipped source words with respect to the ending position of the last aligned source phrase. This submodel can be seen as a distortion model.

- **Source phrase length submodel** ($p(c_k|a_k, a_{k-1})$): assigns probabilities to the length of the source phrases given the ending positions of the current and the previous target phrases. This submodel can be seen as a source phrase length submodel given the length of the target phrase.

- **Phrase translation submodel** ($p(f_{\alpha_k}^{\beta_k}|e_{a_{k-1}+1}^{a_k})$): this submodel constitutes a statistical dictionary of phrase pairs.

After making the above explained assumptions, the expression of our proposed phrase model is as follows:

$$
\begin{aligned}
p(f_1^J|e_1^I) \quad = \quad & p(J|I)\sum_K \Bigg[p(K|I,J)\cdot \\
& \sum_{a_1^K,b_1^K,c_1^K} \Bigg(\prod_{k=1}^{K} p(a_k|a_{k-1})\cdot \\
& \qquad p(b_k)\cdot p(c_k|a_k,a_{k-1})\cdot \\
& \qquad p(f_{\alpha_k}^{\beta_k}|e_{a_{k-1}+1}^{a_k}) \Bigg) \Bigg]
\end{aligned}
\tag{3.22}
$$

We make additional assumptions to generate probabilities for each submodel:

- $p(J|I) = \phi_I(J+0.5) - \phi_I(J-0.5)$,
  where $\phi_I(\cdot)$ denotes the cumulative distribution function (cdf) for the normal distribution (the cdf is used here to integrate the normal density function over an interval of length 1). We use a specific normal distribution with mean $\mu_{|e_1^I|}$ and standard deviation $\sigma_{|e_1^I|}$ for each possible target sentence length $|e_1^I|$.

- $p(K|I,J) = \frac{1}{\min{(I,J)}}$,
  that is, we use a uniform probability distribution.

- $p(a_k|a_{k-1}) = \delta(1-\delta)^{a_k-a_{k-1}-1}$,
  where we propose the use of a geometric distribution with probability of success δ on each trial to assign probabilities to the length of the target phrases. The use of a geometric distribution penalises long target phrases. Alternatively, a uniform distribution can be used: $p(a_k|a_{k-1}) = \frac{1}{I_{max}-a_{k-1}}$, where I_{max} is a constant representing the maximum target sentence length. Roughly speaking, such distribution would penalise the length of the bisegmentations.

- $p(b_k) = \frac{1}{2-\delta}\delta(1-\delta)^{abs(b_k)}$

  where we propose the use of a modified geometric distribution with probability of success δ on each trial to assign probabilities to the number of skipped source words. The original geometric distribution, which is defined for positive numbers, is modified here because b_k takes integer values. Specifically, the scaling factor $\frac{1}{2-\delta}$ is added. The use of a geometric distribution penalises longer reorderings.

- $p(c_k|a_k,a_{k-1}) = \frac{1}{1+\tau}\delta(1-\delta)^{abs(c_k-(a_k-a_{k-1}))}$

  where $\tau = \sum_{i=1}^{a_k-a_{k-1}-1}\delta(1-\delta)^i$. We again propose the use of a modified geometric distribution with probability of success δ on each trial to assign probabilities to the length of the source phrases given the length of the target phrases. The original geometric distribution is modified here because the term $c_k - (a_k - a_{k-1})$ takes integer values (c_k and $(a_k - a_{k-1})$ are greater than zero). In this case, the scaling factor $\frac{1}{1+\tau}$ was introduced. The use of a geometric distribution penalises the difference between the source and the target phrase lengths. Alternatively, a Poisson distribution or a uniform distribution (in this case, the length of the bisegmentations would be penalised) could have been used.

Regarding the phrase translation submodel ($p(f_{\alpha_k}^{\beta_k}|e_{a_{k-1}+1}^{a_k})$), its probabilities are obtained by means of the RF or the BRF estimation techniques described in sections 3.2 and 3.3 respectively.

In the final expression of our model, some of the resulting submodels are *deficient*. We say that a statistical model is deficient if it has the property of not concentrating all of its probability on events that can be explained by the generative process. As it is noted in [BDDM93], deficiency is the price we pay to obtain tractable model expressions. In our case, the source phrase length submodel, the target phrase length submodel and the distortion submodel are deficient. Specifically, the target phrase length model may assign probabilities greater than zero to bisegmentations that do not satisfy the condition given by Equation (3.11). The deficiency of the target phrase length model can be corrected by slightly relaxing the modelling assumptions, obtaining the following submodel: $p(a_k|I, K, k, a_{k-1})$. Regarding the source phrase length submodel and the reordering submodel, they may assign probabilities greater than zero to bisegmentations that do not satisfy the condition given by Equation (3.12). In this case, the required modifications to obtain non-deficient models are more complex since zero- or one-order dependencies do not add enough information to ensure that the correctness condition holds.

3.5.3 Log-Linear Model

As stated in section 1.4.4, log-linear models constitute the state-of-the-art in statistical machine translation. In this section we will show how our phrase-based model derivation can be used as a criterion to add components to a log-linear model for SMT.

According to Equation (3.13), and following the maximum-approximation, the fundamental equation of machine translation (see section 1.3) can be reframed as:

$$\hat{e}_1^{\hat{I}} \approx \arg\max_{I,e_1^I} \left\{ Pr(e_1^I) \cdot \max_{K,a_1^K,b_1^K,c_1^K} Pr(f_1^J, K, a_1^K, b_1^K, c_1^K \mid e_1^I) \right\} \tag{3.23}$$

Following the log-linear approach, Equation (3.23) can be rewritten as follows:

$$\hat{e}_1^{\hat{I}} = \arg\max_{I,e_1^I} \{ \max_{K,a_1^K,b_1^K,c_1^K} \sum_{m=1}^{M} \lambda_m \cdot h_m(f_1^J, K, a_1^K, b_1^K, c_1^K, e_1^I) \} \tag{3.24}$$

According to Equation (3.24), we introduce a set of seven feature functions in our log-linear model (from h_1 to h_7): an n-gram language model (h_1), a source sentence-length model (h_2), inverse and direct phrase-based models (h_3 and h_4 respectively), a target phrase-length model (h_5), a source phrase-length model (h_6), and a distortion model (h_7). The details for each feature function are listed below:

- **n-gram language model (h_1)**
  $h_1(e_1^I) = \prod_{i=1}^{I+1} p(e_i|e_{i-n+1}^{i-1})$ [d]

[d] e_0 denotes the *begin-of-sentence* symbol (BOS), e_{I+1} denotes the *end-of-sentence* symbol (EOS), $e_i^j \equiv e_i...e_j$

- **source sentence-length model (**h_2**)**
  $h_2(e_1^I, f_1^J) = \log(p(J|I))$

- **inverse phrase-based model (**h_3**)**
  $h_3(e_1^I, K, a_1^K, b_1^K, c_1^K, f_1^J) = \log(\prod_{k=1}^{K} p(f_{\alpha_k}^{\beta_k}|e_{a_{k-1}+1}^{a_k}))$

- **direct phrase-based model (**h_4**)**
  $h_4(e_1^I, K, a_1^K, b_1^K, c_1^K, f_1^J) = \log(\prod_{k=1}^{K} p(e_{a_{k-1}+1}^{a_k}|f_{\alpha_k}^{\beta_k}))$

- **target phrase-length model (**h_5**)**
  $h_5(K, a_1^K) = \log(\prod_{k=1}^{K} p(a_k|a_{k-1}))$

- **source phrase-length model (**h_6**)**
  $h_6(K, a_1^K, c_1^K) = \log(\prod_{k=1}^{K} p(c_k|a_k, a_{k-1}))$

- **distortion model (**h_7**)**
  $h_7(K, b_1^K) = \log(\prod_{k=1}^{K} p(b_k))$

It is worthy of note that, except h_4, all the above described log-linear components have been obtained from a proper decomposition of the probability distribution $Pr(e_1^I|f_1^J)$.

3.6 Summary

In this chapter we have studied different aspects of the modelling and the training problems in PB-SMT. We have proposed an alternative technique to train phrase models which we have called BRF estimation. BRF estimation tries to reduce the strong heuristic component that the standard estimation technique presents by counting the frequencies of the phrase pairs in the set of possible bisegmentations. Since the set of possible bisegmentations has a huge size, we prune it by means of the set of consistent phrase pairs in which the standard estimation technique is based. Specifically, only those bisegmentations that are compatible with the set of consistent pairs are considered when collecting the bilingual counts. As it has been shown, our BRF estimation technique can be efficiently implemented by means of dynamic programming techniques.

We have described an estimation technique able to work with very large corpora. Our proposed technique allows to transform main memory requirements into hard disk requirements. In contrast with existing estimation techniques, our proposed technique is able to obtain the necessary information to generate direct and inverse phrase pair probabilities by executing the estimation algorithm in only one translation direction.

Finally, we have given a specific phrase-based model derivation. The generative process associated to this derivation generates the translations from left to right as well as the regular decoding algorithms described in the literature do. Our proposed phrase model includes a set of submodels governing different aspects of the bisegmentation process, such as the length of the source and the target phrases, reordering of the phrases, etc. In addition to this, we have described a log-linear model which includes all these submodels as components, resulting in a fully-fledged state-of-the-art statistical translation model.

Chapter 4

Search for Phrase-Based Statistical Machine Translation

4.1 Introduction

As was already explained in section 1.3, the building process of an SMT system involves addressing three problems, namely, the modelling, the training and the search problems. In the previous chapter, different solutions for the modelling and the training problems in PB-SMT were proposed. In this chapter we study the search problem in PB-SMT.

The goal of the search, also referred to as generation or decoding, is to find the best translation candidate for a given source sentence among all possible target language sentences. For this purpose, search algorithms explore a graph-structured search space which represents the set of possible translations. This search space exploration is guided by the statistical models involved in the translation process. A specific translation of a source sentence is given by a path in the graph representing the search space. The best translation is given by the path of highest probability.

The rest of this chapter is organised as follows: a branch-and-bound search algorithm for PB-SMT is described in section 4.2. Specific decoding techniques to deal with large phrase-based models are described in section 4.3. In section 4.4 we formalise the concept of phrase-level alignment for a sentence pair. Also, a modification of the branch-and-bound search algorithm to find the phrase-level alignment of highest probability is proposed. Finally, we provide a summary of the chapter in section 4.5.

4.2 Branch-and-Bound Search for PB-SMT

In this section we propose a specific way to solve the search problem in PB-SMT based on the well-known branch-and-bound paradigm [LD60].

The search problem in SMT is formally defined as a maximisation problem where the goal is to find the target translation e_1^I of highest probability given the source sentence f_1^J. As was explained in section 1.3, the most basic formulation of this probability is given by the product of the language and translation model probabilities.

The language and translation models can be instantiated in different ways. Here we propose the use of an n-gram model as language model and a phrase-based model as translation model. This phrase-based model is defined according to the derivation presented in section 3.5. In such derivation, the translation process for a sentence pair is explained by means of a specific set of hidden alignment variables (K, a_1^K, b_1^K, c_1^K). The meaning of each hidden variable is the following:

- K: bisegmentation length.
- a_1^K: vector of ending positions of the K target phrases.
- b_1^K: vector with the number of skipped source positions with respect to the ending position of the previously aligned source phrase.
- c_1^K: vector of lengths of the K source phrases.

Given the previous assumptions and following the maximum-approximation, the search problem can be formally expressed as follows:

$$\begin{aligned}\hat{e}_1^{\hat{I}} &= \underset{I,e_1^I}{\arg\max}\{Pr(e_1^I)\cdot Pr(f_1^J|e_1^I)\}\\ &\approx \underset{I,e_1^I}{\arg\max}\left\{\prod_{i=1}^{I} p(e_i|e_{i-n+1}^{i-1})\cdot p(J|I)\cdot\right.\\ &\left.\max_{K,a_1^K,b_1^K,c_1^K}\prod_{k=1}^{K}\left[p(a_k|a_{k-1})\cdot p(b_k)\cdot p(c_k|a_k,a_{k-1})\cdot p(f_{\alpha_k}^{\beta_k}|e_{a_{k-1}+1}^{a_k})\right]\right\}\end{aligned} \tag{4.1}$$

where the following submodels are included: an n-gram language model, $p(e_i|e_{i-n+1}^{i-1})$, a source sentence length submodel, $p(J|I)$, a bisegmentation length submodel, $p(K|I,J)$, a target phrase length submodel, $p(a_k|a_{k-1})$, a reordering submodel, $p(b_k)$, a source phrase length submodel, $p(c_k|a_k,a_{k-1})$, and an inverse phrase translation submodel, $p(f_{\alpha_k}^{\beta_k}|e_{a_{k-1}+1}^{a_k})$; the α and β variables are defined as follows:

$$\begin{aligned}\alpha_k &= \beta_k - c_k + 1\\ \beta_k &= \beta_{k-1} + b_k + c_k\\ \beta_0 &= 0\end{aligned}$$

The maximisation problem given by Equation (4.1) can be solved using different search algorithms. Since the search problem in SMT has been demonstrated to be NP-complete [Kni99, UM06], we cannot expect to develop efficient search algorithms that obtain the optimal solution. In the search algorithms that have been proposed so far, the well-known technique of dynamic programming [Bel57] is combined with techniques that introduce certain restrictions in the search space, such as beam search [Jel98]. As a consequence of the introduction of such restrictions, the resulting search algorithm does not guarantee finding the optimal solution.

As was explained in section 1.6, the vast majority of the search algorithms start from a null-hypothesis and iteratively extend partial hypotheses by adding words from left to right. This iterative process is repeated until a complete hypothesis has been generated. The hypothesis extension procedure is driven by the statistical models involved in the translation process.

For the sake of simplicity, we will work with the maximisation following the Bayes rule given by Equation (4.1). However, this formulation can be straightforwardly extended to the log-linear model defined in section 3.5.3.

The remaining part of this section is structured as follows: we present a dynamic programming algorithm to solve Equation (4.1) in section 4.2.1. The definition of the dynamic programming algorithm will help us to define and study the properties of our proposed basic branch-and-bound algorithm for PB-SMT in section 4.2.2, including the necessary extensions that are required to perform hypotheses recombination. A monotone version of the previously presented algorithm is described in section 4.2.3. Stack pruning techniques and multiple-stack algorithms are introduced in section 4.2.4. Required modifications to obtain a breadth-first search algorithm are presented in section 4.2.5. Generalised multiple-stack algorithms for best- and breadth-first search are explained in sections 4.2.6 and 4.2.7 respectively. Additional pruning techniques to restrict the search space are described in section 4.2.8. Rest score estimation techniques are presented in section 4.2.9. Finally, the concept of word graph and how are they generated is described in section 4.2.10

4.2.1 Dynamic Programming Algorithm

The search problem in PB-SMT was formalised as a dynamic programming problem in [Zen07]. In this section we apply this formalisation to the search problem defined by Equation (4.1). As will be shown in the following sections, the formalisation of the search problem in PB-SMT as a dynamic programming problem will help us in defining our branch-and-bound search algorithm as well as in analysing its time complexity.

Dynamic Programming Equations

The search space can be represented by a directed acyclic graph in which the states represent partial hypotheses and the edges represent extensions of these partial hypotheses. Given a sentence pair (f_1^J, e_1^I), a complete translation is determined by a path of length K in the search graph, where K is the length of the bisegmentation. The k'th edge within a path adds the k'th target phrase to the partial translation and the $(k+1)$'th state represents the current partial translation along with a valid set of values of the bisegmentation variables, (a_1^K, b_1^K, c_1^K). Figure 4.1 shows an example of a path of length 3 in the search graph when translating the Spanish source sentence “la casa verde .” into English. Starting from the

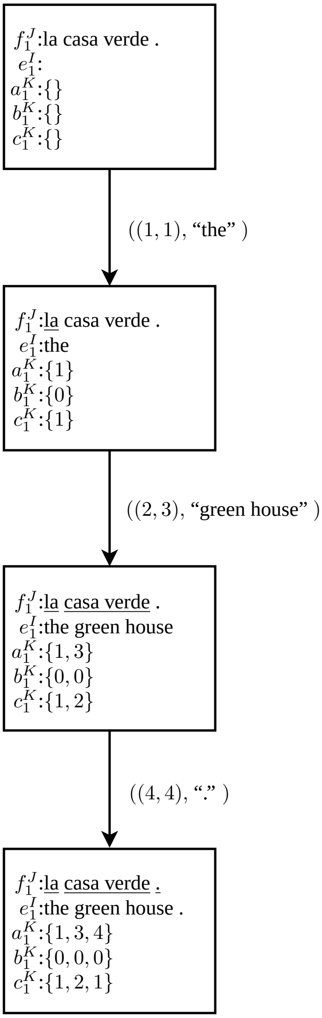


Figure 4.1: Example of a path in the search graph. The path determines a posible translation of the Spanish source sentence "la casa verde ." along with a valid set of values for the bisegmentation variables (a_1^K, b_1^K, c_1^K).

null hypothesis, a complete hypothesis is built by adding new phrase pairs to the hypothesis. Each arc of the path is labelled with a pair of elements, $((j,j'),\tilde{e})$, where (j,j') represents the boundaries of the source phrase to be aligned and $\tilde{e}$ represents the newly added target phrase. Each node of the path reflects the source sentence, f_1^J, with its aligned source phrases (underlined words of f_1^J), the partial translation, e_1^I, and the current values of the bisegmentation variables, (a_1^K, b_1^K, c_1^K). After three hypothesis extensions, the final translation "the green house ." is obtained.

Each possible hypothesis extension of a partial hypothesis will be assigned a probability given by the language and translation models. Among all possible paths of the search graph, we will be interested in that of the highest probability. As has been explained above, a state of the graph represents a partial hypothesis which contains detailed information about the bisegmentation process and the target words that compose the partial translation of the source sentence. It is worthy of note that only a subset of this information is relevant to assign a probability to further extensions of the partial hypothesis. The minimum information that is required to assign probabilities to hypothesis extensions is called state of the language and translation models. This minimum information may also include information required to guarantee that the generated hypotheses can be explained by the generative process of the statistical models. Two partial hypotheses sharing the same state of the language and translation models can be completed in the same way and thus we will only be interested in the hypothesis of higher probability. The state information for the language and translation models is determined by the specific modelling assumptions.

For the statistical models that appear in Equation (4.1), the state information is composed of the following elements:

- $\mathcal{SP}$: represents the set of currently unaligned positions of the source sentence. $\mathcal{SP}$ allows to check that the constraints of the bisegmentation are satisfied (all source words have to be aligned without overlaps). $\mathcal{SP}$ is also involved in the generation of phrase translation probabilities.

- m: represents the number of target words that compose the partial translation, e_1^m, of the source sentence.

- σ: represents the last $n-1$ target words that has been added to the partial translation, e_1^m, where n is the order of the n-gram language model. In other words, σ is the language model history of the current partial hypothesis.

- j: represents the rightmost source position of the last source phrase that has been aligned. j is required to appropriately generate distortion probabilities.

According to the previous considerations, the search graph defined above can be greatly simplified. Specifically, a state of the graph can be represented by a quadruple $(\mathcal{SP}, m, \sigma, j)$. We define the quantity $Q(\mathcal{SP}, m, \sigma, j)$ to denote the maximum probability of a path leading from the initial state to the state $(\mathcal{SP}, m, \sigma, j)$. In addition to this, we also define $\hat{Q}$ as the probability of the optimal translation. We obtain the following dynamic programming recursion equations:

$$Q(\{1, ..., J\}, 0, \mathsf{BOS}, 0) = 1 \tag{4.2}$$

$$\begin{aligned} Q(\mathcal{SP}, m, \sigma, j) = & \max_{\substack{j'', j' | j' \leq j \leq J \wedge \{j', j'+1, ..., j\} \cap \mathcal{SP} = \emptyset \\ m', \tilde{e}, \sigma' | m' + |\tilde{e}| = m \wedge \sigma = \mathtt{tail}(n-1, \sigma' \tilde{e})}} \\ & \Big\{ Q(\mathcal{SP} \cup \{j', j'+1, ..., j\}, m', \sigma', j'') \cdot \\ & \prod_{i=m'+1}^{m} p(e_i | e_{i-n+1}^{i-1}) \cdot \\ & p(m|m'+1) \cdot p(j' - j'') \cdot \\ & p(j - j' + 1 | m, m'+1) \cdot p(f_{j'}^{j} | \tilde{e}) \Big\} \end{aligned} \tag{4.3}$$

$$\hat{Q} = \max_{m, \sigma, j} \Big\{ Q(\emptyset, m, \sigma, j) \cdot p(\mathsf{EOS}|\sigma) \cdot p(J|m) \Big\} \tag{4.4}$$

where BOS denotes the begin of sentence marker, EOS denotes the end of sentence marker and $\mathtt{tail}(x,s)$ is a function that returns the last x words of the string s.

Implementation Details

The solution to the dynamic programming problem given by Equation (4.4) can be implemented in many ways. To avoid repeated computations we have to traverse the search graph in a topological order, that is, before we process a state we have to make sure that we have visited all predecessor states. Richard Zens [Zen07] proposed a breadth-first algorithm which we have adapted here to our maximisation problem. Algorithm 4.1 shows our proposed search algorithm. The algorithm works in a similar way as the dynamic programming recursion presented above does: for every given state, the probabilities of the extensions of the predecessor states arriving to the given state are computed, keeping the extension of highest probability. The first five loops of the algorithm (lines 3, 4, 5, 6 and 9) allow to iterate over the predecessor states $(\mathcal{SP} - \{j', ..., j'+l\}, \cdot, \cdot, \cdot)$ that arrive to successor states $(\mathcal{SP}, \cdot, \cdot, \cdot)$ by aligning the source positions $\{j', ..., j'+l\}$. The *forall* loop in line 10 iterates over the set of target phrases that are contained in the set $\mathcal{T}_{j', j'+l}$, where $\mathcal{T}_{j', j'+l}$ is composed of the target phrase translations for the source phrase f_j^{j+l} that are present in the phrase table. Given a fully determined predecessor state $(\mathcal{SP}', m', \sigma', j'')$, and a target phrase $\tilde{e} \in \mathcal{T}_{j', j'+l}$, we compute the probability p of the successor state determined by them, $(\mathcal{SP}, m' + |\tilde{e}|, tail(n-1, \sigma'\tilde{e}), j'+l)$. If the resulting probability p for the successor state is greater than the current best probability, we update it. In addition to this, we also update the variables $A(\cdot, \cdot, \cdot, \cdot)$ and $B(\cdot, \cdot, \cdot, \cdot)$, that for a given state represents the best target phrase arriving to it and the best predecessor state, respectively. Once the search process has been completed, these variables allow us to retrieve the target sentence of highest probability.

It is worth pointing out that the first and the second *for* loops of Algorithm 4.1 are introduced to ensure that the search graph is traversed in a topological order. Specifically, the first loop guarantees that the states corresponding to partial hypotheses with a lower number

```
    input    : $f_1^J$, $e_1^I$, $n$ (order of the $n$-gram language model),
               $p(e_1^I)$ ($n$-gram language model), $p(f_1^J|e_1^I)$ (translation model),
               $\mathcal{T}_{j_1,j_2}, \forall j_1, j_2 | j_1 \leq j_2 \wedge j_1 \geq 1 \wedge j_2 \leq J$ (set of translations for every
               phrase $f_{j_1}^{j_2}$ of $f_1^J$ in phrase table)
    output   : $Q(\mathcal{SP}, m, \sigma, j)$,
               $A(\mathcal{SP}, m, \sigma, j)$ (best target phrase arriving to state),
               $B(\mathcal{SP}, m, \sigma, j)$ (backpointer to the best predecessor state)
    auxiliar: tail ($x$,$s$) (returns last $x$ words of string $s$)
 1  begin
 2      $Q(\cdot,\cdot,\cdot,\cdot) := -\infty$; $Q(\{1,...,J\}, 0, \text{BOS}, 0) := 1$
 3      for $c = J - 1$ to $0$ do
            // $c$ is the cardinality of $\mathcal{SP}$
 4          for $l = 1$ to $J - c$ do
                // $l$ is the length of the source phrase
 5              forall $\mathcal{SP}' \subset \{1,...,J\} \wedge |\mathcal{SP}'| = c + l$ do
 6                  forall $j' \in \{1,...,J\} \wedge \{j', j'+1,...,j'+l-1\} \subseteq \mathcal{SP}'$ do
 7                      $j := j' + l - 1$
 8                      $\mathcal{SP} := \mathcal{SP}' - \{j', j'+1,...,j\}$
 9                      forall $(m', \sigma', j'') \in Q(\mathcal{SP}', \cdot, \cdot, \cdot)$ do
10                          forall $\tilde{e} \in \mathcal{T}_{j',j}$ do
11                              $m := |\tilde{e}| + m'$
12                              $\sigma :=$tail$(n-1, \sigma'\tilde{e})$
13                              $p := Q(\mathcal{SP}', m', \sigma', j'')\cdot$
14                                  $\prod_{i=m'+1}^{m} p(e_i|e_{i-n+1}^{i-1})\cdot$
15                                  $p(m|m'+1) \cdot p(j'-j'')\cdot$
16                                  $p(j-j'+1|m, m'+1) \cdot p(f_{j'}^{j}|\tilde{e})$
17                              if $p > Q(\mathcal{SP}, m, \sigma, j)$ then
18                                  $Q(\mathcal{SP}, m, \sigma, j) := p$
19                                  $A(\mathcal{SP}, m, \sigma, j) := \tilde{e}$
20                                  $B(\mathcal{SP}, m, \sigma, j) := (\mathcal{SP}', m', \sigma', j'')$
21  end
```

Algorithm 4.1: Pseudocode for the `dp_search` algorithm.

of aligned source positions are visited first. Regarding the second loop, it guarantees that the hypothesis extensions that align a lower number of source positions are visited first.

Regarding the computational complexity of Algorithm 4.1, the first four loops (lines 3, 4, 5 and 6) have a complexity in $\mathcal{O}(2^J \cdot J^2)$, where the exponential term 2^J comes from the number of possible subsets of J unaligned source words. The loop in line 9 has a complexity in $\mathcal{O}(M \cdot E_{n-1} \cdot J)$ where M is the highest target sentence length that can be obtained using phrase translations contained in the phrase table and E_{n-1} is the maximum number of target language model histories. Finally, the loop in line 10 has a complexity in $\mathcal{O}(T)$, where T is the maximum number of phrase translations for a source phrase. In summary, the computational complexity of the algorithm is in $\mathcal{O}(2^J \cdot J^3 \cdot M \cdot E_{n-1} \cdot T)$.

Hence, the proposed dynamic programming algorithm presents an exponential complexity. This exponential complexity can be avoided by introducing restrictions in the search space. One example of these restrictions is not to allow reorderings of the target phrases during the search process. This kind of search is known as *monotonic search* and will be explained in the following section.

Monotonic Search

The dynamic programming search algorithm proposed above can be easily modified to perform monotonic search. The search is monotonic if no reorderings of the phrase translations are allowed. To avoid reorderings we have to ensure that, for a given partial hypothesis, there are no unaligned source positions between two aligned source positions.

Algorithm 4.1 requires little modifications to perform monotonic search. Specifically, the loop in line 5 is replaced by the assignment $\mathcal{SP}' = \{1, 2, ..., c + l\}$; and the loop in line 6 is replaced by the assignment $j' = c + 1$ (this ensures that there are no unaligned source positions between two aligned ones). Monotonic search allows to significantly reduce the computational complexity of the translation process. As was explained above, the complexity of the non-monotonic search algorithm can be decomposed into three terms: the first four loops in Algorithm 4.1 contribute with the term $(2^J \cdot J^2)$, the loop in line 9 contributes with the term $(M \cdot E_{n-1} \cdot J)$ and the loop in line 10 contributes with the term T. If monotonic search is performed, the first term is now (J^2), since the third and the fourth loops has been replaced by assignments. The second term is now $(M \cdot E_{n-1})$, since j'' in line 9 can take only one possible value. Finally, the third complexity term remains unchanged. Therefore, the final complexity of the monotonic dynamic programming algorithm is in $\mathcal{O}(J^2 \cdot M \cdot E_{n-1} \cdot T)$.

It is worth noticing that, when performing monotonic search, the probability of the distortion submodel $p(b_k)$ for each possible extension is 1, since b_k is equal to 1 in all cases.

4.2.2 Branch-and-Bound Search for PB-SMT

The branch-and-bound search algorithm that we propose is based on the well-known $A^\star$ search algorithm. The basic $A^\star$ search algorithm [HNR68] (or sometimes, stack decoder) is an iterative algorithm that can be described as follows:

1. initialise the stack with the null hypothesis
2. remove the hypothesis with the highest score from the stack
3. if this hypothesis is a goal hypothesis, output this hypothesis and terminate
4. produce all extensions of this hypothesis and push the extensions into the stack
5. goto step 2

One key aspect of the $A^\star$ search algorithm is the use of a stack data structure to organise the search space. The term stack does not imply here a last-in, first-out (LIFO) container. Instead, this stack data structure stores the hypotheses in ascending order of their scores (it is actually a priority queue), allowing the best hypothesis generated so far to be extended at each iteration.

$A^\star$ search uses an additive scoring function of the form $q(h) = f(h) + r(h)$, where $f(h)$ is the score to arrive from the null hypothesis to hypothesis h, and $r(h)$ gives a heuristic estimation of the rest score from h to a complete hypothesis. This heuristic estimation function is called *admissible* if it never underestimates this score. If the heuristic estimation of the rest score is admissible, then the $A^\star$ search algorithm is optimal (see [HNR68] for more details).

The $A^\star$ search algorithm was introduced in the field of speech recognition by Jelinek et al. [JMB75], and later imported for its use in SMT with single-word translation models in [BBD$^+$96, WW97, GJK$^+$01, OUN01]. As far as we know, the application of $A^\star$ search in PB-SMT has not been extensively addressed. Tomás and Casacuberta [TC01] proposed an $A^\star$-based search algorithm, but their algorithm does not take advantage of dynamic programming techniques to efficiently implement the search process.

Search Algorithm

We propose to solve the search problem given by Equation 4.4 following the $A^\star$ search procedure described above. Before introducing the search algorithm, we will explain the specific representation for the partial hypotheses that we have adopted. A hypothesis h is represented as a vector of elements containing information about each hypothesis extension. Each individual element consists in a pair of source positions determining the aligned source phrase plus the target phrase that is chosen as translation of the aligned source phrase. Thus, according to our proposed representation, a hypothesis has the form: $h \equiv [((j,j'),\tilde{e}),((j'',j'''),\tilde{e}'),...]$, where the target sentence is generated from left to right by concatenating the target phrases $(\tilde{e}\tilde{e}'...)$ and there are no overlaps in the aligned source positions (the search path example given in Figure 4.1 shows how a complete hypothesis is built by adding new elements to the above mentioned vector of hypothesis extensions; in the figure, the newly added elements are the labels of the arcs).

Algorithm 4.2 shows the pseudocode of our branch-and-bound search algorithm for PB-SMT. The proposed algorithm works by expanding hypotheses until a complete hypothesis is found. A hypothesis is complete if it has not unaligned source positions, that is, if $\mathcal{SP}_h = \emptyset$, where $\mathcal{SP}_h$ is the set of unaligned source positions for the hypothesis h. The `obtain_trg_sent` function returns the partial translation associated to a given hypothesis and the `back` function returns the last element of the vector that is used to represent the hypotheses. The top of the stack is extracted by means of the `pop` function and hypotheses are inserted into the stack using the `push` function. If the hypothesis in the top of the stack, h, is not a complete hypothesis, it is expanded by means of the `expand` function which is described below. The results of the expansion are stored in the set $\mathcal{H}$. Each hypothesis h' contained in $\mathcal{H}$ is assigned a score which is calculated incrementally from the score of the predecessor hypothesis h and the information about the last extension. It should be noted that, in the scoring function $q(h) = f(h) + r(h)$ that we have used, $f(h)$ is given by the logarithm of the probability of h ($A^\star$ search was developed for additive scoring functions), and $r(h)$ (the rest score function) is zero in all cases. The $r(h)$ function can be specifically defined to improve the performance of the search algorithm. We will return on this point in sections 4.2.5 and 4.2.9.

```
input    : $f_1^J$, $n$ (order of the $n$-gram language model),
           $p(e_1^I)$ ($n$-gram language model), $p(f_1^J|e_1^I)$ (translation model),
           $\mathcal{T}_{j_1,j_2}, \forall j_1, j_2 | j_1 \leq j_2 \wedge j_1 \geq 1 \wedge j_2 \leq J$ (set of translations for every
           phrase $f_{j_1}^{j_2}$ of $f_1^J$ in phrase table)
output   : $\hat{e}_1^{\hat{I}}$ (optimal translation)
auxiliar : s (stack),
           $\mathcal{SP}_h$ (set of unaligned source word positions of hypothesis $h$),
           $\mathcal{H}$ (set of expanded hypotheses)
begin
    $h_\emptyset := [\,((0,0), \mathsf{BOS})]$
    push(s,$0$,$h_\emptyset$)
    end :=false
    while !end do
        $(q,h)$ :=pop (s)
        if $\mathcal{SP}_h = \emptyset$ then
            $\hat{e}_1^{\hat{I}}$:=obtain_trg_sent ($h$)
            end :=true
        else
            $\mathcal{H}$ =expand ($h$,$\mathcal{T}_{j_1,j_2}$)
            $e_1^{m'}$ :=obtain_trg_sent ($h$)
            $m' := |e_1^{m'}|$
            $\sigma'$ :=tail ($n-1$,BOS$e_1^{m'}$)
            $((\cdot, j''), \cdot)$ :=back ($h$)
            forall $h' \in \mathcal{H}$ do
                $((j', j), \tilde{e})$ :=back ($h'$)
                $m := |\tilde{e}| + m'$
                $p := \prod_{i=m'+1}^{m} p(e_i|e_{i-n+1}^{i-1})\cdot$
                      $p(m|m'+1) \cdot p(j' - j'')\cdot$
                      $p(j - j' + 1|m, m'+1) \cdot p(f_{j'}^{j}|\tilde{e})$
                if $\mathcal{SP}_{h'} = \emptyset$ then
                    $p := p \cdot p(\mathsf{EOS}|\mathtt{tail}(n-1, \sigma'\tilde{e}))\cdot p(J|m)$
                $q' := q + \log p$
                push (s,$q'$,$h'$)
end
```

Algorithm 4.2: Pseudocode for the bb_search algorithm.

The expansion algorithm works by aligning source phrase positions, (j_1, j_2), from the set $PP(\mathcal{SP}_h)$, where the function $PP(\cdot)$, given a set of word positions returns the set of all possible phrase positions that can be obtained using these word positions. For instance, $PP(\{1,3,4,5\})$ would contain the phrase positions: $\{(1,1),(3,3),(4,4),(5,5),(3,4),(4,5),(3,5)\}$. Given a set of word positions, $\mathcal{SP}$,

```
input    : $h$ (hypothesis to be expanded)
           $\mathcal{T}_{j_1,j_2}, \forall j_1, j_2 | j_1 \leq j_2 \wedge j_1 \geq 1 \wedge j_2 \leq J$ (set of translations for every
           phrase $f_{j_1}^{j_2}$ of $f_1^J$ in phrase table)
output   : $\mathcal{H}$ (set of expanded hypotheses)
auxiliar : $\mathcal{SP}_h$ (set of unaligned source positions of $h$),
           $PP(\mathcal{SP}_h)$ (set of all possible unaligned source phrase positions of $h$)
begin
    forall $(j_1, j_2) \in PP(\mathcal{SP}_h)$ do
        forall $\tilde{e} \in \mathcal{T}_{j_1,j_2}$ do
            $\mathcal{H} := \mathcal{H} \cup \{\texttt{append}(h, ((j_1, j_2), \tilde{e}))\}$
end
```

Algorithm 4.3: Pseudocode for the `expand` algorithm.

$PP(\mathcal{SP})$ can be formally defined as follows:

$$PP(\mathcal{SP}) = \{(j, j') \mid \{j, ..., j'\} \in \mathcal{SP} \wedge j \leq j'\} \tag{4.5}$$

Algorithm 4.3 shows the pseudocode of the expansion algorithm. For each unaligned source phrase position $(j_1, j_2) \in PP(\mathcal{SP}_h)$, the expansion algorithm generates new expanded hypotheses by adding new elements $((j_1, j_2), \tilde{e})$ to the vector representing the hypotheses to be expanded. Given (j_1, j_2), the target phrases $\tilde{e}$ are extracted from $\mathcal{T}_{j_1,j_2}$, which represents the set of translations for $f_{j_1}^{j_2}$ in phrase table. The new elements are added by means of the `append` function, which given a vector representing a hypothesis and a new element, simply appends the new element at the end of the vector.

Hypothesis Recombination

As was explained in section 4.2.1, the search space can be represented as a directed acyclic graph in which the states represent partial hypothesis and the edges represent extensions of these partial hypotheses. This search space can be greatly simplified by taking into account that two partial hypotheses are equivalent if they share the same state information for the language and the translation models. The `bb_search` algorithm described above does not take advantage of these considerations and thus carries out unnecessary calculations. To solve this problem, it is crucial to perform *recombination of search hypotheses* [OUN01]: every two partial hypotheses that share the same state information of the language and the translation models can be recombined, keeping only the hypothesis with the highest score.

To efficiently implement hypothesis recombination, we replace the `push` function in Algorithm 4.2 by the `push_rec` function. Hypothesis recombination requires the introduction of three new data structures, namely, a set of hypothesis states ($\mathcal{ST}$), a table that stores the highest score for each hypothesis state ($HS_{\forall(\mathcal{SP},m,\sigma,j)}$) and another table that for each hypothesis state stores a pointer to the position of the hypothesis having this state information in the stack ($PT_{\forall(\mathcal{SP},m,\sigma,j)}$). Algorithm 4.4 shows the pseudocode for the `push_rec` function. The `push_rec` function, given a hypothesis h, first obtains its state information, $(\mathcal{SP}_h, m, \sigma, j)$. If $(\mathcal{SP}_h, m, \sigma, j)$ is contained in the set $\mathcal{ST}$, h is only inserted in the stack

```
input    : s (stack),
           $q$ (score of the hypothesis to be pushed into the stack),
           $h$ (hypothesis to be pushed into the stack),
           $\mathcal{ST}$ (set of hypothesis states),
           $HS_{\forall(\mathcal{SP},m,\sigma,j)}$ (Highest score for states),
           $PT_{\forall(\mathcal{SP},m,\sigma,j)}$ (Pointer to stack position for states)
output   : s,$\mathcal{ST}$,$HS_{\forall(\mathcal{SP},m,\sigma,j)}$,$PT_{\forall(\mathcal{SP},m,\sigma,j)}$ (updated variables)
auxiliar: $\mathcal{SP}_h$ (set of unaligned source positions of $h$)
begin
    $e_1^m$ :=obtain_trg_sent ($h$)
    $m := |e_1^m|$
    $\sigma$ :=tail ($n-1$,BOS$e_1^m$)
    $((\cdot, j), \cdot)$ :=back ($h$)
    if $(\mathcal{SP}_h, m, \sigma, j) \in \mathcal{ST}$ then
        if $q > HS_{(\mathcal{SP}_h,m,\sigma,j)}$ then
            remove(s,$PT_{(\mathcal{SP}_h,m,\sigma,j)}$)
            $HS_{(\mathcal{SP}_h,m,\sigma,j)} := q$
            $PT_{(\mathcal{SP}_h,m,\sigma,j)}$ :=push(s,$q$,$h$)
    else
        $HS_{(\mathcal{SP}_h,m,\sigma,j)} := q$
        $PT_{(\mathcal{SP}_h,m,\sigma,j)}$ :=push(s,$q$,$h$)
        $\mathcal{ST} := \mathcal{ST} \cup (\mathcal{SP}_h, m, \sigma, j)$
end
```

Algorithm 4.4: Pseudocode for the push_rec algorithm.

if its probability is higher than the highest score that has been seen so far: $HS_{(\mathcal{SP}_h,m,\sigma,j)}$. The remove function is used to remove the recombined hypothesis from the stack. The stack pointer, $PT_{(\mathcal{SP}_h,m,\sigma,j)}$, is used to increase the speed of the remove function (we assume that the conventional push function returns a pointer to the hypothesis inserted into the stack). If $(\mathcal{SP}_h, m, \sigma, j)$ is not contained in $\mathcal{ST}$, then no hypothesis recombination is required.

Our proposed branch-and-bound algorithm with hypothesis recombination explores the same search space as the dynamic programming algorithm given in section 4.2.1. The main difference between these two search algorithms is that the branch-and-bound algorithm uses best-first search to obtain the solution while the dynamic programming algorithm uses breadth-first search. Therefore, the complexity of the branch-and-bound algorithm with hypothesis recombination is bounded by the complexity of the dynamic programming algorithm: $\mathcal{O}(2^J \cdot J^3 \cdot M \cdot E_{n-1} \cdot T)$.

4.2.3 Monotonic Search

The branch-and-bound search algorithm proposed in the previous section can be easily modified to perform monotonic search. As was explained in section 4.2.1 for dynamic programming search, the search is monotonic if no reorderings of the phrase translations are

```
input  : $h$ (hypothesis to be expanded)
         $\mathcal{T}_{j_1,j_2}, \forall j_1, j_2 | j_1 \leq j_2 \wedge j_1 \geq 1 \wedge j_2 \leq J$ (set of translations for every
         phrase $f_{j_1}^{j_2}$ of $f_1^J$ in phrase table)
output : $\mathcal{H}$ (set of expanded hypotheses)
begin
    $((\cdot, j), \cdot) :=$ back $(h)$
    for $j' = j + 1$ to $J$ do
        forall $\tilde{e} \in \mathcal{T}_{j+1,j'}$ do
            $\mathcal{H} := \mathcal{H} \cup \{$append$(h, ((j+1, j'), \tilde{e}))\}$
end
```

Algorithm 4.5: Pseudocode for the mon_expand algorithm.

allowed. To avoid reorderings, the expand algorithm given by Algorithm 4.3 is replaced by the mon_expand algorithm given by Algorithm 4.5. The mon_expand algorithm does not allow unaligned source positions between two aligned source positions.

The complexity of the monotonic branch-and-bound algorithm with hypothesis recombination is bounded by the complexity of the monotonic dynamic programming algorithm discussed in section 4.2.1: $\mathcal{O}(J^2 \cdot M \cdot E_{n-1} \cdot T)$.

4.2.4 Stack Pruning and Multiple Stacks

The branch-and-bound search algorithm presented above guarantees that the optimal solution is obtained. Due to the high complexity of the search, we cannot expect to efficiently obtain this optimal solution. The stack used by our proposed branch-and-bound search algorithm can be pruned to reduce the computational complexity. Typically, a limitation in the maximum number of hypotheses that can be stored into the stack is imposed. If this maximum number of hypotheses is exceeded, then the stack is pruned.

It should be noted that for a given hypothesis, the more aligned source words, the lower the probability. This constitutes a problem when stack search is applied, since those hypotheses with a higher number of aligned source words will be pruned sooner due to the stack length limitation. One possible solution to this problem consists in introducing the use of multiple stacks instead of only one. The key idea of multiple-stack search algorithms consists in assigning hypotheses to stacks such that there is “fair competition” within each stack, i.e., hypotheses stored in the same stack should align roughly the same number of source words (and the same words) if possible.

Multiple-stack search algorithms for single-word translation models store those hypotheses with different subsets of aligned source words in different stacks [Ger01]. That is to say, given an input sentence f_1^J composed of J words, multiple-stack search algorithms employs 2^J stacks to translate it. Such an organisation improves the pruning of the hypotheses when the stack length limitation is exceeded, since only those hypotheses with the same aligned source positions can compete with each other. However, these multiple-stack algorithms have the negative property of spending significant amounts of time in selecting the hypotheses to be expanded, since at each iteration, the best hypothesis in a set of 2^J stacks must be searched for [OGVC03].

Additionally, certain PB-SMT systems, e.g. the Moses decoder [KHB+07], use an alternative approach which consists in assigning to the same stack, those hypotheses with the same number of aligned source words (the Moses decoder is a dynamic programming-based translation system with beam search, but it also uses stacks to organise the search space).

The required modifications in Algorithm 4.2 to perform multiple stack search include the following:

- The single stack `s` used by the algorithm is replaced by a collection of stacks, $\texttt{s}_i$, that are created on demand.
- The `push` function call no longer receives the stack `s` as input parameter. Instead, it receives $\texttt{s}_i$. Given a hypothesis h, the stack into which h is stored is given by a mapping function $\mu(h)$. The mapping function $\mu(h)$ takes the hypothesis h as input and returns the identifier i of the stack into which h is to be inserted.
- The `pop` function now has to find the hypothesis of highest probability at each iteration among all stacks $\texttt{s}_i$.

The key aspect of a multiple-stack search algorithm is the way in which the mapping function $\mu(h)$ is defined. The mapping function determines a set of equivalence classes for the partial hypotheses. We note this set of equivalence classes as $\mu[\mathcal{H}]$, where $\mathcal{H}$ represents the space of partial hypotheses. The number of stacks used by the multiple-stack search algorithm is given by $\big|\mu[\mathcal{H}]\big|$.

4.2.5 Breadth-First Search

Let us suppose that we are using a multiple-stack search algorithm with maximum stack length equal to L_s. The stack length limitation guarantees that a stack will never contain more than L_s hypotheses. However, during the execution of the algorithm, the number of hypotheses that can be chosen for expansion from a given stack can be greater than L_s. This is due to the best-first nature of the search: after extracting one or many hypotheses from a given stack, new hypotheses can be inserted into the same stack in subsequent iterations of the search algorithm.

The pruning due to the stack length limitation can be more aggressive if we force the search algorithm to expand first those hypotheses with a lower number of aligned source positions. That is, if the search algorithm performs a breadth-first exploration of the search space. Under these circumstances, it is guaranteed that the search algorithm will only expand at most L_s hypotheses from each stack, since after extracting the first hypothesis from a given stack, no new hypotheses will be inserted into it.

Our proposed multiple-stack search algorithm can perform a breadth-first search by only modifying its scoring function: $q(h) = f(h) + r(h)$. Specifically, the algorithm will perform a breadth-first search if the following condition is satisfied:

$$f(h) + r(h) > f(h') + r(h') \ \forall h, h' \mid |\mathcal{SP}_h| = |\mathcal{SP}_{h'}| + 1$$

One way to ensure that the previous condition holds is to fix an appropriate value for the rest score estimation function $r(h)$, which has to verify the following inequality:

$$r(h) > f(h') + r(h') - f(h) \ \forall h, h' \mid |\mathcal{SP}_h| = |\mathcal{SP}_{h'}| + 1$$

$f(h')$ and $f(h)$ can be bounded by the highest and the lowest log-probability, respectively, that can be assigned to a hypothesis: $f(h') \leq 0$ and $f(h) > \log \epsilon$, where ϵ is a very small positive number. Taking into account the previous considerations, a trivial recurrence relation can be obtained. From this recurrence relation, one valid definition of $r(h)$ to obtain a breadth-first search algorithm is:

$$r(h) = |\mathcal{SP}_h| \cdot (-\log \epsilon) \tag{4.6}$$

It is worth of notice that the previous definition of $r(h)$ constitutes an admissible heuristic since it never underestimates the rest score of a hypothesis h.

If we perform breadth-first search, the pop function executed at each iteration of the search no longer needs to explore the whole set of stacks to obtain the best hypotheses. Instead, the pop function returns the top of the stack containing the hypotheses that are nearest to the null hypothesis.

To calculate the complexity of our proposed breadth-first multiple-stack algorithm, we need to calculate the complexity of the expansion algorithm. The complexity of the expansion depends on the number of unaligned phrases for the hypothesis to be expanded: $|PP(\mathcal{SP})|$ (see Algorithm 4.3). In the worst case, where there are no aligned source positions, we have to align a total of $\frac{J^2+J}{2}$ phrases ($|PP(\{1, ..., J\})| = \sum_{j=1}^{J} j$). In addition to this, for each unaligned phrase position there are at most T phrase translations. Therefore, the complexity of the expansion algorithm is in $\mathcal{O}(J^2 \cdot T)$.

If the hypotheses are stored in J stacks, then the breadth-first search algorithm executes a total of $J \cdot L_s$ expansions, thus obtaining a complexity in $\mathcal{O}(J^3 \cdot L_s \cdot T)$. It should be noted that the resulting complexity is no longer exponential in J due to the combination of stack pruning techniques and breadth-first search. By contrast, the best-first search algorithm executes a less aggressive pruning of the stacks and therefore its complexity cannot be bounded by the complexity of the breadth-first algorithm. However, in certain cases, the time cost of best-first search may be lower than that of breadth-first search since best-first search complexity is closely related to the ability of the statistical models to guide the search. More specifically, the lower the perplexity of the statistical models involved in the translation process, the lower the time cost of the search algorithm. This will be empirically demonstrated in section 5.2.

4.2.6 Generalised Multiple-Stack Algorithm for Best-First Search

The stack search algorithms for SMT described in the literature typically use 1, J or 2^J stacks to perform the search. As was explained in section 4.2.4, the use of multiple stacks allows to improve the stack pruning efficiency but increases the computational cost of the pop function. By contrast, the single-stack search algorithm executes an efficient pop function but the stack length limitation tends to prune those hypotheses with a higher number of aligned source positions.

Here we propose a possible way to make a tradeoff between the advantages of these algorithms by introducing a new parameter which will be referred to as the *granularity* of the algorithm. The granularity parameter determines the number of stacks that are used during the decoding process. The generalised multiple-stack algorithm described in this section is appropriate when we perform a best-first search.

Selecting the Granularity of the Algorithm

To appropriately define the concept of granularity we first have to define the concept of *alignment vector* for a hypothesis h. The alignment vector, $\mathbf{v}$, for a hypothesis h is a binary vector of J bits: $(\{0,1\})^J$, where the j'th bit is set to 1 if $j \notin \mathcal{SP}_h$. In addition to this, we also define the L_a parameter, which represents the maximum number of hypotheses than can be stored by the search algorithm (this differs from the previously defined L_s parameter, which imposes a limitation on the maximum number of hypotheses that can be stored into a given stack).

The granularity (G) of a generalised multiple-stack algorithm is an integer which takes values between 1 and J. Given a sentence f_1^J to be translated, a generalised stack algorithm with a granularity parameter equal to g, will have the following features:

- The algorithm will use at most 2^g stacks to perform the translation.
- Each stack will contain hypotheses which may have 2^{J-g} different alignment vectors.
- If the algorithm can store at most L_a hypotheses, then, the maximum size of each stack will be equal to $\frac{L_a}{2^g}$.

Mapping Hypotheses to Stacks

The key aspect of a multiple-stack search algorithm is the way in which hypotheses are mapped to stacks. Here we define a mapping function based on the alignment vector, $\mathbf{v}$, for a given hypothesis. Given a alignment vector composed of J bits, the mapping function, $\mu(\mathbf{v})$, returns a stack identifier composed of only g bits:

$$\mu : (\{0,1\})^J \longrightarrow (\{0,1\})^g \tag{4.7}$$

The mapping function can be defined in many ways, but there are two essential principles which must be taken into account:

- The mapping function must be efficiently calculated.
- Hypotheses whose alignment vector have a similar number of bits set to one must be assigned to the same stack.

A possible way to implement the mapping function, namely $\mu_1(\mathbf{v})$, consists in simply shifting the alignment vector $J-g$ positions to the right, and then keeping only the first g bits. Such a proposal is very easy to calculate, however, it has a poor performance according to the second principle explained above.

A better alternative to implement the mapping function, namely $\mu_2(\mathbf{v})$, can be formulated as a composition of two functions. A constructive definition of such an implementation is detailed next:

1. Given f_1^J, we order the set of J bit numbers as follows: first the numbers which do not have any bit equal to one, next, the numbers which have only one bit equal to one, and so on.

2. Given the list of numbers described above, we define a function which associates to each number of the list, the order of the number within this list.

3. Given the alignment vector of a partial hypothesis, $\mathbf{v}$, the stack on which this partial hypothesis is to be inserted is obtained by a two step process: First, we obtain the image of $\mathbf{v}$ returned by the function described above. Next, the result is shifted $J - g$ positions to the right, keeping the first g bits.

Let β be the function that shifts a bit vector $J - g$ positions to the right, keeping the first g bits; and let α be the function that for each alignment vector returns its order:

$$\alpha : (\{0,1\})^J \longrightarrow (\{0,1\})^J \tag{4.8}$$

Then, $\mu_2(\mathbf{v})$ is expressed as follows:

$$\mu_2(\mathbf{v}) = \beta \circ \alpha(\mathbf{v}) \tag{4.9}$$

Table 4.1 shows an example of the values which returns the $\mu_1(\mathbf{v})$ and the $\mu_2(\mathbf{v})$ functions when the input sentence has 4 words and the granularity of the decoder is equal to 2. As it can be observed, $\mu_2(\mathbf{v})$ function performs better than $\mu_1(\mathbf{v})$ function according to the second principle described at the beginning of this section.

Table 4.1: Values returned by the μ_1 and μ_2 function defined as a composition of the α and β functions.

$\mathbf{v}$	$\mu_1(\mathbf{v})$	$\alpha(\mathbf{v})$	$\mu_2(\mathbf{v})$
0000	00	0000	00
0001	00	0001	00
0010	00	0010	00
0100	01	0011	00
1000	10	0100	01
0011	00	0101	01
0101	01	0110	01
0110	01	0111	01
1001	10	1000	10
1010	10	1001	10
1100	11	1010	10
0111	01	1011	10
1011	10	1100	11
1101	11	1101	11
1110	11	1110	11
1111	11	1111	11

Single and Multiple-Stack Algorithms

The stack search algorithms using 1 and 2^J stacks can be instantiated as particular cases of the general formalism that has been proposed. Specifically, given the input sentence, f_1^J , a generalised stack decoding algorithm with $G = 0$ will have the same features as the single-stack search algorithm. By contrast, if $G = J$, the resulting search algorithm will have the same features as the multiple-stack search algorithm with 2^J stacks. Values of G between these extrema define a new family of multiple-stack search algorithms that allow to make a tradeoff between the stack pruning efficiency and the computational cost of the `pop` function.

4.2.7 Generalised Multiple-Stack Algorithm for Breadth-First Search

The use of multiple stacks in breadth-first search can also be generalised as well as it has been shown in the previous section for best-first search. The mapping functions used in breadth-first search have an additional requirement with respect to those used in best-first search. Specifically, those hypotheses with a different number of aligned source positions have to be assigned to different stacks. The number of stacks is again determined by the number of equivalence classes. Since in this case, the computational cost of the `pop` function is not affected by the number of stacks (see section 4.2.5), the number of equivalence classes for the hypotheses can be arbitrarily high.

The most basic multiple-stack algorithm for breadth-first search uses J stacks, one for each possible number of aligned source positions. The required mapping function is:

$$\mu(h) = J - \mathcal{SP}_h \tag{4.10}$$

Starting from the basic multiple-stack search algorithm defined above, we can propose new search algorithms by refining the partition determined by the mapping function. For instance, we can assign hypotheses with non-monotonic alignments to different stacks:

$$\mu(h) = (J - \mathcal{SP}_h, \texttt{is\_mon}(h)) \tag{4.11}$$

where the $\texttt{is\_mon}(h)$ predicate is evaluated to $true$ if h contains non-monotonic alignments. This mapping function can be useful when performing non-monotonic search.

Some additional examples of mapping functions are the following:

$$\mu(h) = (J - \mathcal{SP}_h, |\texttt{trg\_sent}(h)|) \tag{4.12}$$

where $|\texttt{trg\_sent}(h)|$ returns the number of target words that compose the partial hypothesis.

The history of the n-gram language model (or a part of it) can be used to define an alternative mapping function:

$$\mu(h) = (J - \mathcal{SP}_h, \texttt{tail}(1, \texttt{trg\_sent}(h))) \tag{4.13}$$

where $\texttt{tail}(1, \texttt{trg\_sent}(h))$ is the last target word added to the partial hypothesis.

Finally, we propose another example of mapping function that can be useful when performing non-monotonic search:

$$\mu(h) = (J - \mathcal{SP}_h, \texttt{last\_alig\_src\_pos}(h)) \tag{4.14}$$

where $\texttt{last\_alig\_src\_pos}(h)$ returns the index of the last source position that was aligned.

The maximum number of partitions is reached when the mapping function returns the state of the language and the translation models for a given hypothesis. Under these circumstances, the search algorithm obtains the optimal solution.

The computational complexity of the search depends on the number of stacks, which is given by the number of equivalence classes: $\big|\mu[\mathcal{H}]\big|$. Specifically, the complexity is in $\mathcal{O}(\big|\mu[\mathcal{H}]\big| \cdot J^2 \cdot L_s \cdot T)$. It is worthy of note that the mapping function can be seen as a possible way to prune the search space.

It is interesting to consider the search algorithms that are obtained when the maximum stack length, L_s, is set to 1. Depending on the mapping function, we can range from an algorithm with J stacks and complexity in $\mathcal{O}(J^3 \cdot T)$, to an optimal algorithm with one stack per each state of the language and the translation models and complexity in $\mathcal{O}(2^J \cdot J^3 \cdot M \cdot E_{n-1} \cdot T)$. In addition to this, this family of algorithms can replace the stack data structure by a variable storing a single hypothesis.

4.2.8 Additional Pruning Techniques

The only pruning technique that has been introduced so far is the stack length limitation. In addition to this, we can also apply the following set of pruning techniques:

- **Maximum source phrase length** (L_p): during the expansion of a hypothesis, the source phrases to be aligned, f_{j1}^{j2}, cannot exceed a certain length.

- **Maximum number of target phrase translations** (T_t): the expansion process works by aligning source phrases f_{j1}^{j2} with target phrases extracted from the set $\mathcal{T}_{j_1,j_2}$ of phrase translations for f_{j1}^{j2} contained in phrase table. One possible way to restrict the search consists in considering only a subset of the best target phrase translations contained in $\mathcal{T}_{j_1,j_2}$ as candidates to extend a given hypothesis.

- **Maximum number of skipped source positions** (S_s): during the expansion of a hypothesis, the number of source positions that can be skipped with respect to the rightmost position of the last aligned source phrase is restricted. In other words, we set a maximum value for the hidden variable b_k used in the specific phrase-based model derivation presented in section 3.5. If the maximum number of skipped source positions is set to zero, then we obtain a monotonic search algorithm. These reordering constraints are known as the IBM constraints [BBD$^+$96] and they were originally applied in SMT systems based on single-word translation models.

The pruning techniques that have been explained above can be straightforwardly introduced into our proposed search algorithm. Specifically, only the expansion algorithm has to be appropriately modified.

4.2.9 Rest Score Estimation

The efficiency of our proposed branch-and-bound search algorithm can be improved by defining appropriate rest score estimation functions. The definition of rest score estimation functions in SMT has been previously studied in [WW97, OUN01] for single-word translation models and later extended to the alignment template approach [ON02] and to phrase-based models [Koe03, Zen07]. Here we define rest score estimation functions for two different translation submodels, namely, the phrase translation submodel and the distortion submodel.

Regarding the phrase translation submodel, we follow the technique proposed in [Koe03], which is based on the maximum probability for translating source positions $\{j, ..., j'\}$:

$$p^*(j, j') = \max\left\{\max_{\tilde{e}\in\mathcal{T}_{j,j'}} p(f_j^{j'}|\tilde{e}),\ \max_{j\leq k<j'}\{p^*(j,k)\cdot p^*(k+1,j')\}\right\} \tag{4.15}$$

where $p^*(j, j')$ has to be calculated for the set of phrase positions contained in $LP(\mathcal{SP})$. Given a set of word positions, the function $LP(\cdot)$ returns the set of phrase positions determining the longest phrases that can be obtained using these word positions. For instance, $LP(\{1, 3, 4, 5\})$ would contain the following phrase positions: $\{(1, 1),(3, 5)\}$. $LP(\cdot)$ can be formally defined as follows:

$$\begin{aligned} LP(\mathcal{SP}) \quad = \quad & \{(j,j') \mid \{j,...,j'\} \in \mathcal{SP} \wedge j \leq j' \\ & \wedge\neg\exists\{j'',...,j'''\} \in \mathcal{SP}\ :\ (j,j') \neq (j'',j''') \wedge j'' \leq j \wedge j''' \geq j'\} \end{aligned} \tag{4.16}$$

The rest score estimation function for the phrase translation submodel is given by:

$$r_p(\mathcal{SP}) = \sum_{(j,j')\in LP(\mathcal{SP})} \log p^*(j, ..., j') \tag{4.17}$$

Finally, we have defined a very simple rest score estimation function for the distortion submodel. Specifically, given the set of unaligned source positions of the partial hypothesis, $\mathcal{SP}$, and the rightmost position of the last aligned source phrase, j, the rest score estimation is given by the number of skipped source positions with respect to j of highest probability according to the distortion submodel $p(\cdot)$:

$$r_d(\mathcal{SP}, j) = \max_{j'\in\mathcal{SP}} \log p(j' - j) \tag{4.18}$$

The overall rest score estimation function is obtained as the sum of the two rest score estimation functions defined above:

$$r(\mathcal{SP}, j) = r_p(\mathcal{SP}) + r_d(\mathcal{SP}, j) \tag{4.19}$$

It should be noted that the rest score estimation functions described above can be combined with that defined in section 4.2.5 to perform breadth-first search.

4.2.10 Generation of Word Graphs

Our branch-and-bound search algorithm with hypothesis recombination described above can be used to obtain the best translation according to the statistical models involved in the translation process. However, there are situations in which we are not only interested in this single best translation but also in alternative translations. The list of the N best translations for a given source sentence according to the statistical models receives the name of *N-best list*. These N-best lists can be obtained from a specific data structure called *word graph*.

A word graph is a weighted directed acyclic graph in which each node represents a partial translation hypothesis and each edge is labelled with a word (or group of words) of the target sentence and is weighted according to the scores given by an SMT model. Word graphs can be easily generated as a by-product of the translation process. The generation of word-graphs for SMT is described in [UON02] for the single-word based IBM 4 Model, and in [Koe03, HZN07] for phrase-based models.

Our branch-and-bound algorithm with hypothesis recombination can be easily modified to generate word graphs. We define a word graph as a set of quadruples of the form $((\mathcal{SP}, m, \sigma, j), (\mathcal{SP}', m', \sigma', j'), \tilde{e}, q)$; where each quadruple represents an edge from state $(\mathcal{SP}, m, \sigma, j)$ to state $(\mathcal{SP}', m', \sigma', j')$. This edge is labelled with the words $\tilde{e}$ and has the score q. After the execution of the expansion algorithm for a given hypothesis h, the word graph is extended with a new quadruple for each hypothesis h' contained in the set $\mathcal{H}$ of expanded hypotheses. Specifically, the new quadruple is composed of the state information for h and h', the newly added target phrase $\tilde{e}$ (where $((\cdot,\cdot),\tilde{e}) = \texttt{back}(h')$) and the score q, which is calculated as the difference of the scores of h' and h: $q = q(h') - q(h)$.

A word graph can be seen as a compact representation of an N-best list. A single translation is given by a path from the initial state to a state representing a complete hypothesis in the word graph. The information stored in word graphs allows us to retrieve not only the translations associated to the best states of the language and the translation models, but also those translations associated to recombined hypotheses. Different algorithms have been proposed in the literature to obtain N-best lists from word-graphs [Epp99, UON02, JM03].

In this thesis we use word graphs as a key component in IMT systems, as will be shown in section 6.3.

4.3 Efficient Decoding with Very Large Phrase-Based Models

The great size of the phrase tables used in PB-SMT is a source of problems not only during the training process as explained in the previous chapter (see section 3.4), but also during the decoding process, since the whole phrase table is to be stored in memory.

A simple solution to this problem is to extract the subset of the phrase table that is needed to translate a test set and to store it in memory. This solution is incorporated in translation systems like the Pharaoh decoder, but it is not a general and realistic solution since the test set must be previously known.

An approach that has been more successful consists in the use of data structures with very low memory requirements [CBBS05, ZV05]. However, these techniques may not be suitable for very large corpora unless there are machines with great memory sizes (2 GBytes or more).

We propose an alternative way to solve this problem which is strongly inspired by a classic concept of computer architecture: *cache memory*. Our proposed solution also uses a specific data structure to represent phrase tables which is different to previously defined data structures with the same purpose. The proposed techniques involve accessing model parameters from disk and have points in common with those described in [ZN07] and in [FC07].

Zens et al. [ZN07] propose efficient techniques to access phrase-based model probabilities using a prefix-tree structure for the source phrases. Federico et al. [FC07] apply different techniques, including caching of probabilities, to efficiently access the parameters of n-gram language models.

4.3.1 Cache Memory Architecture

Cache memory is based on the *principle of locality* of references: if one location is read by a program, then nearby locations are likely to be read soon afterward. In the case of machine translation, this principle manifests itself in two different ways:

1. The majority of the phrase pairs contained in a phrase model have a very low frequency. Therefore, we can predict that these phrase pairs will probably not be required during the decoding process.

2. When translating a sentence, only a small number of the entries that compose the phrase model are accessed. Additionally, each entry will be accessed many times because of the iterative nature of the decoding process. Therefore, we can identify both temporal and spatial locality principles.

The locality principle explained above leads us to propose a memory hierarchy composed of two levels. The first level stores the bilingual pairs that are accessed during the translation of each sentence. This level is local to the sentence to be translated, and will be erased whenever the translation process of a new sentence is started.

The second level contains a certain percentage of the most frequent phrase pairs stored within the phrase model. This level is kept in memory during the whole translation process.

Finally, the whole phrase table is stored on the hard disk and is structured to allow the retrieval of the probability of the bilingual pairs. This is done with logarithmic complexity by means of binary search.

It is important to point out that the basic information element that is handled within the memory hierarchy consists of a single source phrase $\tilde{f}$ with all its target translations. This is done to favour spatial locality.

Thus, when the decoder needs to retrieve the probability of a phrase pair $(\tilde{f}, \tilde{e})$, it searches for the pair in the first level cache. If it is present, its probability is returned. Otherwise, the translations of $\tilde{f}$ are searched for in the second level cache. If these translations exist, they are copied in the first level cache and the probability of the phrase pair is returned if $\tilde{e}$ has been stored as a possible translation of $\tilde{f}$. If there is no translation for $\tilde{f}$ in the second level cache, then the hard disk is accessed.

When the translations of $\tilde{f}$ are searched for in the hard disk, they may or may not exist. In either case, the result of the search is copied in the first level cache, and the probability of the phrase pair is returned.

When the translation process of each sentence has finished, the first level cache is erased, and the decoder only keeps in memory the selected percentage of the model. The percentage of phrase pairs that are stored in the second level cache will be referred to as the α parameter. According to the first locality principle explained above, the phrase pairs stored in the second

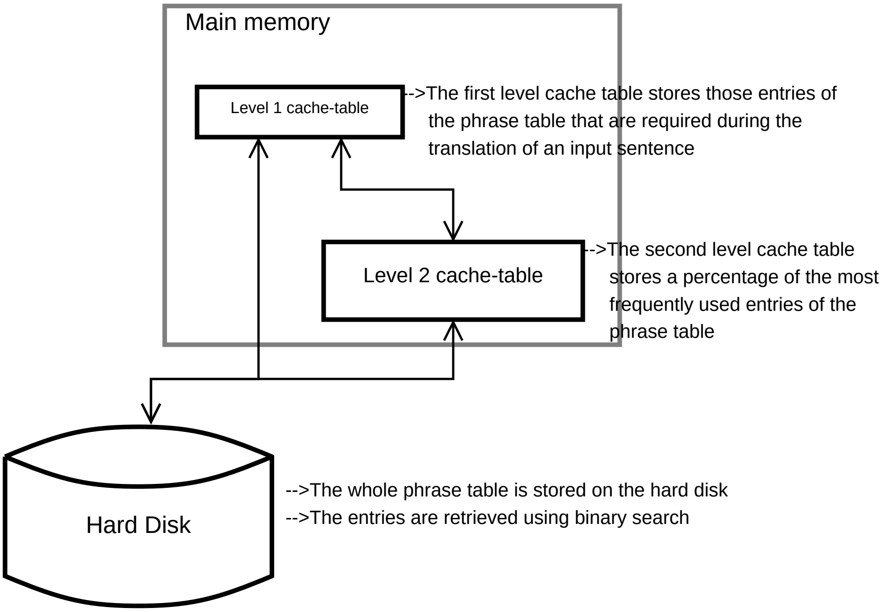

Main memory
Level 1 cache-table
-->The first level cache table stores those entries of the phrase table that are required during the translation of an input sentence
Level 2 cache-table
-->The second level cache table stores a percentage of the most frequently used entries of the phrase table
Hard Disk
-->The whole phrase table is stored on the hard disk
-->The entries are retrieved using binary search

Figure 4.2: Cache memory architecture.

level will be those with higher frequency. Figure 4.2 shows a diagram of the above described cache memory architecture.

In the experiments we have carried out, α takes values between 0 and 100. Both these values are particular cases with interesting features:

α**=0** : the second-level cache will be empty. Therefore, there is no phrase pair permanently stored in memory. This will increase the amount of cache misses. However, it allows us to translate without having to store the model in memory.

α**=100** : the whole model will be stored in the second-level cache (i.e. the whole model is allocated in memory and the retrievals are cached). This allows us to translate without any cache misses and can be viewed as the baseline that is implemented by common decoders such as the Pharaoh decoder.

4.3.2 Selecting a Suitable Data Structure for Phrase Pairs

Because of the huge size of the phrase tables, it is crucial to find a data structure with low memory requirements to represent the phrase pairs.

For the training process described in section 3.4, the choice of the representation for the phrase pairs is not an important problem, since it is possible to reduce the memory requirements by simply reducing the fragment size.

However, the data structures must be carefully chosen for the case of the decoding process. Specifically, it is important to use a fast data structure to represent the first-level cache table, and to use a low complexity data structure in terms of space to represent the second-level cache table.

In our work, we have used the same representation for the first- and the second-level cache memory. Such a representation makes a tradeoff between time and space complexity and consists in an *asymmetrical double trie* like the one shown in Fig. 4.3, where there is a trie associated to the source language (left) and another associated to the target language (right). In the upper part of the figure, a small set of English-Spanish phrases is shown. In the lower part of the figure a depiction is given of how these phrase pairs are stored by the proposed data structure.

la casa roja - the red house

la casa roja - the red flat

la casa verde - the green house

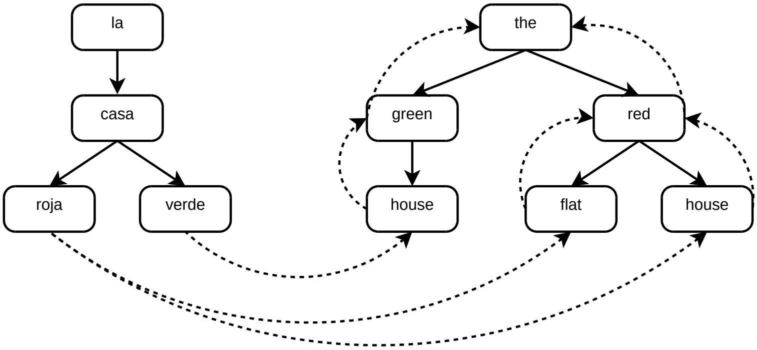


Figure 4.3: An example of the double-trie data structure for the storage of bilingual pairs. The trie at the left stores the source phrases and the one at the right stores the target phrases.

In order to retrieve the probability $p(\tilde{f}|\tilde{e})$ of a phrase pair $(\tilde{f}, \tilde{e})$, first, the target phrase $\tilde{e}$ is to be searched in the target trie. As a result of the search, a pointer that represents the target phrase and the count of the target phrase $c(\tilde{e})$ are obtained. Second, the source phrase $\tilde{f}$ is to be searched in the source trie. Once the search is done, we have to find the pointer to $\tilde{e}$ that was obtained in the previous step. This final step allows us to retrieve $c(\tilde{f}, \tilde{e})$. Once the two counts are retrieved, the probability of the phrase pair is given by $c(\tilde{f}, \tilde{e})/c(\tilde{e})$.

The number of comparisons that are to be done in order to retrieve the probability of the phrase pair $(\tilde{f}, \tilde{e})$ is given by the following expression:

$$log(s) + log(t) + n\,, \tag{4.20}$$

where s and t are the number of source words and target words respectively, that are stored by the data structure, and n is the number of source phrases that translates the target phrase $\tilde{e}$.

Given that $s \approx t$ and $n \ll s$ (see Table 3.3), we can conclude that the retrieval has a logarithmic complexity.

The proposed data structure also allows us to obtain the set of translations of a given source phrase $\tilde{f}$, which is a basic operation performed by standard decoding algorithms. For this purpose, the source phrase $\tilde{f}$ is to be searched in the source trie. As a result of the search, the set of pointers which represents all possible translations of $\tilde{f}$ are obtained. The target phrases that are represented by each pointer can be obtained by means of the pointers to the father nodes stored in the target trie.

With regard to the space complexity, if we implement tries as binary trees (as proposed in [Knu73]), and assuming that an integer number is represented with one word from the processor, the number of integers required to store the whole statistical dictionary of phrase pairs using the data structure we have defined is given by the following expression:

$$s \times c_s \times 3 + t \times c_t \times 5 + pp \times 2$$

where s and t are the total number of source and target words, respectively, stored in the phrase table; c_s and c_t are factors between 0 and 1 whose meaning will be described below; and pp is the number of pairs which compose the phrase table.

Each trie node requires 3 integers, one integer to store the word it contains, and another two as pointers to child and brother nodes. The target trie requires an additional integer in each node to store the pointer to its father node, and one more to store the count of the phrase it represents. Finally, each alignment between bilingual pairs requires one integer and the count of the phrase pair one integer more.

With regard to the factors c_s and c_t, they represent the compression ratio obtained by the use of the trie data structure for the source and the target languages, respectively. Specifically, the tries will compress all those phrases that share the same prefix (see Fig. 4.3). Such a compression is represented in the expression shown above with factors between 0 and 1. The value of these factors depends on the features of the corpus. For instance, for the Europarl corpus these compression factors are not greater than 0.3 for both the source and the target tries.

Our proposed data structure not only allows the retrieval of inverse probabilities $p(\tilde{f}|\tilde{e})$, but also the retrieval of direct probabilities $p(\tilde{e}|\tilde{f})$. Since $p(\tilde{e}|\tilde{f})$ is given by $c(\tilde{f},\tilde{e})/c(\tilde{f})$, now we need to retrieve the values of the counts $c(\tilde{f},\tilde{e})$ and $c(\tilde{f})$. Regarding the value of $c(\tilde{f},\tilde{e})$, it is obtained in the same way as it was shown for the retrieval of inverse probabilities. With respect to the value of $c(\tilde{f})$, since $c(\tilde{f}) = \sum_{\tilde{e}'} c(\tilde{f},\tilde{e}')$, we only have to search for the phrase $\tilde{f}$ in the source trie and sum the joint counts $c(\tilde{f},\cdot)$ for all its translations, obtaining the value of $c(\tilde{f})$. It is worth pointing out that this data structure can be used in combination with the training procedure described in section 3.4.2, allowing direct and inverse probabilities to be accessed after executing the training process in only one translation direction. Specifically, the training procedure generates the counts $c(\tilde{f},\tilde{e})$ and $c(\tilde{e})$ for each phrase pair, and our proposed data structure uses these counts to efficiently generate direct and inverse probabilities for each phrase pair.

In addition to the above commented features, the use of counts allows the proposed data structure to be dynamically modified (i.e. new phrase pairs can be added or the counts of existing ones can be modified). Such capability will be exploited to implement an incremental phrase-based model, as it will be shown in chapter 7.

In spite of the low space complexity of the data structure presented in this work, more efficient implementations have been proposed, such as the suffix-arrays described in [CBBS05, ZV05]. However, suffix-arrays are not able to efficiently obtain exact probabilities for the phrase pairs; instead, the probabilities are approximated so as to reduce the retrieval costs. By contrast, as it has been shown, our proposed data structure allows to obtain exact probabilities with logarithmic time complexity.

4.4 Phrase-Level Alignment Generation

In this section we study the problem of generating alignments at phrase level between a sentence pair. As is discussed below, the generation of phrase-level alignments is not a trivial task due to problems with unseen events, which may cause that a given sentence pair cannot be adequately aligned. The phrase-level alignment generation techniques proposed here allows us to solve this problem and can be useful in a range of applications, including multisource SMT [ON01], Viterbi-like estimation of phrase-based models [WMN10], discriminative training [LBCKT06], training of phrase segmentation models [SDAS08], etc. Moreover, in Chapter 6 of this thesis, we show how the problem of generating phrase-level alignments can be modified for its application in IMT (see section 6.2).

The problem of finding the best alignment at phrase level has not been extensively addressed in the literature. A first attempt can be found in [GVON$^+$05], where different techniques to obtain alignments at phrase level are proposed. However, the proposed techniques heavily rely on word alignment models or on word alignment matrices.

As was explained in section 1.4.3, the concept of bisegmentation and the concept of phrase-based alignment for a sentence pair are interchangeable. A bisegmentation or phrase-based alignment of length K of a sentence pair (f_1^J, e_1^I), $\tilde{A}(f_1^J, e_1^I)$, is defined as a triple: $\tilde{A}(f_1^J, e_1^I) \equiv (\tilde{f}_1^K, \tilde{e}_1^K, \tilde{a}_1^K)$, where $\tilde{f}_1^K \equiv f_1^J$, $\tilde{e}_1^K \equiv e_1^I$ and $\tilde{a}_1^K$ is a specific one-to-one mapping between the K segments/phrases of both sentences $(1 \leq K \leq \min(I, J))$.

Then, given a pair of sentences (f_1^J, e_1^I) and a phrase model, we are interested in the best phrase-alignment (or Viterbi phrase-alignment), $\tilde{A}_V(f_1^J, e_1^I)$, between them. $\tilde{A}_V(f_1^J, e_1^I)$ can be computed as[a]:

$$\tilde{A}_V(f_1^J, e_1^I) = \operatorname*{arg\,max}_{\tilde{f}_1^K, \tilde{e}_1^K, \tilde{a}_1^K} \left\{ Pr(\tilde{f}_1^K, \tilde{a}_1^K | \tilde{e}_1^K) \right\} \tag{4.21}$$

where, following the assumptions given in [Tom03], $Pr(\tilde{f}_1^K, \tilde{a}_1^K | \tilde{e}_1^K)$ can be approximated as:

$$p(\tilde{f}_1^K, \tilde{a}_1^K | \tilde{e}_1^K) = \prod_{k=1}^{K} p(\tilde{a}_k | \tilde{a}_1^{k-1}) \cdot p(\tilde{f}_k | \tilde{e}_{\tilde{a}_k}) \tag{4.22}$$

On the basis of Equation (4.22), a very straightforward technique can be proposed for finding the best phrase-alignment of a sentence pair (f_1^J, e_1^I). This can be conceived as a sort of *constrained* translation. In this way, the search process only requires the use of a regular SMT system which filters its phrase table in order to obtain those translations of f_1^J that are compatible with e_1^I.

[a] It should be noted that $Pr(\tilde{a}_1^K | \tilde{f}_1^K, \tilde{e}_1^K) = Pr(\tilde{f}_1^K, \tilde{a}_1^K | \tilde{e}_1^K) / Pr(\tilde{f}_1^K | \tilde{e}_1^K)$.

In spite of its simplicity, this technique has no practical interest when applied on regular tasks. Specifically, the technique is not applicable when the alignments cannot be generated due to coverage problems of the phrase-based alignment model (i.e. one or more phrase pairs required to compose a given alignment have not been seen during the training process). This problem cannot be easily solved, since standard estimation tools such as THOT and Moses do not guarantee the complete coverage of sentence pairs even if they are included in the training set; this is due to the great number of heuristic decisions involved in the estimation process. In addition to this, certain search space pruning techniques may also introduce coverage problems, such as the reordering constraints (see section 4.2.8).

One possible way to overcome the above-mentioned coverage problems requires the definition of an alternative technique that is able to consider every source phrase of f_1^J as a possible translation of every target phrase of e_1^I. Such a technique requires the following two elements:

1. A search algorithm that enables efficient exploration of the set of possible phrase-alignments for a sentence pair.

2. A general mechanism to assign probabilities to phrase pairs, no matter if they are contained in the phrase table or not.

In the following sections we describe the details of the technique used to obtain phrase alignments, focusing our attention on the two key elements that have been mentioned above. Specifically, the search algorithm used to explore the set of possible alignments is described in section 4.4.1, and the mechanism to assign probabilities to phrase pairs is described in sections 4.4.2 and 4.4.3.

4.4.1 Search Algorithm

Regarding the search algorithm to be used, we propose the use of a modified version of the branch and bound search algorithm for PB-SMT described in section 4.2. Except for the scoring function (which will be studied in sections 4.4.2 and 4.4.3), only the expansion algorithm has to be appropriately modified to allow the exploration of the set of possible phrase-alignments.

The expansion process consists in appending target phrases as translation of previously unaligned source phrases of a given hypothesis. Let us suppose that we want to obtain a phrase alignment between the sentences $f_1^J \equiv$ "Para ver la lista de recursos", and $e_1^I \equiv$ "To view a list of resources". Figure 4.4 shows an example of the results obtained by the expansion algorithm that we propose for a given hypothesis h.

The hypothesis h has aligned the source phrase "Para", appending the target phrase "To". The set $\mathcal{SP}_h$ contains the set of unaligned source positions of h, $\mathcal{SP}_h = \{2,3,4,5,6\}$. The expansion algorithm works by aligning source phrase positions, $u = (j_1, j_2)$, from the set $PP(\mathcal{SP}_h)$, where the function $PP(\cdot)$, given a set of word positions returns the set of all possible phrase positions that can be obtained using these word positions (the $PP(\cdot)$ function is formally defined by Equation (4.5)).

Let $e_{i'}^I \equiv$ "view a list of resources" be the remaining words that are to be appended to h to complete the target sentence e_1^I. Under these circumstances, we have to take into account

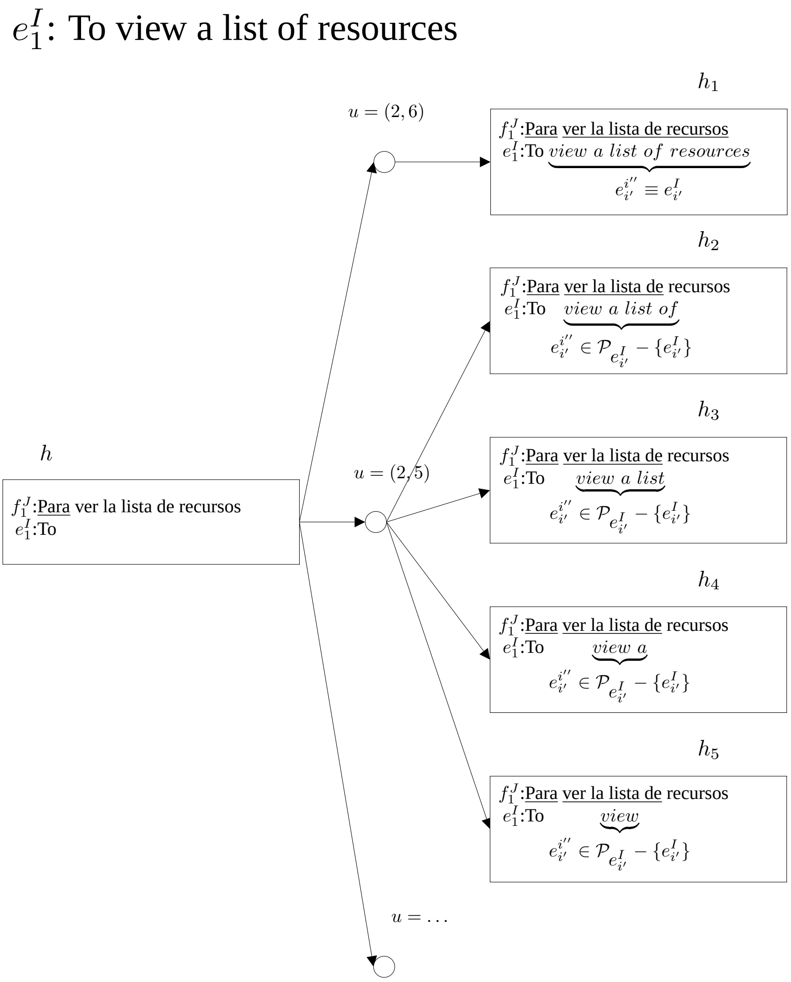

e_1^I: To view a list of resources
h
f_1^J:Para ver la lista de recursos
e_1^I:To
$u=(2,6)$
$u=(2,5)$
$u=\ldots$
h_1
f_1^J:Para ver la lista de recursos
e_1^I:To $\underbrace{view\ a\ list\ of\ resources}$
$e_{i'}^{i''} \equiv e_{i'}^I$
h_2
f_1^J:Para ver la lista de recursos
e_1^I:To $\underbrace{view\ a\ list\ of}$
$e_{i'}^{i''} \in \mathcal{P}_{e_{i'}^I} - \{e_{i'}^I\}$
h_3
f_1^J:Para ver la lista de recursos
e_1^I:To $\underbrace{view\ a\ list}$
$e_{i'}^{i''} \in \mathcal{P}_{e_{i'}^I} - \{e_{i'}^I\}$
h_4
f_1^J:Para ver la lista de recursos
e_1^I:To $\underbrace{view\ a}$
$e_{i'}^{i''} \in \mathcal{P}_{e_{i'}^I} - \{e_{i'}^I\}$
h_5
f_1^J:Para ver la lista de recursos
e_1^I:To $\underbrace{view}$
$e_{i'}^{i''} \in \mathcal{P}_{e_{i'}^I} - \{e_{i'}^I\}$

Figure 4.4: Example of the expansion of the hypothesis h given $f_1^J \equiv$ “Para ver la lista de recursos” and the target sentence $e_1^I \equiv$ “To view a list of resources”.

whether we are aligning the last source phrase positions or not. For example, let us suppose that we align the source phrase determined by positions $u = (2, 6) \in PP(\mathcal{SP}_h)$ ($f_2^6 \equiv$ “ver la lista de recursos”). Since those are the last source positions to be aligned, we have to ensure that the whole target sentence e_1^I is generated. For this purpose, we append $e_{i'}^I$ to h, resulting in the hypothesis h_1.

By contrast, if we are not aligning the last source positions of h, we can also append strings, $e_{i'}^{i''}$, from the set $\mathcal{P}_{e_{i'}^I}$ of sub-prefixes of $e_{i'}^I$, with the exception of $e_{i'}^I$ itself, to the newly generated hypotheses. Before appending a string to a hypothesis, we have to ensure that after aligning a source phrase, there are enough remaining target words to be aligned with

```
input    : $e_1^I$ (reference target sentence), $h$ (hypothesis to be expanded)
output   : $\mathcal{H}$ (set of expanded hypotheses)
auxiliar: $\mathcal{SP}_h$ (set of unaligned source word positions of $h$),
          $PP(\mathcal{SP}_h)$ (set of all possible unaligned source phrase positions of $h$),
          $LP(\mathcal{SP}_h)$ (set of phrase positions of the longest unaligned phrases),
          $e_{i'}^I$ (remaining target sentence to be aligned at phrase level),
          $\mathcal{P}_{e_{i'}^I}$ (set of sub-prefixes of $e_{i'}^I$)
begin
    forall $(j_1, j_2) \in PP(\mathcal{SP}_h)$ do
        $e_{i'}^I$ =get_remaining_trg_sent ($h$,$e_1^I$);
        if $\mathcal{SP}_h - \{j_1, ..., j_2\} \neq \emptyset$ then
            forall $e_{i'}^{i''} \in \mathcal{P}_{e_{i'}^I} - \{e_{i'}^I\}$ do
                if $(|e_{i'}^I| - |e_{i'}^{i''}|) \geq |LP(\mathcal{SP}_h - \{j_1, ..., j_2\})|$ then
                    $\mathcal{H} := \mathcal{H} \cup \{$append$(h, ((j_1, j_2), e_{i'}^{i''}))\}$
        else
            $\mathcal{H} := \mathcal{H} \cup \{$append$(h, ((j_1, j_2), e_{i'}^I))\}$
end
```

Algorithm 4.6: Pseudocode for the `phralig_expand` algorithm.

unaligned source phrases. The number of remaining target words is given by: $(|e_{i'}^I| - |e_{i'}^{i''}|)$. Given the current set of unaligned positions, $\mathcal{SP}_h$, and the next source phrase to be aligned, determined by the positions (j_1, j_2), we have to align at least $|LP(\mathcal{SP}_h - \{j_1, ..., j_2\})|$ source phrases, where the function $LP(\cdot)$, given a set of word positions, returns the set of phrase positions determining the longest phrases that can be obtained using these word positions ($LP(\cdot)$ was formally defined by Equation (4.16)). Thus, we have to ensure that the following condition holds: $(|e_{i'}^I| - |e_{i'}^{i''}|) \geq |LP(\mathcal{SP}_h - \{j_1, ..., j_2\})|$. These restrictions allow the translation system to complete the target sentence e_1^I in subsequent expansion processes.

As an example of the previous considerations, let us suppose that we align the source phrase determined by positions $u = (2, 5) \in PP(\mathcal{SP}_h)$ ($f_2^5 \equiv$ "ver la lista de"). In this case we can append the string "view a list of", resulting in the hypothesis h_2. Alternatively, the subprefix of $e_{i'}^I$, "view a list", can be appended, resulting in the hypothesis h_3. Finally, we can also append the strings "view a" and "view" resulting in the hypotheses h_4 and h_5, respectively. In all cases the above explained restrictions were satisfied, since $|LP(\mathcal{SP}_h - \{2, 3, 4, 5\}|$ is equal to 1 and $(|e_{i'}^I| - |e_{i'}^{i''}|)$ was greater or equal to 1 for the appended strings.

Algorithm 4.6 shows the expansion algorithm that we propose for its application in the generation of phrase alignments. The algorithm is a formalisation of the ideas depicted in Figure 4.4. This expansion algorithm permits phrase reorderings, but it can be easily modified to only obtain monotonic alignments.

The time cost of the `phralig_expand` algorithm can be reduced by the introduction of pruning techniques. In this case, the only pruning technique that we propose to apply consists in restricting the maximum number of target phrases that can be linked to an unaligned source phrase during the expansion process. Specifically, in those cases where e_1^I has not already been generated, only a subset of the strings contained in the set $\mathcal{P}_{e_{i'}^{I}}$ are considered as candidates for the expansion process. One possible criterion to choose the substrings is based on the length of the source phrase $f_{j_1}^{j_2}$ to be aligned determined by u. Only those target substrings with lengths similar to the length of $f_{j_1}^{j_2}$ are considered.

Regarding the complexity of the search algorithm, it depends on the selected configuration of the branch-and-bound algorithm. Let us assume that we use a breadth-first multiple stack algorithm with J stacks (see section 4.2.5). Under these circumstances, the search algorithm has to expand L_s hypothesis from J stacks. The complexity of the `phralig_expand` algorithm is in $\mathcal{O}(J^2 \cdot I)$. Therefore, the complexity of the search algorithm is in $\mathcal{O}(J^3 \cdot L_s \cdot I)$.

4.4.2 Smoothing Techniques

During the generation of phrase alignments using the search algorithm that has been described above, it is necessary to assign scores to the phrase pairs that are appended to the newly generated hypothesis. Since these appended phrase pairs are not necessarily extracted from the phrase table (in principle, given a sentence pair, each target phrase can be considered as translation of a given source phrase), a general scoring mechanism is required to assign probabilities to phrase pairs, no matter if they are contained in the phrase table or not.

To solve this problem, we propose the application of smoothing techniques over the phrase table. Although smoothing is an important issue in language modelling and other areas of statistical NLP (see for example [MS01] for more details), it has not received much attention from the SMT community. However, most of the well-known language model smoothing techniques can be imported to the SMT field and specifically to the PB-SMT framework, as it is shown in [FKJ06].

In spite of the fact that PB-SMT and the generation of phrase-alignments are similar tasks, it should be noted that the two problems differ in a key aspect. While in PB-SMT the probabilities of unseen events are not important (since the decoder only proposes phrase translations that are contained in the model, see [FKJ06]), in the generation of phrase alignments, assigning probabilities to unseen events is one of the most important problems that has to be solved.

In the rest of this section, we describe the smoothing techniques that we have implemented. They are similar to those proposed in [FKJ06], although in our case we have strongly focused on the appropriate treatment of unseen events.

Phrase-based Model Statistical Estimators

Training data can be exploited in different ways to estimate statistical models. Regarding the phrase-based models, the standard estimation technique is based on the relative frequencies of the phrase pairs (see section 3.2). Taking this standard estimation technique as a starting point, a number of alternative estimation techniques can be derived.

In this thesis we propose the application of the following estimation techniques for phrase-based models:

- Maximum-likelihood estimation
- Good-Turing estimation

- Absolute-discount estimation
- Kneser-Ney smoothing
- Simple discount

As was mentioned above, ML estimation uses the concept of relative frequency as a probability estimate. Once the counts of the phrase pairs have been obtained, different well-known estimation techniques can be applied. In this thesis we propose the application of Good-Turing, Absolute-discount, Kneser-Ney and Simple discount estimation.

The well-known Good-Turing smoothing technique [CG91] works by replacing observed counts, c, by modified counts, c_g:

$$c_g = (c+1) \cdot \frac{n_{c+1}}{n_c} \tag{4.23}$$

where c_g is a modified count value used to replace c in subsequent relative-frequency estimates, and n_c is the number of events having count c.

It follows from Equation (4.23) that the total count mass assigned to unseen phrase pairs is $0_g n_0 = n_1$. This mass is distributed among contexts $\tilde{e}$ in proportion to $c(\tilde{e})$, giving the Good-Turing estimator:

$$p(\tilde{f}|\tilde{e}) = \frac{c_g(\tilde{f},\tilde{e})}{\sum_{\tilde{f}} c_g(\tilde{f},\tilde{e}) + p(\tilde{e}) \cdot n_1} \tag{4.24}$$

where $p(\tilde{e}) = c(\tilde{e})/\sum_{\tilde{e}} c(\tilde{e})$.

Absolute-discounting and Kneser-Ney smoothing subtract a fixed discount from all non-zero counts [KN95]. The general probability distribution for these smoothing techniques was originally defined as a backoff combination by Kneser and Ney [KN95] and later reframed as an interpolation by Chen and Goodman [CG96]. Here we adopt the interpolated version:

$$p(\tilde{f}|\tilde{e}) = \frac{c(\tilde{f},\tilde{e}) - D}{\sum_{\tilde{f}} c(\tilde{f},\tilde{e})} + \alpha(\tilde{e}) \cdot p_b(\tilde{f}|\tilde{e}) \tag{4.25}$$

where D is the subtracted fixed discount, $\alpha(\tilde{e})$ is the normalisation factor and $p_b(\tilde{f}|\tilde{e})$ is the smoothing distribution. Following the formulae given in [NEK94], the fixed discount is calculated as $D = n_1/(n_1 + 2n_2)$. The normalisation factor $\alpha(\tilde{e})$ is defined as follows:

$$\alpha(\tilde{e}) = \frac{D}{\sum_{\tilde{f}} c(\tilde{f},\tilde{e})} \cdot N_{1+}(\bullet,\tilde{e}) \tag{4.26}$$

where $N_{1+}(\bullet,\tilde{e})$ is the number of source phrases $\tilde{f}$ for which $c(\tilde{f},\tilde{e}) > 0$.

Regarding the smoothing distribution, $p_b(\tilde{f}|\tilde{e})$, we define two different versions. For the sake of simplicity, in these two versions, the distribution dependency on $\tilde{e}$ is removed. The first version is given by $p(\tilde{f}) = c(\tilde{f})/\sum_{\tilde{f}} c(\tilde{f})$, and the second one is the Kneser-Ney lower order distribution:

$$p_b(\tilde{f}) = N_{1+}(\tilde{f},\bullet)/\sum_{\tilde{f}} N_{1+}(\tilde{f},\bullet) \tag{4.27}$$

where $N_{1+}(\tilde{f}, \bullet)$ is defined analogously to $N_{1+}(\bullet, \tilde{e})$.

Finally, we propose the use of an alternative discount which we have called Simple discount, where a fixed probability mass is discounted from seen events instead of a fixed count:

$$p(\tilde{f}|\tilde{e}) = (1 - \delta) \cdot \frac{c(\tilde{f}, \tilde{e})}{\sum_{\tilde{f}} c(\tilde{f}, \tilde{e})} \tag{4.28}$$

where δ is the discounted probability mass assigned to unseen phrase pairs.

Lexical Distributions

A good way to tackle the problem of unseen events is the use of probability distributions that decompose phrases into words. Two different techniques are mentioned in [FKJ06] for this purpose: the noisy-or and an alternative technique which is based on alignment matrices.

In this thesis we have applied another technique which is based on the IBM 1 Model probability as defined in [BDDM93]:

$$p(f_1^J|e_1^I) = \frac{\epsilon}{(I+1)^J} \cdot \prod_{j=1}^{J} \sum_{i=1}^{I} p(f_j|e_i) \tag{4.29}$$

We use the IBM 1 model to assign probabilities to phrase pairs instead of sentence pairs, i.e. we obtain probabilities for individual phrases $\tilde{f}$ and $\tilde{e}$ instead of for the sentences f_1^J and e_1^I.

Combining Estimators

The statistical estimators described above can be combined in the hope of producing better models. We have chosen three different well-known techniques for combining estimators:

- Linear interpolation
- Backing-off
- Log-linear interpolation

The linear interpolation technique consists of making a linear combination of different estimators, ensuring that the weights of such combination determine a probability function. The general form of the interpolation schemes proposed here is as follows:

$$p_{LI}(\tilde{f}|\tilde{e}) = \lambda_{\tilde{f},\tilde{e}} \cdot p_{PB}(\tilde{f}|\tilde{e}) + (1 - \lambda_{\tilde{f},\tilde{e}}) \cdot p_{LEX}(\tilde{f}|\tilde{e}) \tag{4.30}$$

where p_{PB} is a phrase-based statistical estimator and p_{LEX} is the lexical distribution described above.

The backing-off combination technique consults different models in order depending on their specificity. The general form of the proposed backoff schemes is as follows:

$$p_{BO}(\tilde{f}|\tilde{e}) = \begin{cases} \alpha(\tilde{f}|\tilde{e}) & if\ c(\tilde{f}, \tilde{e}) > 0 \\ \gamma(\tilde{e}) \cdot p_{LEX}(\tilde{f}|\tilde{e}) & if\ c(\tilde{f}, \tilde{e}) = 0 \end{cases} \tag{4.31}$$

where $\alpha(\tilde{f}|\tilde{e})$ is a modified version of $p_{PB}(\tilde{f}|\tilde{e})$ that reserves some probability mass for unseen events, the scale factor $\gamma(\tilde{e})$ is introduced to make the conditional distribution sum up to one and $p_{LEX}(\tilde{f}|\tilde{e})$ is the lexical distribution described above.

Phrase-based model estimators and lower order distributions can also be combined by means of log-linear interpolation. In this case, the procedure consists in adding a phrase based statistical estimator and a lexical distribution as scoring components of a log-linear model.

In all cases, the main goal of the different combinations of statistical estimators is to achieve good treatment of unseen events. However, each combination technique has its own properties.

The key difference between interpolation and backing off is that the latter only uses information from the smoothing distribution (the lexical distribution) for low frequency or unseen events. Since for phrase alignment generation, better prediction of unseen events has a great impact, backing-off seems a especially suitable approach.

Finally, the main difference between linear and log-linear combination is that the former moderates extreme probability values and preserves intermediate values, whereas the latter preserves extreme values and makes intermediate values more extreme. When assigning probabilities to unseen events, the phrase-based model statistical estimators will produce very low or zero probabilities that will be moderated by linear combination (using the LEX distribution), and preserved by log-linear combination. Because of this, we expect linear combination to work better than log-linear combination.

4.4.3 A loglinear Approach to Phrase-to-Phrase Alignments

As was explained above, the score of a given alignment can be calculated according to Equation (4.22). This scoring function can be refined to take into account some basic aspects of a phrase alignment, such as the lengths of the source and target phrases, and the reorderings of phrase alignments. For this purpose, we follow the specific phrase-based model decomposition presented in section 3.5, where the alignment variable $\tilde{a}_1^K$ is replaced by our own set of hidden variables (K, a_1^K, b_1^K, c_1^K). In this set of alignment variables, K represents the length of the bisegmentation, a_1^K is a vector of ending positions of the K target phrases, b_1^K is a vector with the number of skipped source positions with respect to the ending position of the previously aligned source phrase and c_1^K represents a vector containing the lengths of the K source phrases.

The new scoring function given by our proposed phrase-based model decomposition is as follows:

$$\begin{aligned} p(f_1^J, K, a_1^K, b_1^K, c_1^K | e_1^I) \quad &= \quad p(J|I) \cdot p(K|I,J) \cdot \prod_{k=1}^{K} \Big[p(a_k|a_{k-1}) \cdot \\ & \qquad p(b_k) \cdot p(c_k|a_k, a_{k-1}) \cdot p(f_{\alpha_k}^{\beta_k} | e_{a_{k-1}+1}^{a_k}) \Big] \end{aligned} \tag{4.32}$$

where the following submodels are included: a source sentence length submodel, $p(J|I)$, a bisegmentation length submodel, $p(K|I,J)$, a target phrase length submodel, $p(a_k|a_{k-1})$, a reordering submodel, $p(b_k)$, a source phrase length submodel, $p(c_k|a_k, a_{k-1})$ and an inverse phrase translation submodel, $p(f_{\alpha_k}^{\beta_k}|e_{a_{k-1}+1}^{a_k})$; the α and β variables are defined as follows:

$$\begin{aligned}\alpha_k &= \beta_k - c_k + 1\\ \beta_k &= \beta_{k-1} + b_k + c_k\\ \beta_0 &= 0\end{aligned}$$

It should be noted that the source sentence length submodel always assigns the same probability to each possible phrase alignment between f_1^J and e_1^I, since in all cases the sentence lengths remain unchanged. Additionally, according to the phrase model derivation presented in section 3.5, the bisegmentation length submodel is assumed to be uniform. Thus, the calculation of the probability for a given phrase alignment given by Equation (4.32) can be simplified by dropping the terms $p(J|I)$ and $p(K|I, J)$.

In a similar way as it was described in section 3.5.3 for the case of standard PB-SMT, we can introduce the submodels that are present in Equation (4.32) as components of a log-linear model (except the source sentence length and the bisegmentation length submodels). This log-linear model can also be complemented with additional score components. The generation of the Viterbi phrase alignment using this log-linear model can be formally expressed as follows:

$$\tilde{A}_V(f_1^J, e_1^I) = \underset{K, a_1^K, b_1^K, c_1^K}{\arg\max} \left\{ \sum_{m=1}^{M} \lambda_m \cdot h_m(f_1^J, K, a_1^K, b_1^K, c_1^K, e_1^I) \right\} \tag{4.33}$$

We propose the use of a specific instantiation of the previous general log-linear model given by Equation (4.33). Specifically, this log-linear model instantiation is composed of a total of five score components or feature functions (from h_1 to h_5): inverse and direct phrase-based models (h_1 and h_2 respectively), a target phrase-length model (h_3), a source phrase-length model (h_4), and a distortion model (h_5). The details for each feature function are listed below:

- **inverse phrase-based model (h_1)**
  $h_1(e_1^I, K, a_1^K, b_1^K, c_1^K, f_1^J) = \log(\prod_{k=1}^{K} p(f_{\alpha_k}^{\beta_k}|e_{a_{k-1}+1}^{a_k}))$

- **direct phrase-based model (h_2)**
  $h_2(e_1^I, K, a_1^K, b_1^K, c_1^K, f_1^J) = \log(\prod_{k=1}^{K} p(e_{a_{k-1}+1}^{a_k}|f_{\alpha_k}^{\beta_k}))$

- **target phrase-length model (h_3)**
  $h_3(K, a_1^K) = \log(\prod_{k=1}^{K} p(a_k|a_{k-1}))$

- **source phrase-length model (h_4)**
  $h_4(K, a_1^K, c_1^K) = \log(\prod_{k=1}^{K} p(c_k|a_k, a_{k-1}))$

- **distortion model (h_5)**
  $h_5(K, b_1^K) = \log(\prod_{k=1}^{K} p(b_k))$

To implement the above described score components we use the same probability models that were proposed in section 3.5.2 for its use in standard PB-SMT. Specifically, we use inverse and direct phrase models to implement h_1 and h_2 respectively; h_3 can be implemented by means of a geometric probability distribution (penalises long target phrases) or a uniform probability distribution (penalises the length of the bisegmentation); a geometric, a uniform or a Poisson probability distribution can be used to implement h_4 (geometric and Poisson distributions penalise the difference between the length of the source and target phrases and the uniform distribution penalises the length of the bisegmentation); finally, h_5 is implemented by means of a geometric probability distribution.

It is worth noticing that the inverse and the direct phrase models used to implement the scoring functions h_1 and h_2, respectively, can be smoothed using the techniques that were described in section 4.4.2.

Regarding the weights of the log-linear combination, λ_m, $m \in \{1, 2, 3, 4, 5\}$, they can be computed by means of the MERT algorithm [Och03].

4.5 Summary

In this chapter we have studied different issues regarding the search problem in PB-SMT. We have proposed a branch-and-bound search algorithm for PB-SMT. The computational complexity of the proposed algorithm is bounded by the complexity of the well-known dynamic programming algorithm defined in [Zen07]. We have provided both single- and multiple-stack versions of the basic search algorithm. Our proposed algorithm performs a best-first search by default, but it can also perform a breadth-first search by only appropriately modifying its scoring function. We have studied the problem of mapping hypotheses to stacks when multiple-stack search algorithms are used, proposing specific mapping techniques for its application in best-first or breadth-first search algorithms. Finally, we have also described different techniques to prune the search space and to estimate the score of completing a partial hypothesis.

Additionally, we have proposed techniques to deal with large phrase-based models during the search process. These techniques include the use of a cache-memory architecture and a specific data structure to represent phrase tables.

Finally, we have formally described the problem of generating statistical phrase-to-phrase alignments between the source and the target sentences, modifying our proposed PB-SMT search algorithm for its use in this task. The proposed modifications include a new expansion algorithm, the application of smoothing techniques and the definition of a log-linear model composed of a specific set of components.

CHAPTER 5

PB-SMT EVALUATION

In this chapter we show the results of the experiments that we carried out to test the SMT techniques proposed in chapters 3 and 4. Specifically, the results obtained by the techniques to estimate phrase-based models from large corpora are shown in section 5.1, best- and breadth-first search techniques for SMT are compared in section 5.2, our proposed generalised multiple stack search algorithms are evaluated in section 5.3, the log-linear model for SMT is evaluated in section 5.4, the bisegmentation-based RF estimation technique is tested in section 5.5, the results obtained by the decoding techniques for large corpora are shown in section 5.6 and finally, the proposed phrase-based alignment generation techniques are evaluated in section 5.7.

5.1 Phrase Model Estimation from Very Large Corpora

We carried out experiments to compare the `frag_by_frag_training` algorithm proposed in section 3.4.2 with respect to the conventional estimation technique. The `frag_by_frag_training` training algorithm is designed to train phrase-based models from very large corpora. For this purpose, the estimation is done in a *fragment-by-fragment* fashion. The experiments measure both the spatial and temporal costs of the estimation process. This is done in order to quantify the amount of memory which is saved by means of the `frag_by_frag_training` algorithm and the overhead introduced by its use.

Table 5.1 shows the spatial cost in GBytes and the temporal cost in seconds that have both the estimation from the whole corpus and the fragment-by-fragment estimation. In all cases, the phrase-based models were obtained by means of the THOT toolkit presented in Appendix B of this thesis. All the experiments presented in this section have been executed on a PC with a 2.60 Ghz Intel Pentium 4 processor with 2GBytes of memory. The experiments were executed on the English-Spanish Europarl corpus (see section 1.10.2 for a detailed description), ranging from a maximum phrase size of 2 to 8. The experimentation was not extended to additional language pairs because in this case, the language pair under consideration does not qualitatively affect the results (the time and spatial costs for both estimation techniques only depend on the number of phrase pairs that compose the model and we did not observe significant differences in this aspect for the different language pairs).

Table 5.1: Statistics of both conventional estimation and fragment-by-fragment estimation for the English-Spanish Europarl corpus and different values of the maximum phrase size. The statistics include the time in seconds and the main memory size in GBytes required by the estimation process.

	Conventional estimation		**Fragment-by-fragment estimation**	
m	**Time (s)**	**Size (GB)**	**Time (s)**	**Size (GB)**
2	2 266	0.11	2 336	0.12
4	6 034	0.66	5 848	0.12
6	10 757	1.47	10 234	0.12
8	-	$>$2	17 089	0.12

As can be seen in Table 5.1, the memory requirements of the conventional estimation are higher than 2 GBytes when the maximum phrase size is equal to 8. Because of this, such an estimation may not be feasible in 32-bits machines depending on which operating system is used. In contrast, fragment-by-fragment estimation has a fixed cost that is equal to 0.12 GBytes. This value is the maximum amount of memory that is assigned to the sorting algorithm and can be decreased at the expense of an increase in the time needed to perform the sort.

With regard to the time cost of the algorithms, it should be noted that fragment-by-fragment estimation can be even faster than conventional estimation for great values of the maximum phrase length. As explained in section 3.4.2, fragment-by-fragment estimation introduces time overhead because of the necessity of sorting the phrase counts. However, the time needed to store and update the counts of each phrase pair depends on the size of the model. This size is smaller if the estimation is carried out for small fragments of the corpus.

5.2 Best- versus Breadth-First Search

We carried out experiments to compare the performance of the best-first search algorithm for SMT described in section 4.2.2 with that of the breadth-first search algorithm presented in section 4.2.5. The results were obtained using three different corpora of increasing complexity, including the EuTrans-I, Xerox and Europarl corpora (see section 1.10 for more details). In all the experiments, we obtained translations from Spanish to English (experiments using additional language pairs were executed, yielding very similar results).

Both the best- and the breadth-first search algorithms used the mapping function given by Equation (4.10) to assign hypotheses to stacks (which uses J stacks, one for each possible number of aligned source positions). The log-linear model was instantiated as follows: a standard backoff language model estimated by means of the SRILM toolkit was used to implement h_1; the source sentence length model, h_2, was implemented by means of a set of normal distributions; inverse and direct standard phrase-based models generated by means of the THOT toolkit were used to implement h_3 and h_4, respectively (maximum phrase length was set to 7 words); the target phrase length model, h_5, was implemented by means of a uniform distribution and finally, the source phrase length and the distortion models, h_6 and

h_7, respectively, were implemented using geometric distributions. Table 5.2 summarises the information of the log-linear model described above. We used default values for the weights of the log-linear model.

Table 5.2: Description of the log-linear model used in the experiments.

Feature	Model	Implementation
h_1	Language model	Standard backoff language model (SRILM)
h_2	Source sentence length model	Set of normal distributions
h_3	Inverse phrase model	Standard phrase model (THOT)
h_4	Direct phrase model	Standard phrase model (THOT)
h_5	Target phrase length model	Uniform distribution
h_6	Source phrase length model	Geometric distribution
h_7	Distortion model	Geometric distribution

All the experiments presented in this section were executed on a PC with a 2.40 Ghz Intel Xeon processor with 1GB of memory.

5.2.1 Assessment Criteria

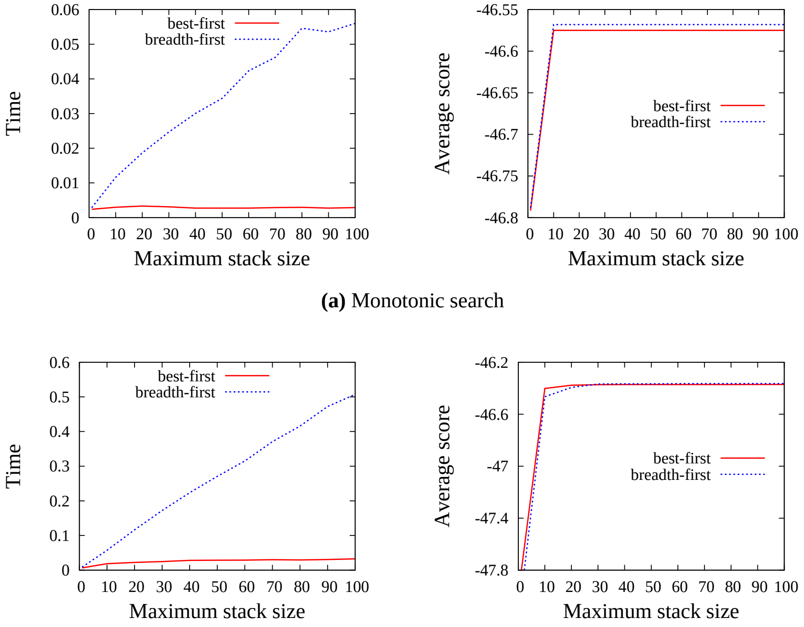
The evaluation of the techniques presented in this section was carried out measuring the time cost per sentence and the average score per sentence. The average score per sentence for a test corpus is obtained by dividing the score of the log-linear model assigned to each translation hypothesis by the total number of sentences that compose the test corpus. The time cost per sentence and the average cost per sentence were measured ranging the value of the maximum stack size parameter from 1 to 100.

We have chosen the average score per sentence as an evaluation criterion because we are interested in comparing the effectiveness of two decoding algorithms that share the same scoring function (both the best- and the breadth-first search algorithms use a log-linear model with the same components and the same weights for each component). Since the goal of the decoding process is to obtain the translation hypothesis of highest score for each source sentence, we can tell that one decoding algorithm is better than another for a given test corpus if it obtains a higher average score per sentence. In addition to this, the time cost that is required to achieve a certain value of the average score is also important to judge the effectiveness of the search process.

In the following sections, we will not report the BLEU measure obtained by the search algorithms. This is because the BLEU measure does not always correlate with the average score per sentence. In addition to this, obtaining translation quality results requires to tune the weights of the log-linear models by means of the MERT algorithm. The main disadvantage of performing weight adjustment in our experimentation setting is that the average score per sentence is no longer comparable for the two search algorithms (since MERT will return a different set of weights for each one). As it will be shown below, comparing the average score per sentence allows us to achieve a better understanding of the advantages and disadvantages of each of the search algorithms. In any case, in the experiments that we carried out (without weight adjustment), the two search algorithms produced very similar BLEU results for a given value of the average score per sentence. We will show translation quality results in section 5.4.

5.2.2 EuTrans-I Experiments

We carried out experiments to compare the best- and breadth-first search algorithms using the EuTrans-I test corpus. Figures 5.1a and 5.1b shows the obtained results for monotonic and non-monotonic search, respectively. For each figure, average translation time per sentence (left) and average score per sentence (right) as a function of the maximum stack size are shown.



(a) Monotonic search

(b) Non-monotonic search

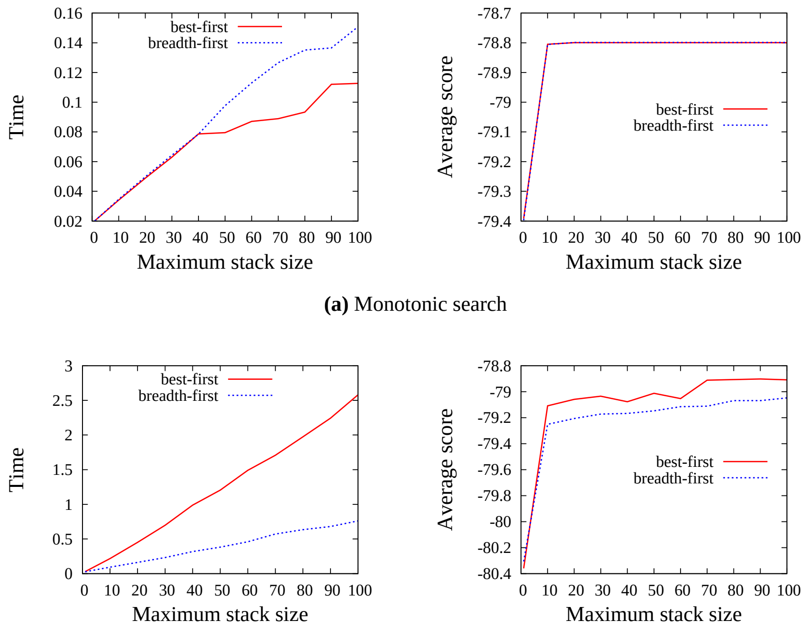
Figure 5.1: Best- versus breadth-first search comparison executed on the EuTrans-I test corpus. Plots show average translation time per sentence (left) and average score per sentence (right) as a function of the maximum stack size when performing monotonic and non-monotonic translation.

According to the figures, best-first search has a lower time cost than breadth-first search for both monotonic and non-monotonic search. As was stated in Chapter 4, the complexity of the best-first search algorithm cannot be bounded by the complexity of the breadth-first search algorithm. However, if the statistical models involved in the translation process have a low perplexity, they can accurately guide the search, reducing the time cost of the algorithm with respect to that of the breadth-first search algorithm (see sections 4.2.4 and 4.2.5 for a more detailed explanation). Since the EuTrans-I corpus has a very low complexity, the behaviour of the time cost is exactly what we expected. It is also worth noticing that the time cost of the best-first algorithm remains constant when we increase the value of the maximum stack size parameter. By contrast, and according to the complexity expression given in section 4.2.5, the time cost of the breadth-first search algorithm grows linearly with respect to maximum stack size.

Regarding the average score per sentence, there are no significant differences between the two search algorithms. It should be noted that the average score per sentence reaches its maximum value for small values of the maximum stack size parameter. In addition to this, non-monotonic search allows to obtain higher average score per sentence than monotonic search.

5.2.3 Xerox Experiments

We repeated the experiments presented in the previous section using the English-Spanish test set of the Xerox corpus. Figure 5.2a shows the results for monotonic search and Figure 5.2b shows those for non-monotonic search. Again, average translation time per sentence (left) and average score per sentence (right) as a function of the maximum stack size are shown for each figure.



(a) Monotonic search

(b) Non-monotonic search

Figure 5.2: Best- versus breadth-first search comparison executed on the English-Spanish test set of the Xerox corpus. Plots show average translation time per sentence (left) and average score per sentence (right) as a function of the maximum stack size when performing monotonic and non-monotonic translation.

According to the results presented in the figures, the time cost of best-first search is lower than that of the breadth-first search only for monotonic search. These results contrasts with the results obtained for the EuTrans-I corpus. As was explained above, the time complexity of the best-first search algorithm depends on the complexity of the translation task. The Xerox corpus is more complex than the EuTrans-I task, and non-monotonic search significantly increases the size of the search space.

Regarding the average score per sentence, it is very similar for both search algorithms when performing monotonic search. By contrast, non-monotonic best-first search obtains higher average score than non-monotonic breadth-first search. This is due to the more aggressive pruning of the stacks executed by the breadth-first search algorithm. In addition to this, it should be noted that in this case, non-monotonic search requires higher values of the maximum stack size parameter to obtain the same average score as monotonic search, due to the greater size of the search space.

5.2.4 Europarl Experiments

Finally, we compared best- and breadth-first search using the Spanish-English test set of the Europarl corpus. Figure 5.3a shows the monotonic results and Figure 5.3b shows the non-monotonic results. Each figure represents average translation time per sentence (left) and average score per sentence (right) as a function of the maximum stack size.

As can be seen in the figures, the time cost per sentence of the best-first search algorithm is significantly higher than that of the breadth-first search algorithm. Again, this was the expected behaviour, since the Europarl corpus is by far the most complex task of the three tasks used to carry out these experiments.

The behaviour of the average score per sentence is the same as that observed for the Xerox corpus: on the one hand, best-first search obtains higher average scores than breadth-first search for non-monotonic translation; on the other hand, non-monotonic translation requires higher values of the maximum stack size parameter to obtain the same score as monotonic translation.

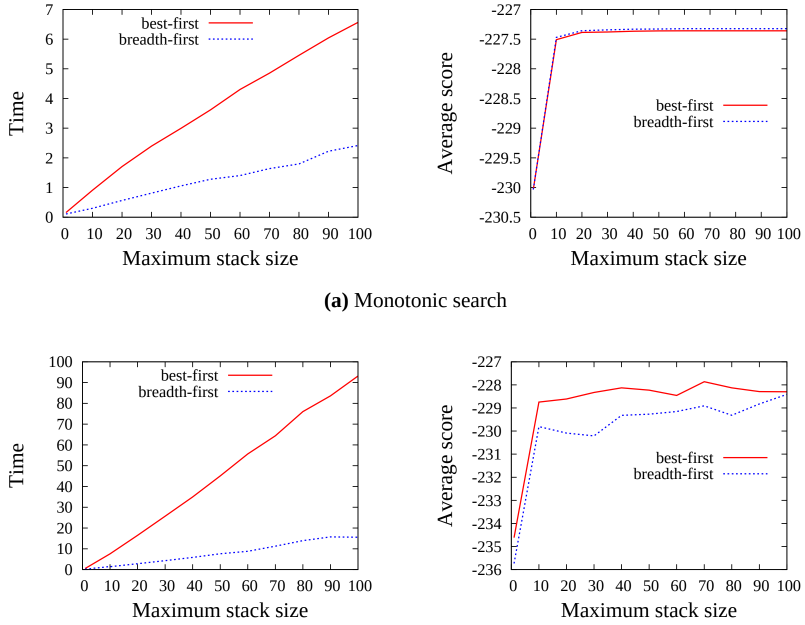
5.3 Generalised Multiple-Stack Search

In this section we report results to evaluate the performance of the generalised best- and breadth-first multiple-stack algorithms described in Chapter 4. The experiments were executed on the same corpora used in section 5.2: the EuTrans-I, Xerox and Europarl corpora. The log-linear model used in the experiments was instantiated following the configuration given in Table 5.2.

All the experiments presented in this section were executed on a PC with a 2.40 Ghz Intel Xeon processor with 1GB of memory.

5.3.1 Best-First Search Experiments

We carried out experiments to test the generalised best-first multiple stack algorithms described in section 4.2.6. For this purpose, we measured the time cost per sentence and the



(a) Monotonic search

(b) Non-monotonic search

Figure 5.3: Best- versus breadth-first search comparison executed on the Spanish-English test set of the Europarl corpus. Plots show average translation time per sentence (left) and average score per sentence (right) as a function of the maximum stack size when performing monotonic and non-monotonic translation.

average score per sentence as a function of the granularity (G) parameter (the granularity parameter determines the number of stacks and the maximum stack size used by the search algorithm). The experiments were obtained varying the value of G (ranging from 0 to 10) and the maximum number of hypotheses, L_a, that the algorithm is allowed to store for all used stacks (ranging from 2^8 to 2^{11}). This is basically the same assessment criteria described in section 5.2.1 for comparing best- and breadth-first search (the only difference is that the maximum stack size limitation is replaced by the granularity parameter).

EuTrans-I Experiments

Figure 5.4 shows the results of the experiments that we carried out for the EuTrans-I test corpus. Specifically, two plots are shown in the figure: the average time per sentence (left) and the average score (right) as a function of the granularity parameter. As can be seen, the bigger the value of G the lower the average time per sentence. For values of G greater than 6 (keeping fixed the value of L_a) the average time per sentence decreases very slightly. This is due to the fact that the number of stacks determined by G is very high and the search algorithm starts to spend more time to decide which hypothesis is to be expanded.

With respect to the average score, the maximum value is obtained for $G = 4$. Values of the G parameter above or below 4 decreased the average score. Low values of G decrease the number of stacks used by the search algorithm. This reduces the accuracy of the stack pruning, since hypotheses having very different alignment vectors are stored into the same stack. By contrast, high values of G increase the number of stacks used by the algorithm. Taking into account that the value of L_a is fixed, then the maximum stack size can take very low values and thus the "optimal" hypothesis can easily be pruned.

Finally, it is worthy of note that the best obtained score using the proposed generalised best-first search algorithms was worse than that obtained with the standard best-first search algorithm (see Figure 5.1). This suggests that the generalised best-first search algorithms execute a less efficient stack pruning than the standard algorithm. It should be noted that the algorithm may assign hypotheses with a different number of aligned words to the same stack (see Table 4.1 for an example). By contrast, the standard best-first search algorithm always assigns hypotheses with a different number of aligned words to different stacks. We think that this can be one of the underlying reasons that explain the obtained results.

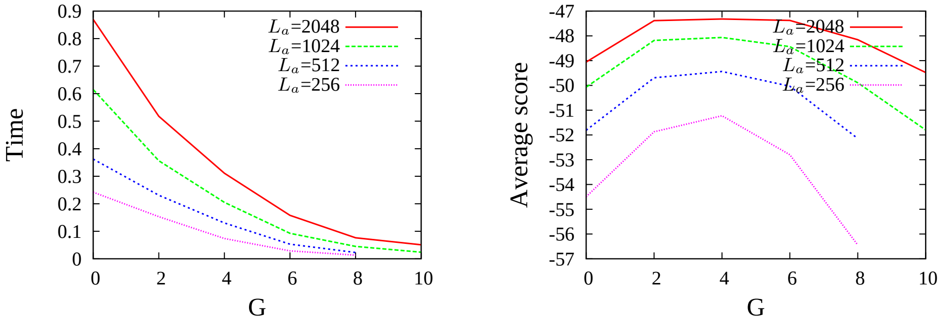


Figure 5.4: Generalised multiple-stack search experiments executed on the EuTrans-I test corpus. Plots show average translation time (left) and average score (right) per sentence as a function of the granularity (G) parameter. Each curve was obtained using different values of the maximum number of hypotheses stored by the search algorithm (L_a).

Xerox Experiments

We also carried out experiments using the English-Spanish test set of the Xerox corpus. The obtained results are shown in Figure 5.5. Again, the plots represent the average time per sentence (left) and the average score (right) as a function of the G parameter.

According to the figure, the behaviour of the time cost was similar to that observed for the EuTrans-I corpus. Specifically, values of the G parameter below or equal to 6 allowed us to significantly reduce the time cost per sentence. By contrast, very slight improvements were obtained for values of the G parameter above 6.

Regarding the average score per sentence, its maximum value was obtained for values of G between 6 and 8. The results were different to those obtained for the EuTrans-I corpus,

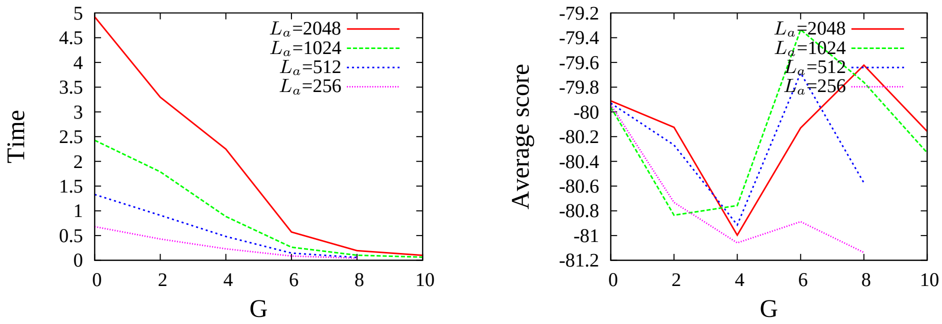


Figure 5.5: Generalised multiple-stack search experiments executed on the English-Spanish test set of the Xerox corpus. Plots show average translation time (left) and average score (right) per sentence as a function of the granularity (G) parameter. Each curve was obtained using different values of the maximum number of hypotheses stored by the search algorithm (L_a).

since the average score for $G = 0$ was higher than those obtained for values of G between 1 and 4. In addition to this, increases in the maximum number of hypotheses (L_a) did not always produce better average scores (specifically for $L_a = 2\,048$). This suggests that the efficiency of the mapping function is negatively influenced by the complexity of the translation task. One difference between the EuTrans-I and the Xerox corpora is that the Xerox corpora has a greater average sentence length. Generalised best-first search uses the same limitation for the maximum number of hypotheses that can be stored without taking into account the length of the sentence to be translated. This circumstance may be negatively affecting the average score.

Again, the best obtained score using the proposed generalised best-first search algorithms was not better than that obtained with the standard best-first search algorithm (see Figure 5.2). This confirms our previous intuition that generalised best-first search performs a less efficient stack pruning.

Europarl Experiments

We repeated the experiments presented above using the Spanish-English test set of the Europarl corpus. The obtained results are shown in Figure 5.6.

Regarding the time cost of the algorithm, the results were very similar to those obtained for the EuTrans-I and the Xerox corpora (with the exception of the result obtained for $G = 2$ and $L_a = 2048$). Again, significant improvements in the time cost were observed for values of the G parameter below 6.

With respect to the average score per sentence, the obtained results were even poorer than those obtained for the Xerox corpus. The best average score was obtained for $G = 0$ in all cases. In addition to this, increases in the L_a parameter did not produce better average scores. According to these observations, we can conclude that generalised best-first search does not scale well with complex corpora.

Finally, the best obtained score using generalised best-first search was again lower than that of standard best-first search.

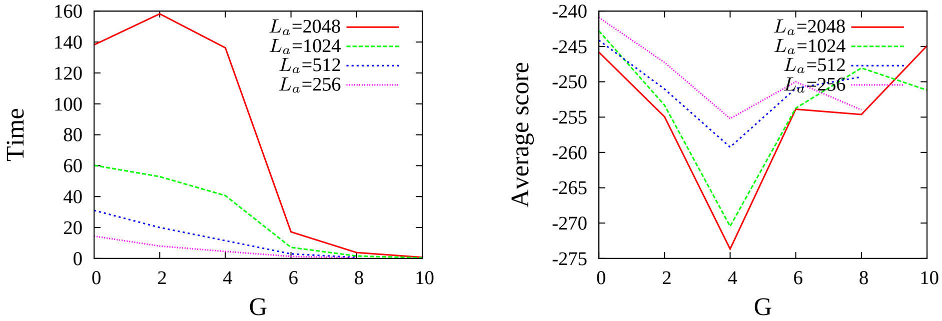


Figure 5.6: Generalised multiple-stack search experiments executed on the Spanish-English test set of the Europarl corpus. Plots show average translation time (left) and average score (right) per sentence as a function of the granularity (G) parameter. Each curve was obtained using different values of the maximum number of hypotheses stored by the search algorithm (L_a).

5.3.2 Breadth-First Search Experiments

We carried out experiments to test the generalised breadth-first multiple stack algorithms described in section 4.2.6. For this purpose we evaluated the performance of different functions to map hypotheses to stacks. Again, we are interested in the time cost of the translation as well as in the average score per sentence. More specifically, we will represent the time cost per sentence as a function of the average score. This slightly differs from the assessment criteria defined in previous sections, where the x axis was defined as the value of the maximum stack size parameter or as the granularity parameter. In this case, two search algorithms with the same value of the maximum stack size parameter and different mapping functions are not directly comparable, since they will use a different number of stacks. To solve this problem, given a specific mapping function, we obtained the time cost and the average score per sentence for different values of the maximum stack size parameter. Once all this measures were obtained for the different mapping functions, we represented the time cost as a function of the average score per sentence in order to compare them.

Here is a list of the mapping functions that were evaluated:

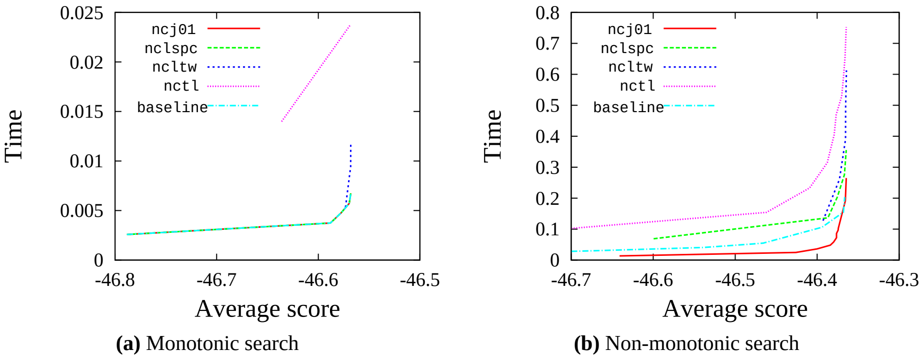
- `baseline`: this function assigns to different stacks those hypotheses with a different number of aligned words and can be formally defined by means of Equation (4.10). This function constitutes the baseline of our experiments.

- `ncj01`: the same as the `baseline` function, but those hypotheses containing non-monotonic alignments are stored in different stacks. This function is formally defined by Equation (4.11).

- `nctl`: the `baseline` function is redefined to take into account the target length of the partial hypothesis. see Equation (4.12) for a more formal definition.
- `ncltw`: the `baseline` function is modified to also include the last target word added to a given hypothesis. This function is formally defined by Equation (4.13).
- `nclspc`: this function refines the `baseline` function by taking into account the last aligned source position, see Equation (4.14) for a more formal definition.

EuTrans-I Experiments

Figure 5.7 shows the empirical results that were obtained for the EuTrans-I test corpus. As can be seen in the figure, there were no significant differences in terms of average score per sentence between the baseline function (`baseline`) and the rest of the mapping functions for both monotonic and non-monotonic search. It should be noted that `ncj01` and `nclspc` are specifically defined for its application in non-monotonic search. By contrast, such mapping functions cannot be distinguished from the `baseline` function when performing monotonic search. As was explained in section 4.2.4, a given mapping function, $\mu(h)$, defines a set of equivalence classes for the partial hypotheses. The number of stacks used by the generalised breadth-first search algorithm and thus its computational complexity are determined by the number of equivalence classes. Therefore, the time cost per sentence gives an idea of the size of the set of equivalence classes of the mapping function. According to the results shown in the figure, the `nctl` mapping function was the most time consuming one, followed by `ncltw`, `nclspc`, `ncj01` and `baseline`.

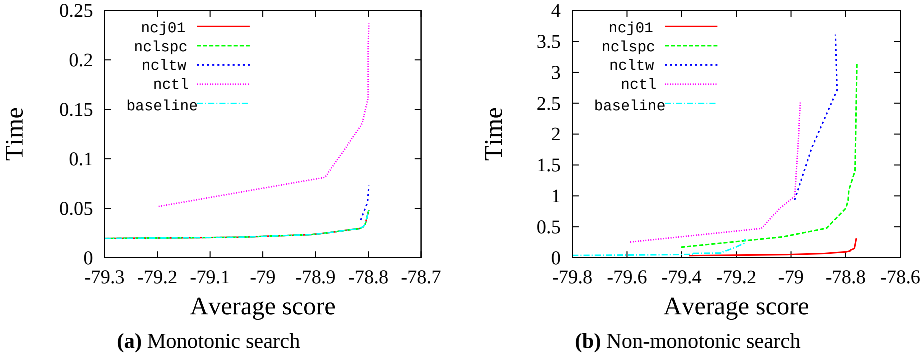


(a) Monotonic search **(b)** Non-monotonic search

Figure 5.7: Breadth-first search comparison executed on the EuTrans-I test corpus. Plots show the time cost per sentence in seconds as a function of the average score per sentence using monotonic (left) and non-monotonic (right) translation. Five different functions for mapping hypotheses to stacks were tested.

Xerox Experiments

We also carried out experiments using the English-Spanish test set of the Xerox corpus. Figure 5.8 shows the obtained results. Regarding the average score per sentence, no significant differences were observed between the different mapping functions when performing monotonic search. In contrast to this, the results using non-monotonic search show that the baseline function (`baseline`) is outperformed by the other four mapping functions. Regarding the time cost that is required to obtain a given average score per sentence, again, the most time consuming function was `ncltw`, followed by `nclspc`, `nctl`, `ncj01` and `baseline`. The `ncj01`mapping function showed the best behaviour with respect to the other functions both in terms of time cost and average score per sentence.



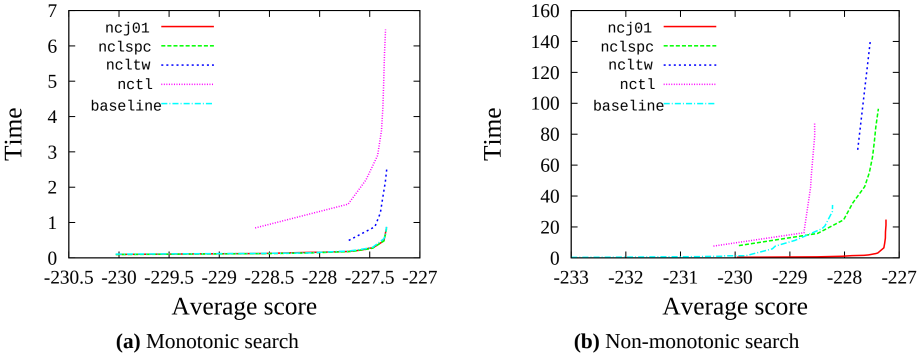
(a) Monotonic search

(b) Non-monotonic search

Figure 5.8: Breadth-first search comparison executed on the English-Spanish test set of the Xerox corpus. Plots show the time cost per sentence in seconds as a function of the average score per sentence using monotonic (left) and non-monotonic (right) translation. Five different mapping functions were compared.

Europarl Experiments

In addition to the previous experiments, we also obtained results using the Spanish-English test set of the Europarl corpus. The results are shown in Figure 5.9. Again, the differences between the five mapping functions in terms of average score per sentence were not significant when performing monotonic search. By contrast, the average score per sentence obtained by the baseline function (`baseline`) was outperformed by the `nclspc`, `nctl` and `ncj01` functions (the `nctl` function obtained the worst average score). Regarding the time cost per sentence, the results were similar to those obtained with the Xerox corpus: the most time consuming function was the `ncltw` function, followed by `nclspc`, `nctl`, `ncj01` and `baseline`. Finally, the `ncj01` mapping function showed again the best behaviour both in terms of time cost and average score per sentence.



(a) Monotonic search

(b) Non-monotonic search

Figure 5.9: Breadth-first search comparison executed on the Spanish-English test set of the Europarl corpus. Plots show the time cost per sentence in seconds as a function of the average score per sentence using monotonic (left) and non-monotonic (right) translation. Five different mapping functions were tested.

5.4 Log-linear Model Performance

We carried out experiments to test the performance of the log-linear model that was described in section 3.5.3. Specifically, we measured the impact of each feature function of the log-linear model in the translation quality. The experiments were executed on the Europarl corpus, which is the standard corpus used in the literature to report SMT results. In addition to this, we also carried out experiments on two different corpora described in section 1.10: the Xerox corpus and the EU corpus. These corpora have been extensively used to report IMT results. The translation quality was measured in terms of the BLEU score described in section 1.9.1.

5.4.1 Decoder Configuration

To obtain the results of the experiments we used a configuration of the decoding algorithm which is based on the experiments that we carried out in the previous sections. Specifically, we used a generalised breadth-first multiple-stack algorithm. This algorithm uses the mapping function given by Equation (4.11) to assign hypotheses to stacks (which is specifically designed for its use with non-monotonic search algorithms). This configuration showed a good behaviour in terms of time cost and average score per sentence. The log-linear model used by the search algorithms was instantiated following the configuration given in Table 5.2. Finally, the weights of the log-linear combination were tuned using the development sets of each corpus by means of the MERT algorithm.

5.4.2 Europarl Experiments

We carried out translation quality experiments using the Europarl test corpora, for the three language pairs, namely, Spanish-English, French-English and German-English. The decoder configuration used in the experiments was described in section 5.4.1. Table 5.3 shows the influence of the different log-linear model components in the translation quality, which was measured using the BLEU score. The experiment started with a log-linear model composed of only two components: a 4-gram language model and an inverse phrase-based model, and then, the rest of the components were incrementally added. To save computation time, non-monotonic search was applied only when the distortion model was incorporated into the log-linear model. The results were obtained using the Spanish-English test set of the Europarl corpus. As was explained in section 1.10.2, the Europarl test corpora used in this thesis are composed of in-domain and out-domain subsets. BLEU scores were obtained for such subsets and also for the whole test corpus.

Table 5.3: Influence of different models on the translation quality measured in terms of BLEU. The results were obtained using the Spanish-English Europarl test corpus (In+Out) and its in-domain (In) and out-domain (Out) subsets. A breadth-first multiple-stack algorithm was used to generate the translations. MERT was used to adjust the weights of the log-linear model.

Search	Models	In	Out	In+Out
monotone	4-gram LM + inverse phrase model	24.0	21.3	23.3
	+ sent. length model	28.0	21.9	26.3
	+ direct phrase model	29.7	24.0	28.0
	+ source phrase length model	30.1	24.4	28.3
	+ target phrase length model	30.9	25.2	29.1
non-monotone	+ distortion model	31.0	25.4	29.2

According to the results presented in Table 5.3, the incrementally added models allowed us to improve the translation quality for the Spanish-English Europarl test set and its in- and out-domain subsets. The sentence length model and the direct phrase-based model caused the greatest impact on translation quality. By contrast, non-monotonic search resulted in a very slight improvement of the BLEU score.

Table 5.4 summarises the translation quality results in terms of BLEU that were obtained when translating the Europarl test corpora from Spanish, French and German to English. The table includes monotone and non-monotone results. We used the same decoder configuration that was used in the previous experiment. Monotone results correspond to an SMT system that includes in the log-linear model all the log-linear components that were mentioned above with the exception of the distortion model (h_7).

As can be seen in Table 5.4, the best results were obtained for the Spanish to English translation direction, followed by the French-English and German-English language pairs. Again, the use of non-monotonic search allowed us to obtain very slight improvements with respect to monotonic search.

Finally, we carried out experiments to compare the translation quality obtained by our proposed translation system with that obtained by the well known Moses decoder [KHB$^+$07].

Table 5.4: BLEU results obtained with the Europarl test corpora when translating from Spanish, French and German to the English language. The translations were generated by means of a breadth-first multiple-stack algorithm. Monotonic and non-monotonic search were used. The weights of the log-linear combination were tuned via MERT.

Search	Spa-Eng	Fre-Eng	Ger-Eng
monotone	29.1±0.8	27.3±0.8	22.6±0.7
non-monotone	29.2±0.8	27.4±0.8	22.6±0.7

To obtain translation quality results with Moses, we downloaded its latest version and created an appropriate experimental setup to allow a fair comparison between the two decoders. Table 5.5 shows the BLEU results that are obtained when the Europarl test corpora is translated. Specifically, we translated from Spanish, French and German to English.

Table 5.5: Comparison of translation quality (measured according to BLEU) between Moses and our proposed translation system (BB-ALG) when translating the Europarl test corpora. 95% confidence intervals are shown.

Search	Spa-Eng	Fre-Eng	Ger-Eng
Moses	30.8±0.7	28.1±0.7	23.4±0.7
BB-ALG	29.2±0.8	27.4±0.8	22.6±0.7

As can be seen in Table 5.5, Moses significantly outperformed the results obtained by our proposed decoder for the Spanish to English translation direction. By contrast, both decoders obtained very similar BLEU results for the other two language pairs.

One possible cause for the observed differences in the results obtained by the two SMT systems could be in the statistical models used by each one. According to the Moses documentation[a], the Moses decoder uses a log-linear model composed of 7 feature functions, including a language model, inverse and direct phrase-based models, phrase and word penalties, a distortion model and inverse and direct lexical components. The set of feature functions used by our proposed decoder does not include any component that is equivalent to the lexical components used by the Moses decoder. The lexical components constitute one way to validate the quality of the translation pairs that compose the phrase tables and they are estimated using the so-called lexical weighting technique [KOM03]. To assess the influence of the lexical components in the translation quality obtained by Moses, we carried out experiments in which such components were removed. The results are shown in Table 5.6.

As can be seen in Table 5.6, the results obtained by the Moses decoder without its lexical components were very similar to those obtained by our proposed decoder. The differences were not statistically significant in all cases and our proposed decoder was able to slightly improve the Moses results for the German-English language pair. Therefore, the obtained results confirm our intuition that the lexical components used by Moses strongly contribute to the differences in translation quality observed in Table 5.5.

[a] http://www.statmt.org/moses/

Table 5.6: Comparison of translation quality (measured according to BLEU) between Moses and our proposed translation system (BB-ALG) when translating the Europarl test corpora. The log-linear model used by Moses did not include lexical components. 95% confidence intervals are shown.

Search	Spa-Eng	Fre-Eng	Ger-Eng
Moses-NoLex	29.7±0.7	27.7±0.7	22.5±0.7
BB-ALG	29.2±0.8	27.4±0.8	22.6±0.7

5.4.3 Additional Experiments

In addition to the previously presented results, we also carried out experiments with two additional corpora, namely, the Xerox corpus and the EU corpus. These corpora have been extensively used to report IMT results in the literature. Here we report translation quality measures to get an overall idea of the difficulty of these tasks (IMT results will be reported in Chapter 8). Again, we used the SMT system configuration described in section 5.4.1.

Table 5.7 shows the translation quality results measured in terms of BLEU when translating the Xerox test corpora from English to Spanish, French and German. The table shows results for both monotone and non-monotone search (monotonic results were obtained by means of a log-linear model including the whole set of feature functions except the distortion model, h_7). As can be seen, the Xerox English-Spanish language pair is the one for which the best translations can be produced. In addition to this, non-monotonic search outperformed monotonic search for the three language pairs.

Table 5.7: BLEU results when translating the Xerox test corpora from English to Spanish, French and German. Translations were generated by means of a breadth-first multiple-stack algorithm. MERT was used to tune the weights of the log-linear model.

Search	Eng-Spa	Eng-Fre	Eng-Ger
monotone	60.2±2.5	31.9±2.0	20.9±1.8
non-monotone	60.4±2.5	32.3±2.0	21.0±1.8

Additionally, we also carried out experiments to compare the translation quality obtained by our translation system with that of the Moses decoder. Table 5.8 shows the obtained results in terms of BLEU for the three test sets of the Xerox corpora. As can be seen in this table, both SMT systems obtained very similar results and the differences between them were not statistically significant. It should be noted that these results differ from those obtained for the Europarl corpus. In that situation, our proposed system obtained worse results than Moses when its lexical components were included in the log-linear model (see Tables 5.5 and 5.6). By contrast, when translating the Xerox corpora, the lexical components used by Moses did not allow it to outperform the results obtained by our proposed system.

The BLEU results that were obtained when translating the EU test corpora from Spanish, French and German to the English language are shown in Table 5.9. The table includes monotonic and non-monotonic results. In this case, the best results were obtained for the

Table 5.8: Comparison of translation quality (measured according to BLEU) between Moses and our proposed translation system (BB-ALG) when translating the Xerox test corpora. 95% confidence intervals are shown.

Search	Eng-Spa	Eng-Fre	Eng-Ger
Moses	59.6±2.4	33.7±2.0	19.3±1.6
BB-ALG	60.4±2.5	32.3±2.0	21.0±1.8

Spanish-English and the French-English language pairs. Again, non-monotonic search allowed us to obtain better results with respect to monotonic search, although the obtained improvements were smaller than those obtained for the Xerox test corpora.

Table 5.9: Translation quality results measured in terms of BLEU when translating the EU test corpora from Spanish, French and German to the English language. A breadth-first multiple-stack algorithm was used to generate the translations. The weights of the log-linear combination were adjusted via the MERT algorithm.

Search	Spa-Eng	Fre-Eng	Ger-Eng
monotone	44.3±1.9	47.3±2.0	37.6±1.9
non-monotone	44.5±1.9	47.5±1.9	37.6±1.9

We also performed translation quality experiments to compare our proposed translation system with the Moses decoder. The BLEU results are shown in Table 5.10. Again, the differences between the two decoders were not statistically significant.

Table 5.10: Comparison of translation quality (measured according to BLEU) between Moses and our proposed translation system (BB-ALG) when translating the EU test corpora. 95% confidence intervals are shown.

Search	Spa-Eng	Fre-Eng	Ger-Eng
Moses	44.1±1.9	46.6±1.8	37.9±2.0
BB-ALG	44.5±1.9	47.5±1.9	37.6±1.9

5.5 Bisegmentation-based RF Estimation

In this section we show the results of the experiments we carried out to evaluate the BRF (bisegmentation-based relative frequency) estimation technique for phrase-based models. This alternative estimation technique was described in section 3.3.

We evaluated our alternative estimation technique using three different corpora of ascending complexity, namely, the EuTrans-I, Xerox and Europarl corpora. To evaluate our alternative estimation technique, we measured the estimation time in seconds and the translation quality in terms of BLEU. The obtained results were compared with those of the standard RF estimation technique. The translation quality experiments were carried out using the decoder configuration described in section 5.4.1.

Table 5.11 shows the time cost in seconds of both RF and BRF estimation for the EuTrans-I, Xerox and Europarl corpora. As can be seen in the table, BRF estimation required more time to be executed in all cases. Nevertheless, the time cost of BRF estimation was affordable even for medium or large size corpora (the Xerox and Europarl corpora were trained in 0.6 and 18 hours, respectively). It is also worthy of note that no additional constraints were necessary to successfully complete the training process. This contrasts with other related proposals such as the one presented in [DGZK06], where the performance of a generative phrase-based model trained by means of the EM algorithm is compared with that of the standard estimation method. In that work, the training process is constrained such that only those bisegmentations that can be obtained using consistent phrase pairs are considered. This is exactly the key aspect of BRF estimation. According to the authors of the work presented in [DGZK06], a maximum phrase length of three words was imposed during the training. In addition to this, different factors caused their estimation algorithm to rule out approximately 54% of the training set. The proposal presented here does not have such disadvantages.

Table 5.11: Comparison of estimation time cost in seconds using RF and BRF estimation when translating from English to Spanish with the EuTrans-I, Xerox and Europarl corpora.

Estimation	EuTrans-I	Xerox	Europarl
RF	8	243	9 780
BRF	16	2 089	64 868

Table 5.12 shows the BLEU scores that were obtained when using RF and BRF estimation. The experiments were carried out using the EuTrans-I, Xerox and Europarl corpora. As can be seen in the table, RF estimation obtained slightly better results than BRF estimation in all cases. However, the differences between the two estimation techniques were not statistically significant.

As was mentioned above, our estimation proposal is strongly related to the one presented in [DGZK06]. The results obtained by their alternative estimation technique also underperformed the results obtained by the standard estimation technique. In spite of this, their technique yielded higher likelihood values than the conventional estimation technique. According to the authors, the observed increases in the data likelihoods are obtained by determinising phrase translation probabilities, i.e. the estimation algorithm obtains sharply peaked conditional distributions, overfitting the training data. We think that the conclusions explained in [DGZK06] are also appliable here due to the similarities of the two proposed estimation techniques. In our case, we also observed higher data likelihoods for BRF estimation with respect to RF estimation, but the translation quality results were worse.

According to the presented results, we have empirically demonstrated that BRF estimation can be efficiently used to estimate phrase models from large corpora. Unfortunately, the proposed estimation technique did not allow us to improve the results with respect to the standard estimation technique. In spite of this, we think that the acceptable time cost of BRF

Table 5.12: Comparison of translation quality (measured according to BLEU) using RF and BRF estimation when translating from English to Spanish with the EuTrans-I, Xerox and Europarl corpora. 95% confidence intervals are shown.

Estimation	EuTrans-I	Xerox	Europarl
RF	93.3±0.4	60.4±2.5	29.2±0.8
BRF	93.2±0.4	60.2±2.5	28.7±0.8

estimation makes it interesting as a starting point to implement more sophisticated estimation techniques, as was explained in section 3.3.4.

5.6 Efficient Decoding with Large Phrase-Based Models

To evaluate the performance of the decoding technique for large phrase-based models described in section 4.3, we carried out a series of experiments using the Europarl corpus. For this purpose, we have estimated a phrase model imposing a maximum phrase size of 7 words.

The access to the model parameters during the decoding process can be viewed as a two-stage process that is repeated for each sentence to be translated. First, the target phrase translations for each phrase of the source sentence are retrieved. Second, the translation probabilities for the bilingual pairs are accessed.

The efficiency of the cache-inspired proposed architecture depends on the number of disk accesses (or cache misses) occurred during the first stage of the translation process of each sentence. Once the first stage of the translation has concluded, no more disk accesses will be necessary since all the required entries of the model will be stored in the main memory. The number of cache misses allows us to compute the rate of cache misses as the number of cache misses divided by the total number of accesses to the model during the whole translation process. In order to determine the efficiency of the proposed architecture it is also interesting to measure the time cost per translation and the time cost per model query, which are closely related to the rate of cache-misses.

Table 5.13 shows the time in seconds required to retrieve the translations for all the phrases that compose the Spanish-English Europarl test set for different values of the α parameter described in section 4.3.1. All the experiments presented in this section have been executed on a PC with a 2.60 Ghz Intel Pentium 4 processor with 2GBytes of memory. The table also shows the number of phrase pairs stored in memory, the number of disk accesses and the time overhead caused by these accesses. As can be observed, the retrieval of the translations from disk introduces significant overhead; however, this overhead can be reduced by increasing the value of the α parameter. It is worth noticing that a great decrease in the rate of cache misses can be achieved for small values of α.

In order to quantify the total locating time, we have translated the Spanish-English Europarl test set by means of a decoding algorithm. Again, such decoding algorithm was instantiated following the SMT system configuration given in section 5.4.1. Since translation quality was not important in our experiments and the MERT algorithm is time consuming, we used default values for the weights of the log-linear combination.

Table 5.13: Number of phrases, disk accesses, total time (in secs), and disk overhead required to retrieve the translations for the phrases of the Spanish-English Europarl test set, ranging over the value of α.

	Phrases	**DiskAccesses**	**Time**	**DiskOvh**
$\alpha = 100$ (Baseline)	31 227 305	0 / 0.0%	8.6	0
$\alpha = 0$	0	559 336 / 100.0%	651.2	649.7
$\alpha = 1$	312 244	462 277 / 82.6%	633.7	625.1
$\alpha = 10$	3 122 708	370 419 / 66.2%	545.6	535.4
$\alpha = 20$	6 245 443	349 727 / 62.5%	525.7	515.4
$\alpha = 40$	12 490 908	288 282 / 51.5%	368.8	358.2
$\alpha = 60$	18 736 374	219 763 / 39.2%	272.4	262.3
$\alpha = 80$	24 981 839	146 141 / 26.1%	175.2	170.2
$\alpha = 99$	30 915 031	71 885 / 12.8%	96.4	86.8

In the experiment, cache models with α equal to 100 (our baseline) and with α equal to 10 were used. Table 5.14 shows the number of queries to the model, the percentage of cache misses, the total locating time, the locating time per sentence, the locating time per query and the translation quality measured in terms of BLEU.

As can be observed, the proposed system needs 0.2 seconds to translate a sentence (five translated sentences per second). Although the time requirements are higher than those of the baseline system, we think that they are acceptable for regular translator users. In addition to this, it could be interesting to compare the time cost per query for the two systems. The latency of modern hard drives is measured in milliseconds, whereas the latency of main memory is measured in microseconds or tenths of microseconds [HP03]. As it is shown, the low rate of cache misses makes possible that the average time cost per query of our system becomes close to the latency of main memory.

It should be noted that the reported results corresponds to a monotonic search algorithm. If non-monotonic search is performed, then even better results can be expected, since the number of possible phrase translations to be retrieved for each source sentence remains the same while the number of queries to the model during the translation significantly increases. This increase of model queries is due to the greater complexity of the non-monotonic expansion algorithm with respect to that of the monotonic one (see sections 4.2.2 and 4.2.3).

Regarding the space requirements, the proposed system only needs one tenth of the memory required by the baseline system to translate the test corpus.

5.7 Phrase-Level Alignment Generation

We carried out experiments to test the phrase-level alignment generation techniques described in section 4.4. The experiments consisted in obtaining phrase-to-phrase alignments between pairs of sentences following a set of different smoothing techniques. The test set was taken from the shared tasks in word alignments developed in HLT/NAACL 2003 [MP03]. This shared task involved four different language pairs, but we only used English-French in our

Table 5.14: Number of queries, % of cache misses, total, per sentence and per query locating time (in secs.) required by all model queries when translating the Spanish-English Europarl test set ($\alpha = 100$ constitutes the baseline system). A breadth-first multiple-stack algorithm was used to generate the translations. Such algorithm implemented monotonic search. Default log-linear weights were used.

	Queries	%CMisses	Time	Time/sent	Time/query	BLEU
$\alpha = 100$	227 694 848	0	94.6	0.03	4.1e-07	26.8
$\alpha = 10$	227 694 848	0.16	636.4	0.2	2.8e-06	26.8

experiments. A subset of the Canadian Hansards corpus was used in the English-French task. The exact details of the corpus used in the experimentation were described in section 1.10.3.

5.7.1 Aligner Configuration

The results were obtained by means of a breadth-first multiple-stack algorithm. Such algorithm used non-monotonic search and the mapping function given by Equation (4.11) to assign hypotheses to stacks. Inverse and direct standard phrase-based models generated by means of the THOT toolkit were used to implement h_1 and h_2. With respect to the probability distribution used to model feature functions h_3 (target phrase length model) and h_4 (source phrase length model), we show the results corresponding to the use of a uniform distribution for h_3 and a geometric distribution for h_4, since such choices led to better results. As was mentioned in section 6.2.2, the use of a uniform distribution for h_3 penalises the length of the segmentation and the use of a geometric distribution for h_4 makes it possible to establish a relationship between the length of source and target phrases (the use of a Poisson distribution also worked well). Finally, the distortion model, h_5, was implemented by means of a geometric distribution.

5.7.2 Assessment Criteria

We were interested in evaluating the quality of the phrase-to-phrase alignments obtained with the different phrase alignment smoothing techniques that we proposed. Unfortunately, there does not exist a gold standard for phrase alignments, so we needed to refine the obtained phrase alignments to word alignments in order to compare them with other existing word alignment techniques.

Taking these considerations into account, we proceeded as follows: Given a pair of sentences to be aligned, we first aligned them at phrase level, obtaining a phrase-to-phrase alignment. Afterwards, we obtained a word-to-word IBM1 alignment for each pair of aligned phrases. Finally, these “intra-phrase” word alignments were joined, resulting in a word level alignment for the whole sentence. We could thus make a fair comparison of the proposed smoothing techniques with the ones presented in the HLT/NAACL 2003 shared task.

To evaluate the quality of the obtained final alignments, different measures were taken into account: precision, recall, F-measure, and alignment error rate. Given an alignment A and a reference alignment G (both A and G can be split into two subsets A_S,A_P and G_S, G_P, respectively representing *Sure* and *Probable* alignments) precision (P_S,P_P), recall (R_S,R_P), F-measure (F_S,F_P) and alignment error rate (AER) were computed (see section 1.9.2 for a detailed description of these measures).

The above described evaluation measures were applied to two different sets of evaluation depending on whether the alignments with the null word were removed or not (see [MP03] for more details):

- NULL alignments: given a word alignment a_1^J for a pair of sentences (f_1^J, e_1^I), if a word f_j $(j \in \{1 \ldots J\})$ is not aligned with any e_i $(i \in \{1 \ldots I\})$, or viceversa, that word is aligned with the null word.

- NO-NULL alignments: null alignments are removed, from the test set and from the obtained alignments.

5.7.3 Alignment Quality Results

In Table 5.15, the alignment quality results using different phrase-to-phrase alignment smoothing techniques are presented, for NO-NULL and NULL alignments. It is worth mentioning that the figures for *Sure* alignments are identical for NO-NULL and NULL alignments. In the table the first row shows the baseline, which consists of the results obtained using a maximum likelihood estimation (ML) without smoothing. The rest of the rows corresponds to different estimation techniques combined with linear interpolation (LI), backoff (BO) or log-linear interpolation (LL).

For the NO-NULL alignment experiment, significant improvements in all alignment quality measures were obtained for all the smoothing techniques compared with the baseline. The baseline system results were worse due to the great number of times in which the segmentation of a sentence pair could not be completed due to coverage problems (in our experiments, 86.5% of the test pairs presented this problem); in such situations, the baseline system aligned all the source words of the source sentence with all the target words of the target sentence. Finally, it is worth pointing out that the use of the LEX distribution produced improvements in the alignment quality with respect to those situations in which such distribution was not used. These better results are obtained due to improved assignment of probabilities to unseen events. In addition to this, linear interpolation and backing-off obtained better results than log-linear interpolation. This is precisely the expected behaviour, as was explained in section 4.4.2. In addition to this, we observed that ML, GT and SD estimation worked better than AD and KN estimation in some cases, e.g. when those estimators were combined with the LEX distribution via linear interpolation. More research is needed to determine the exact cause of the observed differences.

It is also worth mentioning that despite the fact that the phrase alignment techniques proposed here are not specifically designed to obtain word alignments, all the results are competitive with those presented in [MP03]. In the table, the best results for each column are highlighted showing that GT+LEX$_{\text{BO}}$ obtained the best results.

Regarding the results for the NULL alignment experiment, there were small relative improvements in 9 out of 16 smoothing techniques compared with the baseline. The differences between these results and those for NO-NULL alignment experiment are due to the fact that the baseline generated a lot of alignments in which all words were aligned with all words due

Table 5.15: Comparative alignment quality results (in %) using different smoothing techniques for NO-NULL and NULL alignments. A breadth-first multiple-stack algorithm was used to generate the alignments. Such algorithm implemented non-monotonic search. Default log-linear weights were used. Best results are shown in bold.

	NO-NULL & NULL			**NO-NULL**				**NULL**			
Smooth.	P_S	R_S	F_S	P_P	R_P	F_P	AER	P_P	R_P	F_P	AER
ML	64.4	76.6	70.0	77.5	28.3	41.5	20.0	55.1	29.4	38.3	36.4
GT	71.6	79.6	75.4	87.8	27.0	41.3	14.8	52.4	28.8	37.2	39.1
AD	69.1	77.6	73.1	84.0	26.6	40.4	17.1	51.1	28.1	36.3	40.2
KN	68.6	77.9	74.0	83.7	26.7	40.4	17.2	51.5	28.2	36.4	39.9
SD	71.7	79.1	75.2	87.7	27.0	41.2	15.1	52.3	28.8	37.1	39.4
ML+LEX$_{\text{LI}}$	**72.2**	86.4	78.0	**91.3**	29.9	45.1	9.7	59.5	31.5	41.2	31.8
GT+LEX$_{\text{LI}}$	71.0	86.3	77.9	91.2	29.9	45.0	9.8	59.5	31.5	41.2	31.9
AD+LEX$_{\text{LI}}$	71.5	81.5	76.2	89.4	27.9	42.6	12.9	54.4	29.6	38.4	37.0
KN+LEX$_{\text{LI}}$	71.6	82.1	76.4	89.5	28.1	42.9	12.5	54.9	29.8	38.6	36.4
SD+LEX$_{\text{LI}}$	71.0	86.1	77.8	91.2	29.9	45.0	9.9	59.4	31.5	41.1	32.0
GT+LEX$_{\text{BO}}$	71.0	86.4	77.9	91.2	**30.0**	**45.2**	**9.6**	**59.8**	**31.7**	**41.4**	**31.5**
SD+LEX$_{\text{BO}}$	71.1	86.3	78.0	91.1	**30.0**	45.0	9.8	59.5	31.5	41.2	31.8
ML+LEX$_{\text{LL}}$	64.4	76.6	70.0	77.4	28.3	41.5	20.0	55.1	29.4	38.3	36.4
GT+LEX$_{\text{LL}}$	71.3	**86.8**	78.3	90.3	29.8	44.8	10.1	59.2	31.4	41.0	31.9
AD+LEX$_{\text{LL}}$	67.8	80.9	73.8	82.7	28.4	42.2	16.2	55.5	29.7	38.7	35.7
KN+LEX$_{\text{LL}}$	68.0	81.4	74.1	82.7	28.5	42.4	16.0	55.8	29.8	38.8	35.4
SD+LEX$_{\text{LL}}$	71.2	86.7	**78.2**	90.3	29.8	44.8	10.0	59.1	31.3	41.0	32.0

to coverage problems. In those situations, the IBM 1 alignment model tended to align less words with the null word than when it was applied over intra-phrase alignments derived from successful segmentations of sentence pairs. If we compare column P_P of both experiments, a significant reduction in terms of precision is obtained in the case of the NULL alignment experiment. This makes our results less competitive than those presented in [MP03] for the NULL alignment experiment.

Finally, we also carried out alignment quality experiments for the GT+LEX$_{\text{BO}}$ smoothing technique using MERT to tune the weights of the log-linear combination. The objective function to be minimised via MERT was the AER measure for both the NULL and the NO-NULL alignments (specifically, we computed the sum of the two). The results are shown in Table 5.16. As can be seen in the table, the tuning of the log-linear weights produced very similar results than those obtained by the system with default weights. We think that this is because the small size of the development set of the Hansards corpus (it is composed of only 37 sentence pairs according to the corpus statistics presented in section 1.10.3).

Table 5.16: Alignment quality results (in %) using GT+LEX$_{\text{BO}}$ smoothing for NO-NULL and NULL alignments. A breadth-first multiple-stack algorithm was used to generate the alignments. The search algorithm implemented non-monotonic search. Log-linear weights were tuned using MERT.

	NO-NULL & NULL			**NO-NULL**				**NULL**			
Smooth.	P_S	R_S	F_S	P_P	R_P	F_P	AER	P_P	R_P	F_P	AER
GT+LEX$_{\text{BO}}$	71.6	86.3	78.2	91.7	29.8	45.0	9.5	59.2	31.5	41.1	32.1

5.8 Summary

In this chapter we have tested the different training and decoding techniques for SMT that were proposed in chapters 3 and 4.

The techniques to train phrase-based models from very large corpora described in section 3.4 were tested by means of a series of experiments. The results show that the proposed techniques can be used to train phrase models for corpora of an arbitrary size without introducing a significant time overhead.

We carried out experiments to compare the performance of the best-first search algorithm for SMT described in section 4.2.2 with that of the breadth-first search algorithm presented in section 4.2.5. The obtained results show that the time cost of best-first search is lower when translating simple corpora. Moreover, best-first search obtained higher average score per sentence for a given test corpus than breadth-first search due to its less aggressive pruning of the search space. By contrast, best-first search was significantly more time consuming than breadth-first search when translating complex corpora.

Additionally, we evaluated the performance of the generalised best- and breadth-first multiple-stack algorithms described in sections 4.2.6 and 4.2.7, respectively. Generalised best-first search algorithms did not outperform the conventional best-first search algorithm. One possible reason for these results may be that the proposed algorithms store hypotheses with different number of aligned words in the same stack. By contrast, empirical results showed that generalised breadth-first search algorithms outperform conventional breadth-first search algorithms in terms of average score and time cost per sentence.

We compared the performance in terms of translation quality of the Moses decoder with that of a generalised breadth-first search decoder. The obtained results were similar when translating the test sets of the Xerox and EU corpora. By contrast, Moses outperformed the results obtained by our proposed system when translating the Europarl corpus. Nevertheless, if the lexical log-linear components of the Moses decoder were removed (our decoder does not include them), then we did not observe significant differences between the two decoders.

We also performed experiments to test our alternative BRF estimation technique for phrase-based models described in section 3.3. We empirically demonstrated by means of a series of experiments that the time cost of the estimation is affordable for corpora of different complexity. Additionally, translation quality experiments were obtained to compare BRF estimation with the standard estimation technique. Our proposed technique slightly underperformed the standard technique, but the differences were not statistically significant. Despite this, BRF estimation obtained higher likelihood values than the standard estimation technique for corpora of different complexity. One possible explanation for the observed results may be that our estimation technique overfits the training data.

Additionally, we carried out experiments to test the techniques to efficiently handle the phrase-based model parameters estimated from very large corpora proposed in section 4.3. Such techniques are based on a classic concept of computer architecture: cache memory. The results of the experiments show that the proposed cache memory architecture has a extremely low rate of cache misses, allowing a very efficient access to phrase-based model parameters.

Finally, we performed experiments to test the phrase-based alignment generation techniques proposed in section 4.4. Experimental results for a well-known shared task on word alignment evaluation have been reported. The results show the great impact of the smoothing techniques on alignment quality. Such smoothing techniques consisted of different statistical phrase-based model estimators and a lexical distribution which can be combined by means of backoff techniques, linear interpolation or log-linear interpolation. As we expected, backing-off and linear interpolation worked better than log-linear interpolation.

Part III

Interactive Phrase-Based Statistical Machine Translation

CHAPTER 6

INTERACTIVE PHRASE-BASED MACHINE TRANSLATION

6.1 Introduction

As was already introduced in section 1.8, the IMT framework constitutes an alternative to fully automatic MT systems in which the MT system and its user collaborate to generate correct translations. These correct translations are generated in a series of interactions between the IMT system and its user. Specifically, at each interaction of the IMT process, the IMT system generates a translation of the source sentence which can be partially or completely accepted and corrected by the user of the IMT system. Each partially corrected text segment, or prefix, is then used by the SMT system as additional information to generate better translation suggestions. An example of a typical IMT session was shown in Figure 1.3. For the reader's convenience, we briefly present again the statistical formalisation of IMT (the full details can be found in section 1.8).

In the IMT scenario we have to find the suffix, $\mathbf{e}_s$, that best completes the user validated prefix, $\mathbf{e}_p$ (Equation (1.34)):

$$\hat{\mathbf{e}}_s = \underset{\mathbf{e}_s}{\arg\max} \left\{ p(\mathbf{e}_s \,|\, f_1^J, \mathbf{e}_p) \right\} \tag{6.1}$$

Applying the Bayes rule, we arrive at the following expression (Equation (1.35)):

$$\hat{\mathbf{e}}_s = \underset{\mathbf{e}_s}{\arg\max} \left\{ p(\mathbf{e}_s \,|\, \mathbf{e}_p) \cdot p(f_1^J \,|\, \mathbf{e}_p, \mathbf{e}_s) \right\} \tag{6.2}$$

where the term $p(\mathbf{e}_p)$ has been dropped since it does not depend on $\mathbf{e}_s$.

Due to the similarities between the MT and the IMT frameworks, we only need to appropriately modify the search procedures of regular SMT systems to obtain the suffixes required in IMT (note that $\mathbf{e}_p\mathbf{e}_s \equiv e_1^I$).

One common implementation for IMT systems is based on the generation of word graphs (word graphs were introduced in section 4.2.10). During the interactive translation process of a given source sentence, the system makes use of the word graph generated for that sentence in order to complete the prefixes accepted by the human translator. Specifically, the system finds the best path in the word graph which is compatible with the user prefix.

The main advantages of word graph-based IMT systems is their efficiency in terms of the time cost per each interaction. This is due to the fact that the word graph is generated only once at the beginning of the interactive translation process of a given source sentence, and the suffixes required in IMT can be obtained by incrementally processing this word graph. The use of word graphs generated at the beginning of the IMT process is also the main limitation of this kind of IMT systems, since the word graph does not get automatically adapted to the new information provided by the user prefix.

Several IMT systems have been proposed in the literature. For example, in [FLL02] a maximum entropy version of IBM 2 model is used as word-based translation model. The alignment template approach to IMT is proposed in [OZN03]. In that work, a pre-computed word graph is used in order to achieve fast response times. This approach is compared with the use of a direct translation modelling [BHV+05]. An IMT approach based on stochastic finite-state transducers is presented in [CVC+04]. In that work, word graphs are also used to resolve real-time constraints. A phrase-based approach is presented in [TC06], in contrast to other existing techniques, their proposal does not use word graphs.

Recently, in [BBC+09] the IMT approach to CAT is proposed, establishing the state-of-the-art in this discipline. In this work the last three approaches mentioned above are compared.

A common problem in IMT arises when the user sets a prefix which cannot be explained by the statistical models. Under these circumstances, the suffix cannot be appropriately generated, since the system is unable to generate translations that are compatible with the user prefix. In those IMT systems that use word graphs to generate the suffixes, the common procedure to face this problem is to perform a tolerant search in the word graph. This tolerant search uses the well known concept of Levenshtein distance in order to obtain the most similar string for the given prefix (see [OZN03] for more details).

In this chapter, two novel phrase-based IMT techniques are presented. First, an IMT technique based on the generation of partial phrase-based alignments is proposed in section 6.2. This technique is inspired by the techniques to generate phrase-based alignments presented in the previous chapter and does not rely on word graphs to generate the suffixes required in IMT. Second, an alternative IMT technique based on stochastic error-correction models is proposed in section 6.3. This technique relies on word graphs or N-best lists to generate the suffixes required in IMT. To end the chapter, a brief summary of it is provided in section 6.4.

To complement the above mentioned content, Appendix C describes the main features of an IMT prototype that have been implemented using the techniques proposed in this chapter.

6.2 IMT based on Partial Phrase-Based Alignments

We present a new IMT technique which is based on the generation of *partial alignments* at phrase-level. In the proposed IMT technique, the suffix is generated following a two-stage process: first the prefix $\mathbf{e}_p$ is aligned with only a part of the source sentence f_1^J, and second, the unaligned portion of f_1^J (if any) is translated, giving the desired suffix $\mathbf{e}_s$. The generation of such partial alignments is driven by statistical phrase-based models. The problem of generating the suffix in IMT using partial phrase-based alignments is very similar to the problem of finding the best alignment between a pair of sentences described in section 4.4. Specifically, the generation of the suffixes requires the definition of a specific search algorithm as well as the application of smoothing techniques over the phrase models to appropriately assign probabilities to unseen events.

The proposed IMT system is similar to those presented in [BHV$^+$05] and [TC06]. These two techniques do not use word graphs to generate the suffixes required in IMT, as well as our proposal. The key difference between the so-called *interactive generation* strategy proposed in [BHV$^+$05] and the technique proposed here is that they use error-correcting techniques instead of smoothing techniques to assign probabilities to unseen events. Specifically, the error correcting costs are introduced as an additional weight in their log-linear model. We think that our approach is better motivated from a theoretical point of view, as it has been deeply studied and demonstrated in the field of language modelling.

The work presented in [TC06] is based on filtering the phrase table to obtain translations that are compatible with the user prefix. Since this approach seems too restrictive (phrase models always present coverage problems in complex tasks), we guess that also any sort of smoothing is taken into account, but as far as we know the exact technique that is used is not explained. Because of this, we think that the work presented in [TC06] can benefit from the techniques presented here.

The remaining part of this section is structured as follows: first, the specific search algorithm to be used in the generation of the suffixes required in IMT is described in section 6.2.1; and second, the scoring function used to assign scores to partial hypotheses is described in section 6.2.2.

6.2.1 Search Algorithm

Regarding the search algorithm to be used, we propose the use of a modified version of the generalised branch and bound search algorithm for PB-SMT described in section 4.2. Except for the scoring function (which will be studied in section 6.2.2), only the expansion algorithm has to be appropriately modified to allow the generation of partial phrase-based alignments.

The expansion process consists of appending target phrases as translation of previously unaligned source phrases of a given hypothesis. Let us suppose that we are translating the sentence f_1^J $\equiv$ “Para ver la lista de recursos”, and that the user has validated the prefix $\mathbf{e}_p \equiv$ “To view a” (interaction 1 of the IMT session given in Figure 1.3). Figure 6.1 shows an example of the results obtained by the expansion algorithm that we propose for two hypotheses h_1 and h_2.

Hypothesis h_1 has aligned the source phrase “Para ver la” (aligned phrases are noted with underlined words in Figure 6.1), appending the target phrase “To view a” to the final translation. Since for h_1, the user prefix $\mathbf{e}_p$ has already been generated, the expansion process works in the same way as the one executed in a regular translator. Let us suppose that we are aligning the source phrase $\tilde{f}$ $\equiv$“lista de recursos” given by the source positions $u = (4, 6)$. The new hypotheses h_3 and h_4 are generated by appending target phrases $\tilde{e}$ from the set $\mathcal{T}_{4,6}$ of translations for f_4^6 contained in the phrase table.

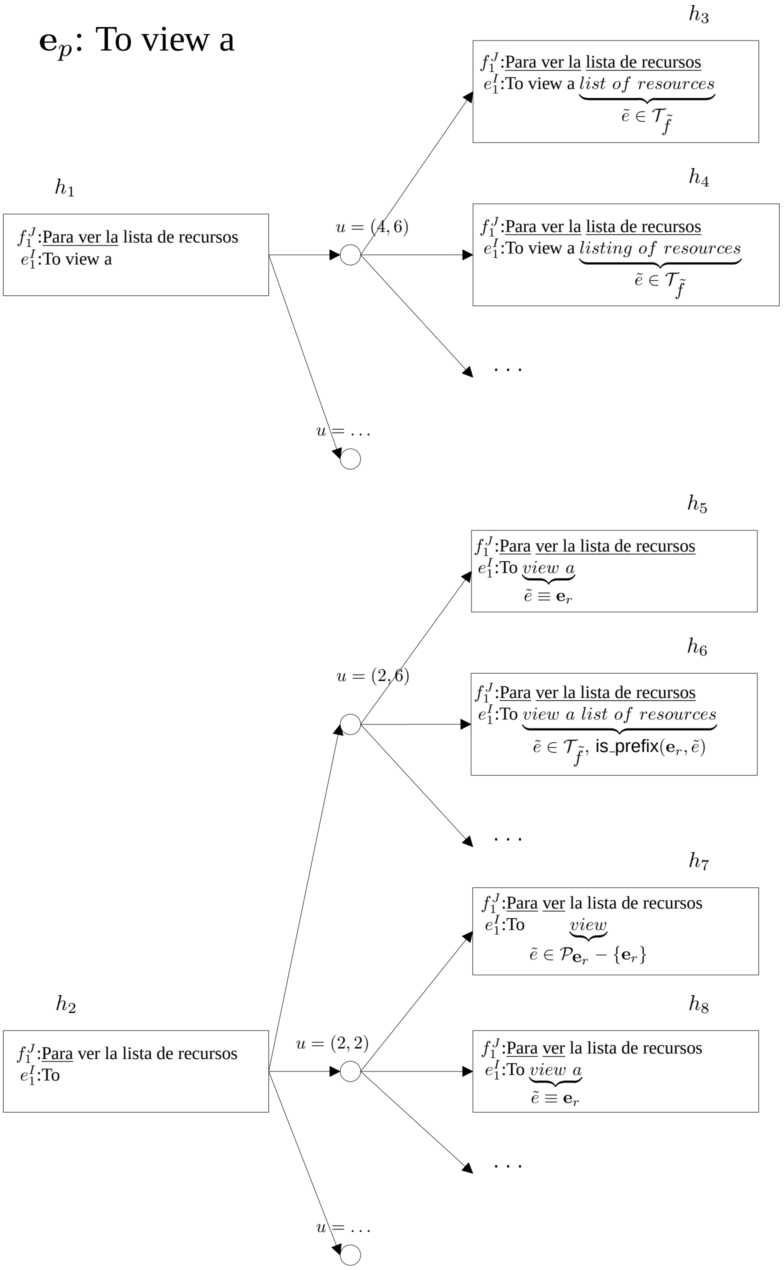

$\mathbf{e}_p$: To view a
h_1
f_1^J:Para ver la lista de recursos
e_1^I:To view a
$u = (4,6)$
$u = \ldots$
h_3
f_1^J:Para ver la lista de recursos
e_1^I:To view a $\underbrace{list\ of\ resources}_{\tilde{e} \in \mathcal{T}_{\tilde{f}}}$
h_4
f_1^J:Para ver la lista de recursos
e_1^I:To view a $\underbrace{listing\ of\ resources}_{\tilde{e} \in \mathcal{T}_{\tilde{f}}}$
...
h_5
f_1^J:Para ver la lista de recursos
e_1^I:To $\underbrace{view\ a}_{\tilde{e} \equiv \mathbf{e}_r}$
$u = (2,6)$
h_6
f_1^J:Para ver la lista de recursos
e_1^I:To $\underbrace{view\ a\ list\ of\ resources}_{\tilde{e} \in \mathcal{T}_{\tilde{f}},\ \text{is\_prefix}(\mathbf{e}_r, \tilde{e})}$
...
h_7
f_1^J:Para ver la lista de recursos
e_1^I:To $\underbrace{view}_{\tilde{e} \in \mathcal{P}_{\mathbf{e}_r} - \{\mathbf{e}_r\}}$
h_2
f_1^J:Para ver la lista de recursos
e_1^I:To
$u = (2,2)$
h_8
f_1^J:Para ver la lista de recursos
e_1^I:To $\underbrace{view\ a}_{\tilde{e} \equiv \mathbf{e}_r}$
...
$u = \ldots$

Figure 6.1: Example of the expansion of two hypotheses h_1 and h_2 given $f_1^J \equiv$ "Para ver la lista de recursos" and the user prefix $\mathbf{e}_p \equiv$ "To view a".

Regarding the hypothesis h_2, it has aligned the source phrase “Para”, appending the target phrase “To”. In this case, the prefix $\mathbf{e}_p$ has not been completely generated. Let $\mathbf{e}_r \equiv$ “view a” be the remaining words that are to be appended to h_2 to complete the user prefix. Under these circumstances, we have to take into account if the subsequent hypothesis extension yields a complete hypothesis or not. For example, let us suppose that we align the phrase positions $u = (2, 6)$ ($\tilde{f} \equiv$ “ver la lista de recursos”). Since this hypothesis extension produce a complete hypothesis, we have to ensure that the whole prefix $\mathbf{e}_p$ is generated. For this purpose, we append $\mathbf{e}_r$ to h_2, resulting in the hypothesis h_5. In addition to this, we can append phrases $\tilde{e}$ contained in the set $\mathcal{T}_{2,6}$ having $\mathbf{e}_r$ as prefix (if any). This allows the generation of hypotheses like h_6, that takes advantage of the information contained in the phrase table.

In contrast, if the next extension of h_2 do not produce a complete hypothesis, we can also append strings from the set $\mathcal{P}_{\mathbf{e}_r}$, of prefixes of $\mathbf{e}_r$, to the newly generated hypotheses, allowing the translation system to complete the whole prefix $\mathbf{e}_p$ in subsequent expansion processes. For example, let us suppose that we align the phrase positions $u = (2, 2)$ ($\tilde{f} \equiv$ “ver”). In this case we can append the phrase “view” which is a subprefix of $\mathbf{e}_r$, resulting in the hypothesis h_7. In addition to this, we can also append $\mathbf{e}_r$ itself, resulting in the hypothesis h_8. Finally, appending phrases from $\mathcal{T}_{2,2}$ having $\mathbf{e}_r$ as prefix (if any) can also be considered, although this situation has not been depicted in Figure 6.1.

Algorithm 6.1 shows the expansion algorithm that we propose for its application in IMT. The algorithm is a formalisation of the ideas depicted in Figure 6.1. This expansion algorithm permits phrase reorderings, but it can be easily modified to only obtain monotonic alignments.

The time cost of the IMT expansion algorithm can be reduced by the introduction of pruning techniques. Such pruning techniques include hypotheses recombination, stack length limitation and restrictions on the maximum number of target phrases that can be linked to an unaligned source phrase during the expansion process. Specifically, in those cases where $\mathbf{e}_p$ has not already been generated, only a subset of the strings contained in the set $\mathcal{P}_{\mathbf{e}_r}$ are considered as candidates for the expansion process. One possible criterion to choose the substrings is based on the length of the phrase $\tilde{f}$ to be translated determined by u. Only those substrings with lengths similar to the length of $\tilde{f}$ are considered. In addition, the set of expanded hypotheses that is returned by the algorithm can be sorted by score, keeping only the best ones.

Regarding the complexity of the search algorithm, it depends on the selected configuration of the generalised branch-and-bound algorithm. Let us assume that a breadth-first multiple stack algorithm with J stacks is used (see section 4.2.5). Under these circumstances, the search algorithm has to expand L_s hypotheses from J stacks, where L_s is the maximum stack length. The complexity of the `imt_expand` algorithm is in $\mathcal{O}(J^2 \cdot \max\{I, T\})$, where I is the length of the source sentence and T is the maximum number of phrase translations for a source phrase. Therefore, the complexity of the search algorithm is in $\mathcal{O}(J^3 \cdot L_s \cdot \max\{I, T\})$.

6.2.2 Scoring Function

The scoring function used in the branch-and-bound search algorithm for PB-SMT described in section 4.2 can also be used here. This is due to the fact that the search problem in IMT is equivalent to the problem of finding the best translation of the source sentence f_1^J where the

```
input    : $\mathbf{e}_p$ (user validated prefix), $h$ (hypothesis to be expanded)
           $\mathcal{T}_{j_1,j_2}$ (set of translations for $f_{j_1}^{j_2}$ in phrase table)
output   : $\mathcal{H}$ (set of expanded hypotheses)
auxiliar : $\mathcal{SP}_h$ (set of unaligned source positions of $h$),
           $PP(\mathcal{SP}_h)$ (set of all possible unaligned source phrase positions of $h$),
           $\mathbf{e}_r$ (remaining prefix words for a given hypothesis),
           $\mathcal{P}_{\mathbf{e}_r}$ (set of prefixes of $\mathbf{e}_r$)
begin
    $\mathbf{e}_r$ =get_remaining_prefix ($h$,$\mathbf{e}_p$);
    if $|\mathbf{e}_r| \neq 0$ then
        forall $(j_1, j_2) \in PP(\mathcal{SP}_h)$ do
            if $\mathcal{SP}_h - \{j_1, ..., j_2\} \neq \emptyset$ then
                forall $\tilde{e} \in \mathcal{P}_{\mathbf{e}_r} - \{\mathbf{e}_r\}$ do
                    $\mathcal{H} := \mathcal{H} \cup \{$append$(h, (j_1, j_2), \tilde{e})\}$
            $\mathcal{H} := \mathcal{H} \cup \{$append$(h, (j_1, j_2), \mathbf{e}_r)\}$
            forall $\tilde{e} \in \mathcal{T}_{j_1,j_2}$ do
                if is_prefix($\mathbf{e}_r$,$\tilde{e}$) and $\mathbf{e}_r \neq \tilde{e}$ then
                    $\mathcal{H} := \mathcal{H} \cup \{$append$(h, (j_1, j_2), \tilde{e})\}$
    else
        forall $(j_1, j_2) \in PP(\mathcal{SP}_h)$ do
            forall $\tilde{e} \in \mathcal{T}_{j_1,j_2}$ do
                $\mathcal{H} := \mathcal{H} \cup \{$append$(h, (j_1, j_2), \tilde{e})\}$
end
```

Algorithm 6.1: Pseudocode for the `imt_expand` algorithm.

target translations e_1^I are restricted to contain $\mathbf{e}_p$ as prefix (see section 1.8). Since the phrase models used during the translation process may not be able to explain the prefix given by the user (i.e. the phrase models may present coverage problems), smoothing techniques are needed to robustly generate the suffixes required in IMT. The smoothing techniques used in the generation of phrase-based alignments that were described in section 4.4.2 can be applied here without any modification.

As was stated in section 4.2, the branch-and-bound search algorithm for PB-SMT can be straightforwardly extended for its use with the log-linear model defined in section 3.5.3. We will use this log-linear model to assign scores to hypotheses in our IMT system based on partial phrase-based alignments.

6.3 IMT based on Stochastic Error-Correction Models

As has been already mentioned, a common problem in IMT arises when the user sets a prefix which cannot be explained by the statistical models. This problem requires the introduction of specific techniques in the IMT systems to guarantee that the suffixes can be generated. The vast majority of the IMT systems described in the literature (with the exception of the work presented in [TC06]) apply error-correcting techniques based on the concept of edit distance to solve the coverage problems. These error-correcting techniques, although they are not included in the statistical formulation of the IMT process, are crucial to ensure that the suffixes completing the prefixes given by the user can be generated.

In this section an alternative formalisation of the IMT framework which includes stochastic error-correction models in its statistical formulation is proposed. The remaining part of this section is organised as follows: first, probabilistic finite-state machines (PFSMs) are adapted for its use as stochastic error-correction models in section 6.3.1. Second, the details of the proposed alternative formalisation of the IMT framework are described in section 6.3.2. Finally, the proposed formalisation of IMT is generalised for its use in other pattern recognition applications in section 6.3.3.

6.3.1 PFSMs as Stochastic Error-Correction Models

To the best of our knowledge, the first stochastic interpretation of edit distance was described in [BJ75]. In that work, PFSMs were used to model the transformations produced by a noisy channel in a given text string.

PFSMs

A PFSM (see [VTdlH+05a, VTdlH+05b] for a detailed description) is a tuple $\mathcal{A} = \langle Q_\mathcal{A}, \Sigma, \delta_\mathcal{A}, I_\mathcal{A}, F_\mathcal{A}, P_\mathcal{A} \rangle$, where:

- $Q_\mathcal{A}$ is a finite set of states;
- Σ is the alphabet;
- $\delta_\mathcal{A} \subseteq Q_\mathcal{A} \times \Sigma \cup \{\lambda\} \times Q_\mathcal{A}$ is a set of transitions;
- $I_\mathcal{A} : Q_\mathcal{A} \rightarrow \mathbb{R}^+$ (initial-state probabilities);
- $P_\mathcal{A} : \delta_\mathcal{A} \rightarrow \mathbb{R}^+$ (transition probabilities);
- $F_\mathcal{A} : Q_\mathcal{A} \rightarrow \mathbb{R}^+$ (final-state probabilities).

$I_\mathcal{A}$, $P_\mathcal{A}$ and $F_\mathcal{A}$ are functions such that:

$$\sum_{q \in Q_\mathcal{A}} I_\mathcal{A}(q) = 1$$

and

$$\forall q \in Q_\mathcal{A}, F_\mathcal{A}(q) + \sum_{a \in \Sigma, q' \in Q_\mathcal{A}} P_\mathcal{A}(q, a, q') = 1$$

In what follows, we will use q without subindex to denote a generic state of Q, the specific states of Q will be denoted as $q_0, q_1 ..., q_{|Q|-1}$, and a sequence of states of length j will be denoted as $(s_0, s_1, ..., s_j)$ where $s_i \in Q$ for $1 \leq i \leq j$.

PFSMs are stochastic machines that generate probabilities for a set of finite strings contained in Σ^*. The generation of a string is a process that has two steps:

- Initialisation: Choose, with respect to a probability distribution I, one state $q_0 \in Q$ as the initial state. Define q_0 as the current state.
- Generation: Let q be the current state. Decide whether to stop, with probability $F(q)$, or to produce a transition (q, a, q') with probability $P(q, a, q')$, where $a \in \Sigma \cup \{\lambda\}$ and $q \in Q$ (λ is the empty string). Output a and set the current state to q'.

One relevant question to be solved regarding PFSMs is how to calculate the probability assigned by a PFSM, $\mathcal{A}$, to a given string $x \in \Sigma^*$. To deal with this problem, let $\theta = (s_0, x'_1, s_1), (s_1, x'_2, s_2), ..., (s_{k-1}, x'_k, s_k)$ be a *path* for $\mathbf{x}$ in $\mathcal{A}$; i.e. a sequence of transitions so that $\mathbf{x} = x'_1 x'_2 ... x'_k$ (note that, in general, $|\mathbf{x}| \leq k$ because some symbols x'_j may be λ).

The probability assigned to the path θ is given by the following expression:

$$p_{\mathcal{A}}(\theta) = I_{\mathcal{A}}(s_0) \cdot \left(\prod_{j=1}^{k} P_{\mathcal{A}}(s_{j-1}, x'_j, s_j) \right) \cdot F_{\mathcal{A}}(s_k) \tag{6.3}$$

In general, a given string $\mathbf{x}$ can be generated by $\mathcal{A}$ through multiple valid paths. Let $\Theta_{\mathcal{A}}(\mathbf{x})$ be the set of all the valid paths for $\mathbf{x}$ in $\mathcal{A}$. The probability of generating $\mathbf{x}$ with $\mathcal{A}$ is given by:

$$p_{\mathcal{A}}(\mathbf{x}) = \sum_{\theta \in \Theta_{\mathcal{A}}(\mathbf{x})} p_{\mathcal{A}}(\theta) \tag{6.4}$$

If $\sum_{\mathbf{x}} p_{\mathcal{A}}(\mathbf{x}) = 1$, then $\mathcal{A}$ defines a probability distribution D in Σ^*. This is guaranteed if $\mathcal{A}$ is *consistent*. A PFSM $\mathcal{A}$ is consistent if all its states appears in at least one valid path of $\Theta_{\mathcal{A}}$ [VTdlH+05a].

We will finish this brief introduction on PFSMs describing the concept of best path $\hat{\theta}$ for a string $\mathbf{x}$ in a given PFSM $\mathcal{A}$. The best path is given by the following expression:

$$\hat{\theta} = \underset{\theta \in \Theta_{\mathcal{A}}(\mathbf{x})}{\arg\max} \{p_{\mathcal{A}}(\theta)\} \tag{6.5}$$

PFSMs as Error-Correction Models

PFSMs can be used as stochastic error-correction models. Stochastic error-correction models based on the well-known concept of edit distance [Lev66] are implemented by means of PFSMs in [BJ75]. Specifically, these PFSMs are built by concatenating error-correction models based on PFSMs for individual symbols.

Figure 6.2 shows an example of PFSM that works as an stochastic error-correction model for a given symbol a contained in the alphabet Σ. We will note such a PFSM as $\mathcal{A}_a$. As can be observed, the figure shows transitions for the different edit distance operations, namely, insertions, substitutions and deletions. The edge associated to the emission of λ (the empty string) is represented with a dashed line.

Figure 6.3 shows the result of concatenating three PFSMs at symbol-level, so as to obtain an stochastic error-correction model for a text string $\mathbf{x} \in \Sigma^*$, where $\mathbf{x} = x_1 x_2 x_3$. The resulting PFSM can be minimised (see [BJ75]), giving the PFSM that is shown in Figure 6.4.

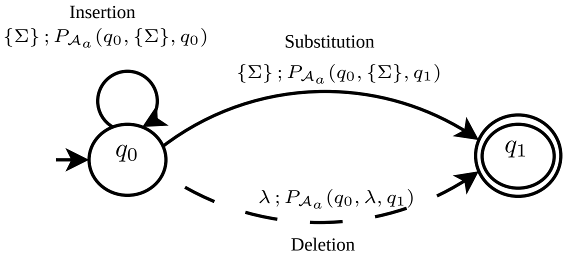

Insertion
$\{\Sigma\}\,;P_{\mathcal{A}_a}(q_0,\{\Sigma\},q_0)$
Substitution
$\{\Sigma\}\,;P_{\mathcal{A}_a}(q_0,\{\Sigma\},q_1)$
q_0
q_1
$\lambda\,;P_{\mathcal{A}_a}(q_0,\lambda,q_1)$
Deletion

Figure 6.2: Error-correction model for symbol $a \in \Sigma$, $\mathcal{A}_a$.

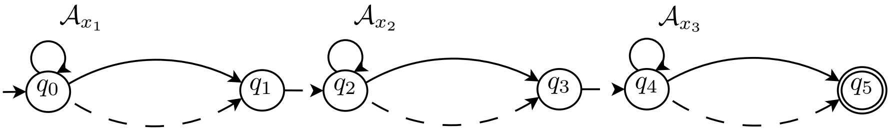


Figure 6.3: Error-correction model for string $\mathbf{x} = x_1x_2x_3$, $\mathcal{A}_x$. The model has been obtained by concatenating $\mathcal{A}_{x_1}$, $\mathcal{A}_{x_2}$ and $\mathcal{A}_{x_3}$.

Parameter Estimation

The problem of estimating the parameters of stochastic error-correction models based on the concept of edit distance has been studied in [RY97], where the use of the EM algorithm is proposed. Due to the great simplicity of the parameters to be estimated, one alternative to the application of the EM algorithm consists in the use of the so-called *ad-hoc* stochastic error-correction models [MDM91]. This technique was initially proposed for its use in the field of optical character recognition (OCR) and consists in reserving a probability mass for the substitution of one symbol by itself and distributing the rest of the probability mass among the different edit operations between strings.

We also propose the application of an ad-hoc stochastic error-correction model. One possible way of defining it consists in assigning a fixed probability mass to the substitution of one symbol by itself $p = 1 - \epsilon$, and uniformly distributing the remaining probability mass among the rest of possible transitions:

$$p' = \frac{\epsilon}{2|\Sigma|} \tag{6.6}$$

where $2|\Sigma|$ represents the number of transitions that have been defined for each state (with the exception of the substitution of one symbol by itself).

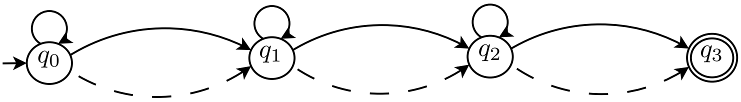


Figure 6.4: Reduced version of $\mathcal{A}_\mathbf{x}$.

Alternatively, the model described above can be refined by assigning different probabilities depending on the type of edit operation that is applied (insertion, substitution or deletion). For this purpose, we introduce three factors, f_i, f_s and f_d, that are used to assign weights to the probabilities assigned to insertions, substitutions and deletions, respectively. Given the value of ϵ and the values for the weighting factors, the auxiliary quantity c is defined as follows:

$$c = \frac{\epsilon}{(f_i|\Sigma|) + (f_s(|\Sigma| - 1)) + f_d} \tag{6.7}$$

The probabilities for the insertion, substitution and deletion operations: p_i, p_s and p_d, respectively, can be expressed in terms of the quantity c and the weighting factors f_i, f_s and f_d:

$$\begin{aligned} p_i &= f_i \times c & (6.8)\\ p_s &= f_s \times c & (6.9)\\ p_d &= f_d \times c & (6.10) \end{aligned}$$

Once we have defined the task in which the stochastic error-correction models will be applied, the values of the weighting factors f_i, f_s and f_d can be established by means of a development data set and the MERT algorithm.

Finding the Best Sequence of Edit Operations

We will end this section about stochastic error-correction models based on PFSMs discussing some issues regarding the problem of finding the best sequence of edit operations for a given string.

Given an error-correction model for the string $\mathbf{x} \in \Sigma^*$, $\mathcal{A}_{\mathbf{x}}$, and a string with errors, $\mathbf{x}'$, we will be interested in finding the best sequence of edit operations that are needed to change $\mathbf{x}$ into $\mathbf{x}'$. This problem is equivalent to the problem of finding the best path in $\mathcal{A}_{\mathbf{x}}$. For this purpose, the well-known *Viterbi algorithm* (see for example [VTdlH+05a] for more details) can be used.

Additionally, the problem of finding the best sequence of edit operations has been studied in a more general setting, where a PFSM and a stochastic error-correction model are given [AV98]. In such setting, the PFSM accounts for the set of different strings belonging to a given language, while the error-correction model accounts for the typical variations that the strings tend to exhibit with respect to their standard form as represented by the PFSM. Under these circumstances, Amengual and Vidal [AV98] propose simple extensions of the Viterbi algorithm that efficiently solve the problem of finding the best sequence of edit operations.

6.3.2 Alternative IMT Formalisation

We propose an alternative formalisation of the IMT paradigm in which the user prefix and the target sentence constitute separated entities. This allows us to introduce stochastic error-correction models in the statistical formulation of the IMT process.

Translating with User Prefix

The starting point of our alternative IMT formalisation consists in solving the problem of finding the sentence e_1^I in the target language that, at the same time, better explains the source sentence f_1^J and the prefix given by the user $\mathbf{e}_p$. This problem can be formally stated as follows:

$$\hat{e}_1^{\hat{I}} = \operatorname*{arg\,max}_{I,e_1^I}\{Pr(e_1^I|f_1^J,\mathbf{e}_p)\} \tag{6.11}$$

Using the Bayes rule we can write:

$$Pr(e_1^I|f_1^J,\mathbf{e}_p) = \frac{Pr(e_1^I)\cdot Pr(f_1^J,\mathbf{e}_p|e_1^I)}{Pr(f_1^J,\mathbf{e}_p)}$$

Since the denominator here is independent of e_1^I, the sentence $\hat{e}_1^{\hat{I}}$ can be found by maximising the expression $Pr(e_1^I)Pr(f_1^J,\mathbf{e}_p|e_1^I)$. We arrive then at the following equation:

$$\hat{e}_1^{\hat{I}} = \operatorname*{arg\,max}_{I,e_1^I}\{Pr(e_1^I)\cdot Pr(f_1^J,\mathbf{e}_p|e_1^I)\} \tag{6.12}$$

Now we make the following naive Bayes assumption: given the hypothesised target string e_1^I, the strings f_1^J and $\mathbf{e}_p$ are considered statistically independent. Thus, we obtain the following expression:

$$\hat{e}_1^{\hat{I}} = \operatorname*{arg\,max}_{I,e_1^I}\{Pr(e_1^I)\cdot Pr(f_1^J|e_1^I)\cdot Pr(\mathbf{e}_p|e_1^I)\} \tag{6.13}$$

In the previous equation the following terms can be found:

- $Pr(e_1^I)$: measures the well-formedness of e_1^I as a sentence of the target language. This distribution can be approximated by means of a statistical language model.
- $Pr(f_1^J|e_1^I)$: measures the appropriateness of the sentence f_1^J as a possible translation of e_1^I. This distribution can be approximated by means of a statistical translation model.
- $Pr(\mathbf{e}_p|e_1^I)$: measures the compatibility of e_1^I with the user prefix $\mathbf{e}_p$. This distribution can be approximated by stochastic error-correction models that are adequately modified for its use in IMT.

It should be noted that the result of the maximisation given by Equation (6.13), $\hat{e}_1^{\hat{I}}$, may not contain the prefix $\mathbf{e}_p$ given by the user, since every possible target sentence e_1^I is compatible with the user prefix with a certain probability. Because of this, the problem defined by Equation (6.13) is not equivalent to the problem of finding the best suffix in IMT.

To solve this problem, an additional assumption over the stochastic error-correction models is imposed. Specifically, they must be able to determine an *alignment* between a part of the target sentence e_1^I and the user prefix $\mathbf{e}_p$. The set of unaligned words of e_1^I, $\mathcal{U}_{e_1^I}$, in an appropriate order constitute the suffix required in IMT. To simplify things, we can also assume that we monotonically align a prefix of e_1^I with $\mathbf{e}_p$. This implies that the reordering problem is left to the language and translation models (i.e. the stochastic error-correction

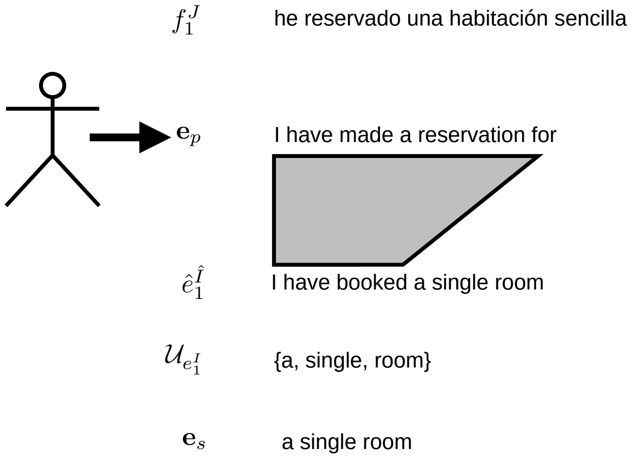


Figure 6.5: Example of how the IMT suffix is determined in our alternative IMT formalisation.

model is used to determine monotonic alignments between the user prefix and a prefix of the non-monotonic target translations of the source sentence). Under these circumstances, the suffix required in IMT is also a suffix of e_1^I.

The probability given by stochastic error-correction models can be expressed in terms of a hidden alignment variable, $\mathbf{p}$, as follows:

$$Pr(\mathbf{e}_p|e_1^I) = \sum_{\mathbf{p}} Pr(\mathbf{e}_p, \mathbf{p}|e_1^I) \tag{6.14}$$

According to Equation (6.14) and following a maximum-approximation, we can modify the problem stated in Equation (6.13) so as to obtain not only the sentence $\hat{e}_1^{\hat{I}}$, but also the alignment variable, $\hat{\mathbf{p}}$, which maximises the probability:

$$(\hat{e}_1^{\hat{I}}, \hat{\mathbf{p}}) = \underset{I, e_1^I, \mathbf{p}}{\arg\max}\{Pr(e_1^I) \cdot Pr(f_1^J|e_1^I) \cdot Pr(\mathbf{e}_p, \mathbf{p}|e_1^I)\} \tag{6.15}$$

Figure 6.5 shows how the IMT suffix is determined in our alternative IMT formalisation. The IMT system receives the source sentence, f_1^J, and the user prefix, $\mathbf{e}_p$, as input, and generates the best translation $\hat{e}_1^{\hat{I}}$ along with an alignment between a prefix of e_1^I and $\mathbf{e}_p$. The suffix $\mathbf{e}_s$ is obtained from the set $\mathcal{U}_{e_1^I}$ of words of e_1^I that are not aligned with $\mathbf{e}_p$.

It is noteworthy that the IMT technique proposed here has one point in common with the IMT technique based on partial phrase-based alignments described in section 6.2. Specifically, both IMT techniques require finding alignments for the user prefix: the IMT technique based on partial phrase-based alignments aligns the user prefix with the source sentence, while the IMT technique based on stochastic error-correction models aligns the user prefix with the target sentences generated as translation of the source sentence.

The stochastic error-correction models used in the proposed IMT formalisation can be defined in many ways. In this thesis, error-correction models based on PFSMs will be used.

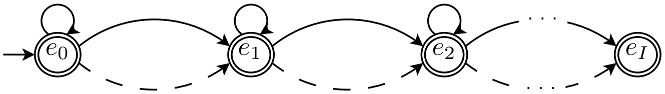


Figure 6.6: Error-correction model based on PFSMs for IMT given the sentence e_1^I: $\mathcal{B}_{e_1^I}$. The states of the PFSM are labelled with the words of the target sentence e_1^I.

It is worthy of note that the stochastic error-correction models can be seen as explicit models of the users of the IMT system.

Error-Correction Models based on PFSMs for IMT

Error-correction models based on PFSMs described in section 6.3.1 require some modifications for its use in IMT, since we want to model the probability distribution $Pr(\mathbf{e}_p|e_1^I)$, where $\mathbf{e}_p$ is a prefix instead of a complete sentence.

As a starting point, a stochastic error-correction model based on PFSMs, $\mathcal{A}_{e_1^I}$, is defined (see Figure 6.4 for an example of this kind of stochastic error-correction models); where $\mathcal{A}_{e_1^I}$ has been obtained as the concatenation of the error-correction models for each one of the words of e_1^I.

To allow this error-correction model to be used in the IMT framework, we only have to introduce one simple modification in $\mathcal{A}_{e_1^I}$. Specifically, we assume that $F_{\mathcal{A}}(q)$ is a non-null fixed quantity for each possible state q contained in $Q_{\mathcal{A}}$. We will note the resulting PFSM as $\mathcal{B}_{e_1^I}$. Figure 6.6 shows how the error-correction model is defined, the states of the PFSM are labelled with the words of the target sentence e_1^I.

Let $\theta = (s_0, x'_1, s_1), (s_1, x'_2, s_2), ..., (s_{k-1}, x'_k, s_k)$ be a valid path for $\mathbf{e}_p$ in $\mathcal{B}_{e_1^I}$, where s_i are states contained in $\mathcal{Q}_{\mathcal{B}_{e_1^I}} = \{e_0, ..., e_I\}$ and x'_i are words of $\mathbf{e}_p$ or the empty string λ. Each transition of the path is associated to an insertion, a substitution or a deletion operation. The path θ determines a monotonic alignment between a prefix of the target sentence e_1^I and the user prefix $\mathbf{e}_p$. The alignment decisions depend on the edit operation that is applied:

- **Insertions**: correspond to transitions of the form (e_i, x', e_i), where x' is a word of $\mathbf{e}_p$. When these transitions are added to θ, the word x' of $\mathbf{e}_p$ is not aligned with any word of e_1^I.

- **Substitutions**: correspond to transitions of the form (e_i, x', e_{i+1}), where x' is a word of $\mathbf{e}_p$. These transitions align the word e_{i+1} of e_1^I with the word x' of $\mathbf{e}_p$.

- **Deletions**: correspond to transitions of the form (e_i, λ, e_{i+1}). When these transitions are added to θ, the word e_{i+1} of e_1^I is aligned with the empty string.

The final state s_k of the path θ will be associated to the position i of the last word of e_1^I which accounts for the user prefix $\mathbf{e}_p$ (this last word and the previous words will be aligned with words of $\mathbf{e}_p$ or with the empty string). Therefore, the suffix $\mathbf{e}_s$ required in IMT will be determined by e_{i+1}^I.

Among all the valid paths for the string $\mathbf{e}_p$ in $\mathcal{B}_{e_1^I}$, $\Theta_{\mathcal{B}_{e_1^I}}(\mathbf{e}_p)$, we will be interested in that of the maximum probability $\hat{\theta}$, where:

$$\hat{\theta} = \underset{\theta \in \Theta_{\mathcal{B}_{e_1^I}}(\mathbf{e}_p)}{\arg\max} \{p_{\mathcal{B}_{e_1^I}}(\theta)\} \tag{6.16}$$

Hence, the best path $\hat{\theta}$ for $\mathbf{e}_p$ in $\mathcal{B}_{e_1^I}$ not only allows us to approximate the probability distribution $Pr(\mathbf{e}_p|e_1^I)$, but also to determine the part of e_1^I that constitutes the suffix $\mathbf{e}_s$ required in IMT.

It is worth noticing that the error-correction model for IMT defined here works at word level, but it could have been defined to work at character level instead. A character-level error-correction model would allow us to assign higher probabilities to the substitution of one word by another similar word. This advantage would be obtained at the cost of a higher time complexity. Alternatively, the proposed ad-hoc word-level error-correction model can also be replaced by a more complex word-level model which defines specific substitution probabilities.

Instantiation of the Alternative IMT Formalism

To instantiate our proposed alternative IMT formalism, the models that approximate the probability distributions that are present in Equation 6.15 have to be appropriately chosen. $Pr(e_1^I)$ and $Pr(f_1^J|e_1^I)$ can be approximated by a language model and a translation model, respectively. Regarding the probability distribution $Pr(\mathbf{e}_p, \mathbf{p}|e_1^I)$, it has to be approximated by an error-correction model that is able to determine an alignment between the target sentence e_1^I and the user prefix $\mathbf{e}_p$.

The stochastic error-correction models based on PFSMs for IMT described above can be used to determine the required alignment between e_1^I and $\mathbf{e}_p$. Specifically, this alignment is given by a path θ in the PFSM. The final state of θ determines the position i of the last word of e_1^I that accounts for the user prefix. Therefore, the suffix $\mathbf{e}_s$ is given by e_{i+1}^I.

Given the previous considerations, a particular instantiation of the proposed alternative IMT formalism can be defined as:

$$(\hat{e}_1^{\hat{I}}, \hat{\theta}) = \underset{I, e_1^I, \theta}{\arg\max}\{p(e_1^I) \cdot p(f_1^J|e_1^I) \cdot p_{\mathcal{B}_{e_1^I}}(\theta)\} \tag{6.17}$$

where θ is contained in the set of all possible paths for the string $\mathbf{e}_p$ in $\mathcal{B}_{e_1^I}$, $\Theta_{\mathcal{B}_{e_1^I}}(\mathbf{e}_p)$.

It should be noted that the IMT formalism based on the Bayes rule given by the previous equation can be replaced by another one based on the log-linear approach. The resulting log-linear model would be composed of the standard components used in fully-automatic PB-SMT, plus one more component corresponding to the log-probability given by the stochastic error-correction model.

The search procedure formalised by Equation (6.17) can be implemented as a process with two steps:

1. Generate a word graph for the source sentence f_1^J. The word graph is generated only once at the first interaction of the IMT process.

2. Apply the stochastic error-correction model over the target sentences contained in the word graph so as to obtain the pair $(\hat{e}_1^{\hat{I}}, \hat{\theta})$ of maximum probability.

The word graph that is required at Step 1 of the search procedure can be obtained as a by-product of the translation of f_1^J. This translation process can be carried out by means of the branch-and-bound search algorithm proposed in this thesis. The time complexity of the translation depends on how the search algorithm is configured. For example, if a breadth-first multiple stack algorithm with J stacks is used (see section 4.2.5), then the time complexity of the algorithm is in $\mathcal{O}(J^3 \cdot L_s \cdot T)$, where L_s is the maximum stack size and T is the maximum number of phrase translations for a source phrase.

Regarding the computation of the pair $(\hat{e}_1^{\hat{I}}, \hat{\theta})$ at Step 2 of the search procedure, it can be performed by means of a straightforward extension of the Viterbi algorithm. Such algorithm extension can be executed efficiently if the nodes of the word graph are visited in a topological order [AV98]. Given the user prefix $\mathbf{e}_p$ and a word graph with average branching factor B and $|Q|$ states, the asymptotic cost of the resulting algorithm is $\mathcal{O}(|\mathbf{e}_p| \cdot |Q| \cdot B)$.

6.3.3 Generalisation

The equations used in our alternative IMT formalisation can be generalised for their use in other pattern recognition applications. Let $\mathbf{x}$ and $\mathbf{y}$ be the source and the target patterns, respectively, and let $\mathbf{d}$ be a *distorted* version of the target pattern $\mathbf{y}$. Equation (6.13) can be rewritten as follows:

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} \{Pr(\mathbf{y}) \cdot Pr(\mathbf{x}|\mathbf{y}) \cdot Pr(\mathbf{d}|\mathbf{y})\} \tag{6.18}$$

If an alignment between the patterns $\mathbf{y}$ and $\mathbf{d}$ can be defined, then Equation (6.15) can be rewritten analogously.

Different pattern recognition applications can be derived from the previous equations depending on how the patterns $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{d}$ are defined. Below we show a list of some possible instantiations of our generalised formalism:

- **Multi-source translation**

  If $\mathbf{x}$ is a sentence in a source language, $\mathbf{y}$ is its corresponding translation in the target language and $\mathbf{d}$ is a translation of $\mathbf{x}$ in another language different to the target language, then the probability distribution $Pr(\mathbf{d}|\mathbf{y})$ is approximated by a translation model instead of an stochastic error-correction model. Under these circumstances, the system would take advantage of the information provided by $\mathbf{d}$ to obtain the best target translation $\hat{\mathbf{y}}$. We will refer to this application as multi-source SMT (MS-SMT). MS-SMT was first described in [ON01] and formalised in the same way as here.

- **Computer-assisted speech transcription**

  The goal of computer-assisted speech transcription (CAST) [RCV07] is to interactively obtain the transcription of an acoustic signal representing a sequence of words. In this case, $\mathbf{x}$ is an acoustic signal, $\mathbf{y}$ is a transcription of $\mathbf{x}$ proposed by the system and $\mathbf{d}$ is a prefix of $\mathbf{y}$ given by the user. The probability distribution $Pr(\mathbf{x}|\mathbf{y})$ is approximated by an acoustic model and $Pr(\mathbf{d}|\mathbf{y})$ is approximated by an stochastic error-correction model. The standard CAST formalisation given in [RCV07] is similar to the one proposed here, but it does not include a stochastic error-correction model.

- **Multimodal computer-assisted translation**

  In multimodal computer-assisted translation (MCAT), a human translator dictates the translation of a given source sentence. Given the source sentence and the target language acoustic sequence, the system should search for the most likely decoding of the acoustic sequence. In our formalisation, $\mathbf{x}$ is the source text to be translated, $\mathbf{y}$ is the target text, and $\mathbf{d}$ is an acoustic signal. The probability distribution $Pr(\mathbf{x}|\mathbf{y})$ is approximated by a translation model and $Pr(\mathbf{d}|\mathbf{y})$ is approximated by an acoustic model.

- **Computer-assisted transcription of text images**

  In computer-assisted transcription of text images (CATTI) [TVRV07], the input pattern $\mathbf{x}$ is a sequence of feature vectors describing a text image along its horizontal axis, and the system generates transcribed words $\mathbf{y}$ given a prefix $\mathbf{d}$ validated by the user. The probability distribution $Pr(\mathbf{x}|\mathbf{y})$ is approximated by morphological word models and $Pr(\mathbf{d}|\mathbf{y})$ is approximated by a stochastic error-correction model. The standard formalisation of the CATTI framework given in [TVRV07] is similar but not equal to the one proposed here.

Summary of Applications

To end this section, the main features of the pattern recognition applications that were described above will be summarised.

Table 6.1 shows, for each pattern recognition application (including IMT, MS-SMT, CAST, MCAT and CATTI), how the patterns $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{d}$ are defined; the models used to approximate the probability distributions[a]; and the output of the system.

Table 6.1: Summary of applications of the generalised formalisation.

Appl.	$\mathbf{x}$	$\mathbf{y}$	$\mathbf{d}$	$Pr(\mathbf{x}\|\mathbf{y})$	$Pr(\mathbf{d}\|\mathbf{y})$	**Output**
IMT	text	text	text(pref.)	trans. model	e.c. model	$(\hat{\mathbf{y}}, \hat{\mathbf{p}})$
MS-SMT	text	text	text	trans. model	trans. model	$\hat{\mathbf{y}}$
CAST	acoust. signal	text	text(pref.)	acoust. model	e.c. model	$(\hat{\mathbf{y}}, \hat{\mathbf{p}})$
MCAT	text	text	acoust. signal	trans. model	acoust. model	$\hat{\mathbf{y}}$
CATTI	text image	text	text	morph. model	e.c. model	$(\hat{\mathbf{y}}, \hat{\mathbf{p}})$

In the pattern recognition systems given in Table 6.1, two different sub-systems can be identified:

- **Automatic sub-system**: the automatic sub-system is related to the probability distributions $Pr(\mathbf{y})$ and $Pr(\mathbf{x}|\mathbf{y})$. These distributions do not depend on $\mathbf{d}$.

[a] The distribution $Pr(\mathbf{y})$ is not included in the table because it has the same meaning for all the applications.

- **User sub-system**: the user sub-system is related to the probability distribution $Pr(\mathbf{d}|\mathbf{y})$, which depends on $\mathbf{d}$.

It is worthy of note that a given sub-system can be reused in different pattern recognition applications if the involved probability distributions remain unchanged. For example, the same user sub-system can be used in the IMT, CAST and CATTI frameworks.

6.4 Summary

In this chapter, two novel IMT techniques have been presented: an IMT technique based on partial phrase-based alignments and an IMT technique based on stochastic error-correction models.

The IMT technique based on partial phrase-based alignments conceives the generation of the suffixes as a two step process: first, the user prefix is aligned with a part of the source sentence; second, the suffix is obtained as the translation of the unaligned portion of the source sentence. The generation of the partial phrase-based alignments is driven by statistical phrase-based models and relies on the application of smoothing techniques to assign probabilities to unseen events.

The IMT technique based on stochastic error-correction models follows an alternative formalisation of the IMT framework in which the user prefix and the target sentence generated by the system constitute separated entities. This alternative formalisation introduces stochastic error-correction models in the statistical formulation of the IMT process. This contrasts with existing IMT systems in which error correction is applied but not formally justified. The proposed IMT technique generates the suffixes required in IMT by partially aligning a prefix of the target hypotheses with the user prefix. Once the partial alignment is determined, the suffix is given by the unaligned portion of the target sentence. It is worth pointing out that stochastic error-correction models can be seen as explicit models of the user of the IMT system. Finally, the proposed alternative formalisation of the IMT framework can be generalised for its use in different pattern recognition applications.

CHAPTER 7
ONLINE LEARNING FOR INTERACTIVE PHRASE-BASED MACHINE TRANSLATION

7.1 Introduction

The vast majority of the existing work on IMT makes use of the well-known *batch learning* paradigm. In the batch learning paradigm, the training of the IMT system and the interactive translation process are carried out in separate stages. This paradigm is not able to take advantage of the new knowledge produced by the user of the IMT system. In this chapter, an application of the *online learning* paradigm to the IMT framework is presented. In the online learning paradigm, the training and prediction stages are no longer separated. This feature is particularly useful in IMT since it allows the user feedback to be taken into account.

The online learning techniques proposed here allow the statistical models involved in the translation process to be updated given the target translations validated by the user. Figure 7.1 shows a schematic view of these ideas, which contrasts with the diagram of a conventional IMT system shown in Figure 1.4. Here, f_1^J is the input sentence and e_1^I is the output derived by the IMT system from f_1^J. By observing f_1^J and e_1^I, the user interacts with the IMT system, validating prefixes and/or pressing keys (k) corresponding to the next correct character, until the desired output $\hat{e}_1^{\hat{I}}$ is produced. The input sentence f_1^J and its desired translation $\hat{e}_1^{\hat{I}}$ can be used to refine the models used by the system. In general, the model is initially obtained through a classical batch training process from a previously given training sequence of pairs $(\mathbf{f}_1, \mathbf{e}_1)$,...,$(\mathbf{f}_n, \mathbf{e}_n)$ from the task being considered. Now, the models can be extended with the use of valuable user feedback by means of online learning techniques.

The online learning paradigm has been previously applied to train discriminative models in SMT [LBCKT06, AK07, WSTI07, CMR08]. These works differ from the one presented here in that we apply online learning techniques to train generative models instead of discriminative models.

More recently, Levenberg et al. [LCBO10] introduced an online training regime for

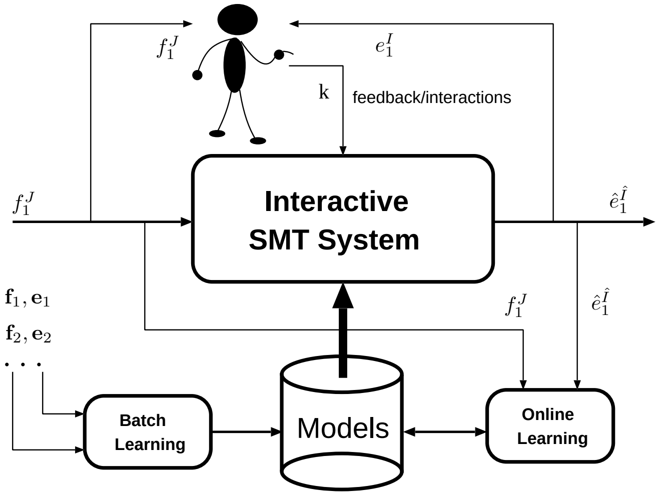


Figure 7.1: An Online Interactive SMT system.

phrase-based models which is applied in a fully-automatic statistical machine translation system. Such training regime is based on the application of the so-called stepwise EM algorithm [CM09]. Our work differs from the work of Levenberg et al. [LCBO10] in that we use the incremental version of the EM algorithm [NH98] instead of the stepwise version (the details of our proposal will be described below). Additionally, our proposed techniques can be applied to incrementally train all of the different models used by an IMT system.

In [NLLF04], dynamic adaptation of an IMT system via cache-based model extensions to language and translation models is proposed. The work by Nepveu et al. [NLLF04] constitutes a domain adaptation technique and not an online learning technique, since the proposed cache components require pre-existent models estimated in batch mode. In addition to this, their IMT system does not use state-of-the-art models.

To our knowledge, the only previous work on online learning for IMT is [CBRS08], where a very constrained version of online learning is presented. This constrained version of online learning is not able to extend the translation models due to technical problems with the efficiency of the learning process. By contrast, we present a purely statistical IMT system which is able to incrementally update the parameters of all of the different models that are used in the system, including the translation model, breaking with the above mentioned constraints. What is more, our system is able to learn from scratch, that is, without any preexisting model stored in the system.

The remaining part of this chapter is structured as follows: batch and online learning paradigms are described in section 7.2. Incremental learning is discussed as one possible way to implement online learning in section 7.3. The basic IMT system that is used to implement our IMT system with online learning is described in section 7.4. The techniques that are required to incrementally update the proposed basic IMT system are explained in section 7.5. Finally, a summary of this chapter is given in section 7.6.

7.2 Batch Learning versus Online Learning

In the IMT system proposed in this chapter, the batch learning paradigm is replaced by the online learning paradigm. The goal in online learning (as in other learning paradigms) is to predict labels from instances. The key aspect of online learning is that soon after the prediction is made, the true label of the instance is presented to the learning algorithm. This information can then be used to refine subsequent predictions.

Online learning algorithms proceed in a sequence of trials. Each trial can be decomposed into three steps:

1. The learning algorithm receives an instance.
2. The learning algorithm predicts a label for the instance.
3. The true label of the instance is presented to the learning algorithm.

After the true label of the instance has been discovered, the learning algorithm uses it to minimise a pre-determined performance criterion. Typically, this pre-determined criterion is based on the amount of error in the label predicted at Step 2, compared to the true label given at Step 3. Two well-known examples of online learning algorithms are the *Perceptron* algorithm [Ros58] and the *Winnow* algorithm [Lit88].

The online learning setting described above contrasts with the batch learning setting, in which all the training patterns are presented to the learner before learning takes place and the learner is no longer updated after the learning stage has concluded.

Batch learning algorithms are appropriate for their use in stationary environments. In a stationary environment, all instances are drawn from the same underlying probability distribution. By contrast, since the online learning algorithms continually receive prediction feedback, they can be used in non-stationary environments.

7.3 Online Learning as Incremental Learning

One possible way to implement online learning consists in the use of incremental learning algorithms.

Incremental learning is appropriate in those learning tasks in which learning must take place over time in a kind of continuous fashion rather than from a training data set available a priori. A review on incremental learning can be found in [GC00].

According to [GC00], the main characteristics of an incremental learning task are the following:

- Examples are not available a priori but become available over time, usually one at a time.
- Learning may need to go on indefinitely.

A learning algorithm is incremental if for any sequence of training samples $\mathbf{x}_1, ..., \mathbf{x}_N$, produces a sequence of parameters: $\Theta^{(0)}, \Theta^{(1)}, ..., \Theta^{(N)}$, such that the algorithm parameters at instant t, $\Theta^{(t)}$, depends only on the previous parameters, $\Theta^{(t-1)}$, and the current sample $\mathbf{x}_t$.

The main features of an incremental learning algorithm are the following:

- No re-processing of previous samples is required.
- Since the knowledge is incrementally acquired, the learner can, at any time, produce an answer to a query and the quality of its answers improves over time.

Incremental learning algorithms are also called memoryless online algorithms (see [AB92]) since they constitute online learning algorithms that discard each new training sample after updating the learner.

It is interesting to consider some issues raised in the design of incremental learning algorithms:

- **Ordering effects**: Chronology, or the order in which knowledge is acquired, is an inherent aspect of incrementality.
- **Learning curve**: An incremental system may start from scratch and gain knowledge from examples given one at a time over time. As a result, the system experiences a sort of learning curve, where the quality of its predictions improves over time.
- **Open-world assumption**: All the data relevant to the problem at hand is not available a priori. Then the world cannot be assumed to be closed. As a consequence of this, there is a need for special learning mechanisms that invalidate portions of knowledge, while not affecting the rest of it.

If the incremental learning algorithm is based on statistical models, then we need to maintain a set of *sufficient statistics* for these models that can be incrementally updated. A sufficient statistic for a statistical model is a statistic that captures all the information that is relevant to estimate this model. If the estimation of the statistical model does not require the use of the EM algorithm (e.g. n-gram language models), then it is generally easy to incrementally extend the model given a new training sample. By contrast, if the EM algorithm is required (e.g. word alignment models), the estimation procedure has to be modified, since the conventional EM algorithm is designed for its use in batch learning scenarios. To solve this problem, an incremental version of the EM algorithm is required.

7.3.1 Incremental View of the EM Algorithm

Neal and Hinton [NH98] proposed an alternative view of the EM algorithm in which it is seen as maximising a joint function of the parameters and of the distribution over the unobserved variables. The E step maximises this function with respect to the distribution over unobserved variables and the M step with respect to the parameters.

As a starting point, Neal and Hinton [NH98] formulate the EM algorithm in a slightly non-standard fashion as follows:

$$\left.\begin{array}{ll}\textbf{E step}: & \text{Compute a distribution over the hidden data, } q^{(t)}\text{, such} \\ & \text{that } q^{(t)}(\mathbf{y}) = p(\mathbf{y}|\mathbf{x}, \Theta^{(t-1)}) \\ \textbf{M step}: & \text{Set } \Theta^{(t)} \text{ to be the } \Theta \text{ that maximises } E_{q^{(t)}}[\log p(\mathbf{x},\mathbf{y}|\Theta)]\end{array}\right\} \tag{7.1}$$

where $\mathbf{x}$ is the observed variable, $\mathbf{y}$ is the hidden variable, $\Theta^{(t)}$ are the model parameters at instant t and $E_{q^{(t)}}[\cdot]$ denotes expectation with respect to the distribution over the range of $\mathbf{y}$ given by $q^{(t)}$.

The E step of the algorithm can be seen as representing the unknown value for $\mathbf{y}$ by a distribution of values, and the M step as then performing maximum-likelihood estimation for the joint data by combining $\mathbf{x}$ and $\mathbf{y}$. As shown by Dempster et al. [DLR77], each EM iteration improves the log-likelihood of the observed data $\mathcal{L}(\Theta, \mathbf{x}) = \log p(\mathbf{x}|\Theta)$ or leaves it unchanged. Such monotonic improvement in $\mathcal{L}(\Theta, \mathbf{x})$ is guaranteed by the generalised EM (GEM) algorithm [DLR77], in which only a partial maximisation is performed in the EM step, with $\Theta^{(t)}$ set to some value $E_{q^{(t)}}[\log p(\mathbf{x}, \mathbf{y}|\Theta^{(t)})]$ such that is greater than $E_{q^{(t)}}[\log p(\mathbf{x}, \mathbf{y}|\Theta^{(t-1)})]$.

An alternative view of the EM algorithm can be proposed in which both the E and the M steps are seen as maximising, or at least increasing the same function, $F(q, \Theta)$, allowing us to also partially performing the E step.

The function $F(q, \Theta)$ is defined as follows:

$$F(q, \Theta) = E_q[\log p(\mathbf{x}, \mathbf{y}|\Theta)] + H(q) \tag{7.2}$$

where $H(q)$ is the entropy of the distribution q:

$$H(q) = -E_q[\log q(\mathbf{y})] \tag{7.3}$$

The following two lemmas state properties of the function F (see [NH98]):

Lemma 1 *For a fixed value of Θ, there is a unique distribution, q_Θ that maximises $F(q, \Theta)$ given by $q_\Theta(\mathbf{y}) = p(\mathbf{y}|\mathbf{x}, \Theta)$. Furthermore, this q_Θ varies continuously with Θ.*

Lemma 2 *If $q(\mathbf{y}) = p(\mathbf{y}|\mathbf{x}, \Theta) = q_\Theta(\mathbf{y})$ then $F(q, \Theta) = \log p(\mathbf{x}|\Theta) = \mathcal{L}(\Theta, \mathbf{x})$.*

The EM algorithm can be formulated in terms of the function $F(q, \Theta)$ as follows:

$$\left.\begin{array}{ll} \textbf{E step}: & \text{Set } q^{(t)} \text{ to the } q \text{ that maximises } F(q, \Theta^{(t-1)}) \\ \textbf{M step}: & \text{Set } \Theta^{(t)} \text{ to the } \Theta \text{ that maximises } F(q^{(t)}, \Theta) \end{array}\right\} \tag{7.4}$$

The EM iterations given by (7.1) and (7.4) are equivalent. That the E steps of the iterations are equivalent, follows directly from Lemma 1. That the M steps are equivalent follows from the fact that the entropy term in the definition of F in Equation (7.2) does not depend on Θ.

Once the EM iteration has been expressed in the form given by (7.4), it is clear that the algorithm converges to values q^* and Θ^* that locally maximise $F(q, \Theta)$. In general, finding a local maximum for $F(q, \Theta)$ will also yield a local maximum for $\mathcal{L}(\Theta, \mathbf{x})$, justifying not only the EM algorithm given by (7.4), but variants of it in which the E and M steps are performed partially. This can be formally stated by means of the following theorem, which is theoretically demonstrated using Lemmas (1) and (2) in [NH98]:

Theorem 1 *If $F(q, \Theta)$ has a local maximum at q^* and Θ^*, then $\mathcal{L}(\Theta, \mathbf{x})$ has a local maximum at Θ^* as well. Similarly, if F has a global maximum at q^* and Θ^*, then $\mathcal{L}$ has a global maximum at Θ^*.*

An incremental variant of the EM algorithm can be justified on the basis of Theorem 1 in those cases in which the maximum-likelihood parameter estimates are obtained from independent data items. Specifically, the observed variable, $\mathbf{x}$, is decomposed as $(\mathbf{x}_1, ..., \mathbf{x}_N)$, and the hidden variable $\mathbf{y}$ is decomposed as $(\mathbf{y}_1, ..., \mathbf{y}_N)$, allowing us to decompose the joint probability distribution as $p(\mathbf{x}, \mathbf{y}|\Theta) = \prod_n p(\mathbf{x}_n, \mathbf{y}_n|\Theta)$.

Using the above mentioned decomposition of the hidden variable $\mathbf{y}$, the search for a maximum of F can be restricted to distributions q that factor as $q(\mathbf{y}) = \prod_n q_n(\mathbf{y}_n)$. This allows us to write F as $F(q, \Theta) = \sum_n F_n(q_n, \Theta)$, where $F_n(q_n, \Theta)$ is given by the following expression:

$$F_n(q_n, \Theta) = E_{q_n}[\log p(\mathbf{x}_n, \mathbf{y}_n|\Theta)] + H(q_n) \tag{7.5}$$

The following incremental EM iteration can be used to find a maximum of F and hence $\mathcal{L}(\Theta, \mathbf{x})$, starting from some guess at the parameters, $\Theta^{(0)}$, and some guess at the distribution, $q_n^{(0)}$, which may or may not be consistent with $\Theta^{(0)}$:

$$\left.\begin{array}{ll} \textbf{E step}: & \text{Choose some data item, } n \text{ to be updated.}\\ & \text{Set } q_m^{(t)} = q_m^{(t-1)} \text{ for } m \neq n.\\ & \text{Set } q_n^{(t)} \text{ to the } q_n \text{ that maximises } F_n(q_n, \Theta^{(t-1)}),\\ & \text{given by } q_n^{(t)} = p(\mathbf{y}_n|\mathbf{x}_n, \Theta^{(t-1)}).\\ \textbf{M step}: & \text{Set } \Theta^{(t)} \text{ to the } \Theta \text{ that maximises } F(q^{(t)}, \Theta)\text{, or,}\\ & \text{equivalently, that maximises } E_{q^{(t)}}[\log p(\mathbf{x}, \mathbf{y}|\Theta)] \end{array}\right\} \tag{7.6}$$

In the previous EM iteration, the E step process one data item at a time, while the M step, as written, looks like if it requires looking at all components of q. This can be avoided in those cases in which the inferential import of the complete data can be summarised by means of a vector of sufficient statistics that are incrementally updateable.

Letting this vector of sufficient statistics be $s(\mathbf{x}, \mathbf{y}) = \sum_n s_n(\mathbf{x}_n, \mathbf{y}_n)$, the standard EM iteration of (7.1) can be reformulated as follows:

$$\left.\begin{array}{ll} \textbf{E step}: & \text{Set } \tilde{s}^{(t)} = E_q[s(\mathbf{x}, \mathbf{y})]\text{, where } q(\mathbf{y}) = p(\mathbf{y}|\mathbf{x}, \Theta^{(t-1)}).\\ & \text{(In detail, set } \tilde{s}^{(t)} = \sum_n \tilde{s}_n^{(t)}\text{, with } \tilde{s}_n^{(t)} = E_q[s_n(\mathbf{x}_n, \mathbf{y}_n)],\\ & \text{where } q_n(\mathbf{y}_n) = p(\mathbf{y}_n|\mathbf{x}_n, \Theta^{(t-1)})).\\ \textbf{M step}: & \text{Set } \Theta^{(t)} \text{ to the } \Theta \text{ with maximum likelihood given } \tilde{s}^{(t)} \end{array}\right\} \tag{7.7}$$

Similarly, the iteration of (7.6), can be implemented using sufficient statistics that are incrementally updated, starting with an initial guess $\tilde{s}^{(0)}$, which may or may not be consistent with $\Theta^{(0)}$:

$$\left.\begin{array}{ll} \textbf{E step}: & \text{Choose some data item, } n \text{ to be updated.}\\ & \text{Set } \tilde{s}_m^{(t)} = s_m^{(t-1)} \text{ for } m \neq n.\\ & \text{Set } \tilde{s}_n^{(t)} = E_{q_n}[s_n(\mathbf{x}_n, \mathbf{y}_n)]\text{, for } q_n(\mathbf{y}_n) = p(\mathbf{y}_n|\mathbf{x}_n, \Theta^{(t-1)}).\\ & \text{Set } \tilde{s}^{(t)} = \tilde{s}^{(t-1)} - \tilde{s}_n^{(t-1)} + \tilde{s}_n^{(t)}.\\ \textbf{M step}: & \text{Set } \Theta^{(t)} \text{ to the } \Theta \text{ with maximum likelihood given } \tilde{s}^{(t)} \end{array}\right\} \tag{7.8}$$

In iteration (7.8), both the E and the M steps take constant time, independent of the number of data items. It is worthy of note that the incremental EM algorithm is expected to converge faster than the conventional EM algorithm, since the model parameters are updated for each data item instead of for the whole training data set.

7.4 Basic IMT System

In this section we describe the basic IMT system that is used as the basis of our online IMT system. Specifically, we propose the use of the IMT system based on PSPBAs that was described in section 6.2.2. Such an IMT system uses a log-linear model composed of seven feature functions (from h_1 to h_7) to generate its translations. This log-linear model is based on a specific set of hidden variables used to determine the phrase alignments: (K, a_1^K, b_1^K, c_1^K). The meaning of each hidden variable is the following: K represents the length of the bisegmentation, a_1^K is a vector of ending positions of the K target phrases, b_1^K is a vector with the number of skipped source positions with respect to the ending position of the previously aligned source phrase and c_1^K represents a vector containing the lengths of the K source phrases.

A specific instantiation of the log-linear model presented in section 3.5.3 have been adopted here. This specific instantiation includes an n-gram language model with interpolated Kneser-Ney smoothing and phrase-based models (estimated in both translation directions) combined with an HMM-based alignment model by means of linear interpolation. These models have been chosen because they are competitive with the state of the art in SMT and, at the same time, they can be incrementally updated by using efficient algorithms, as will be shown in section 7.5.

Below we give the details of the instantiation of each log-linear model component. This detailed description is required to later introduce the incremental update rules for the components.

- **n-gram language model (h_1)**
  $h_1(e_1^I) = \log(\prod_{i=1}^{I+1} p(e_i|e_{i-n+1}^{i-1}))$, where $p(e_i|e_{i-n+1}^{i-1})$ is defined as follows:

$$p(e_i|e_{i-n+1}^{i-1}) = \frac{\max\{c_X(e_{i-n+1}^{i}) - D_n, 0\}}{c_X(e_{i-n+1}^{i-1})} + \frac{D_n}{c_X(e_{i-n+1}^{i-1})} \cdot N_{1+}(e_{i-n+1}^{i-1}\bullet) \cdot p(e_i|e_{i-n+2}^{i-1}) \tag{7.9}$$

  where $D_n = \frac{c_{n,1}}{c_{n,1}+2c_{n,2}}$ is a fixed discount ($c_{n,1}$ and $c_{n,2}$ are the number of n-grams with one and two counts respectively), $N_{1+}(e_{i-n+1}^{i-1}\bullet)$ is the number of unique words that follows the history e_{i-n+1}^{i-1} and $c_X(e_{i-n+1}^{i})$ is the count of the n-gram e_{i-n+1}^{i}, where $c_X(\cdot)$ can represent true counts, $c_T(\cdot)$, or modified counts, $c_M(\cdot)$. True counts are used for the higher order n-grams and modified counts for the lower order n-grams. Given a certain n-gram, its modified count consists in the number of different words that precede this n-gram in the training corpus.

  Equation (7.9) corresponds to the probability given by an n-gram language model with an interpolated version of the Kneser-Ney smoothing [CG96].

- **source sentence-length model (h_2)**
  $h_2(e_1^I, f_1^J) = \log(p(J\,|\,I)) = \log(\phi_I(J + 0.5) - \phi_I(J - 0.5))$, where $\phi_I(\cdot)$ denotes the cumulative distribution function (cdf) for the normal distribution (the cdf is used here to

integrate the normal density function over an interval of length 1). A specific normal distribution with mean μ_I and standard deviation σ_I is used for each possible target sentence length I.

- **inverse and direct phrase-based models (**h_3, h_4**)**
  $h_3(e_1^I, K, a_1^K, b_1^K, c_1^K, f_1^J) = \log(\prod_{k=1}^K p_{LI}(f^{\beta_k}_{\alpha_k}|e^{a_k}_{a_{k-1}+1}))$ where $p_{LI}(f^{\beta_k}_{\alpha_k}|e^{a_k}_{a_{k-1}+1})$ is defined as follows:

  $$\begin{aligned} p_{LI}(f^{\beta_k}_{\alpha_k}|e^{a_k}_{a_{k-1}+1}) &= \lambda \cdot p(f^{\beta_k}_{\alpha_k}|e^{a_k}_{a_{k-1}+1}) + \\ & (1-\lambda).p_{hmm}(f^{\beta_k}_{\alpha_k}|e^{a_k}_{a_{k-1}+1}) \end{aligned} \tag{7.10}$$

  In Equation (7.10), $p(f^{\beta_k}_{\alpha_k}|e^{a_k}_{a_{k-1}+1})$ denotes the probability given by a statistical phrase-based dictionary used in regular phrase-based models, where:

  $$\begin{aligned} \alpha_k &= \beta_k - c_k + 1 \\ \beta_k &= \beta_{k-1} + b_k + c_k \\ \beta_0 &= 0 \end{aligned}$$

  $p_{hmm}(f^{\beta_k}_{\alpha_k}|e^{a_k}_{a_{k-1}+1})$ is the probability given by an HMM-based (intra-phrase) alignment model:

  $$p_{hmm}(\tilde{f}|\tilde{e}) = \epsilon \sum_{a_1^{|\tilde{f}|}} \prod_{j=1}^{|\tilde{f}|} p(\tilde{f}_j|\tilde{e}_{a_j}) \cdot p(a_j|a_{j-1}, |\tilde{e}|) \tag{7.11}$$

  The HMM-based alignment model probability is used here for smoothing.

  Analogously h_4 is defined as:
  $h_4(e_1^I, K, a_1^K, b_1^K, c_1^K, f_1^J) = \log(\prod_{k=1}^K p_{LI}(e^{a_k}_{a_{k-1}+1}|f^{\beta_k}_{\alpha_k}))$

- **target phrase-length model (**h_5**)**
  $h_5(K, a_1^K) = \log(\prod_{k=1}^K p(a_k|a_{k-1})$,
  where $p(a_k|a_{k-1}) = p(a_k - a_{k-1}) = \delta(1-\delta)^{a_k - a_{k-1}}$. h_5 implements a target phrase-length model by means of a geometric distribution with probability of success δ on each trial. The use of a geometric distribution penalises the length of target phrases.

- **source phrase-length model (**h_6**)**
  $h_6(K, a_1^K, c_1^K) = \log(\prod_{k=1}^K p(c_k|a_k, a_{k-1}))$,
  where $p(c_k|a_k, a_{k-1}) = \frac{1}{1+\tau}\delta(1-\delta)^{abs(c_k-(a_k-a_{k-1}))}$, $\tau = \sum_{i=1}^{a_k-a_{k-1}-1} \delta(1-\delta)^i$ and $abs(\cdot)$ is the absolute value function. A modified geometric distribution with probability of success δ on each trial is used to model this feature. The scaling factor $\frac{1}{1+\tau}$ is introduced to make the distribution sum to one because the term $c_k - (a_k - a_{k-1})$ takes integer values (c_k and $(a_k - a_{k-1})$ are greater than zero). This distribution penalises the difference between the source and target phrase lengths.

- **distortion model (**h_7**)**
  $h_7(K, b_1^K) = \log(\prod_{k=1}^K p(b_k))$,

where $p(b_k) = \frac{1}{2-\delta}\delta(1-\delta)^{abs(b_k)}$. A modified geometric distribution with probability of success δ on each trial is used to assign probabilities to the number of skipped source words. The scaling factor $\frac{1}{2-\delta}$ is introduced to make the distribution sum to one (b_k takes integer values). The use of a geometric distribution penalises the reorderings.

The log-linear model, which includes the above described feature functions, is used to generate the suffix $\mathbf{e}_s$ given the user-validated prefix $\mathbf{e}_p$. Specifically, the IMT system generates a partial phrase-based alignment between the user prefix $\mathbf{e}_p$ and a portion of the source sentence f_1^J, and returns the suffix $\mathbf{e}_s$ as the translation of the remaining portion of f_1^J.

7.5 Online IMT System

After translating a source sentence f_1^J, a new sentence pair (f_1^J, e_1^I) is available to feed the IMT system (see Figure 1.3). To do this, a set of sufficient statistics that can be incrementally updated is maintained for the statistical models that implement each feature function $h_i(\cdot)$.

In the following sections, we show how the set of sufficient statistics is defined for each model. Regarding the weights of the log-linear combination, they are not modified due to the presentation of a new sentence pair to the system. These weights can be adjusted off-line by means of a development corpus and well-known optimisation techniques.

7.5.1 Language Model (h_1)

Feature function h_1 implements a language model. According to Equation (7.9), the following data is to be maintained: $c_{k,1}$ and $c_{k,2}$ given any order k, $N_{1+}(\cdot)$, and $c_X(\cdot)$ (see section 7.4 for the meaning of each symbol).

Given a new sentence e_1^I, and for each k-gram e_{i-k+1}^i of e_1^I, where $1 \leq k \leq n$ and $1 \leq i \leq I+1$, the set of sufficient statistics is modified as it is shown in Algorithm 7.1. The algorithm checks the changes in the counts of the k-grams to update the set of sufficient statistics. For a given k-gram, e_{i-k+1}^i, its true count and the corresponding normaliser are updated at lines 13 and 14, respectively. The modified count of the $(k-1)$-gram and its normaliser are updated at lines 7 and 8, respectively, only when the k-gram e_{i-k+1}^i appears for the first time (condition checked at line 2). The value of the $N_{1+}(\cdot)$ statistic for e_{i-k+1}^{i-1} and e_{i-k+2}^{i-1} is updated at lines 10 and 6, respectively, only if the word e_i has been seen for the first time following these contexts. Finally, sufficient statistics for D_k are updated at lines 12 (for higher order n-grams) and 4 (for lower order n-grams), following the auxiliary procedure shown in Algorithm 7.2.

7.5.2 Sentence Length Model (h_2)

Feature function h_2 implements a sentence length model. h_2 requires the incremental calculation of the mean μ_I and the standard deviation σ_I of the normal distribution associated to a target sentence length I. For this purpose, the procedure described in [Knu81] can be used. In this procedure, two quantities are maintained for each normal distribution: μ_I and

```
input  : $n$ (higher order), $e_{i-k+1}^{i}$ ($k$-gram),
         $\mathcal{S} = \{\forall j(c_{j,1}, c_{j,2}), N_{1+}(\cdot), c_X(\cdot)\}$ (current set of sufficient statistics)
output : $\mathcal{S}$ (updated set of sufficient statistics)
begin
    if $c_T(e_{i-k+1}^{i}) = 0$ then
        if $k - 1 \geq 1$ then
            updD($\mathcal{S}$,$k$-1,$c_M(e_{i-k+2}^{i})$,$c_M(e_{i-k+2}^{i}) + 1$)
            if $c_M(e_{i-k+2}^{i}) = 0$ then
                $N_{1+}(e_{i-k+2}^{i-1}) := N_{1+}(e_{i-k+2}^{i-1}) + 1$
            $c_M(e_{i-k+2}^{i}) := c_M(e_{i-k+2}^{i}) + 1$
            $c_M(e_{i-k+2}^{i-1}) := c_M(e_{i-k+2}^{i-1}) + 1$
        if $k = n$ then
            $N_{1+}(e_{i-k+1}^{i-1}) := N_{1+}(e_{i-k+1}^{i-1}) + 1$
    if $k = n$ then
        updD($\mathcal{S}$,$k$,$c_T(e_{i-k+1}^{i})$,$c_T(e_{i-k+1}^{i}) + 1$)
    $c_T(e_{i-k+1}^{i})$:=$c_T(e_{i-k+1}^{i}) + 1$
    $c_T(e_{i-k+1}^{i-1})$:=$c_T(e_{i-k+1}^{i-1}) + 1$
end
```

Algorithm 7.1: Pseudocode for the `update_suff_stats_lm` algorithm.

```
input  : $\mathcal{S}$ (current set of sufficient statistics),$k$ (order), $c$ (current count),
         $c'$ (new count)
output : $(c_{k,1}, c_{k,2})$ (updated sufficient statistics)
begin
    if $c = 0$ then
        if $c' = 1$ then $c_{k,1} := c_{k,1} + 1$
        if $c' = 2$ then $c_{k,2} := c_{k,2} + 1$
    if $c = 1$ then
        $c_{k,1} := c_{k,1} - 1$
        if $c' = 2$ then $c_{k,2} := c_{k,2} + 1$
    if $c = 2$ then $c_{k,2} := c_{k,2} - 1$
end
```

Algorithm 7.2: Pseudocode for the `updD` algorithm.

S_I. Given the n-th training pair $(\mathbf{f}_n, \mathbf{e}_n)$ at instant t, the two quantities are updated according to the following equations:

$$\mu_{i:i\neq|\mathbf{e}_n|}^{(t)} = \mu_i^{(t-1)} \tag{7.12}$$

$$\mu_{|\mathbf{e}_n|}^{(t)} = \mu_{|\mathbf{e}_n|}^{(t-1)} + (|\mathbf{f}_n| - \mu_{|\mathbf{e}_n|}^{(t-1)})/c(|\mathbf{e}_n|) \tag{7.13}$$

$$S_{i:i\neq|\mathbf{e}_n|}^{(t)} = S_i^{(t-1)} \tag{7.14}$$

$$S_{|\mathbf{e}_n|}^{(t)} = S_{|\mathbf{e}_n|}^{(t-1)} + (|\mathbf{f}_n| - \mu_{|\mathbf{e}_n|}^{(t-1)}) \cdot (|\mathbf{f}_n| - \mu_{|\mathbf{e}_n|}^{(t)}) \tag{7.15}$$

where $c(|\mathbf{e}_n|)$ is the count of the number of sentences of length $|\mathbf{e}_n|$ that have been seen so far, and $\mu^{(t-1)}_{|\mathbf{e}_n|}$ and $S^{(t-1)}_{|\mathbf{e}_n|}$ are the quantities previously stored ($\mu^{(0)}_{|\mathbf{e}_n|}$ is initialised to the source sentence length of the first sample and $S^{(0)}_{|\mathbf{e}_n|}$ is initialised to zero). Finally, the standard deviation can be obtained from $S^{(t)}_{|\mathbf{e}_n|}$ as follows: $\sigma^{(t)}_{|\mathbf{e}_n|} = \sqrt{S^{(t)}_{|\mathbf{e}_n|}/(c(|\mathbf{e}_n|)^{(t)} - 1)}$.

7.5.3 Inverse and Direct Phrase-Based Models (h_3 and h_4)

Feature functions h_3 and h_4 implement inverse and direct phrase-based models respectively. These phrase-based models are combined with HMM-based alignment models via linear interpolation. In this thesis we have not studied how to incrementally update the weights of the interpolation. Instead, these weights can be estimated from a development corpus.

Since phrase-based models are symmetric models, only an inverse phrase-based model is maintained (direct probabilities can be efficiently obtained using appropriate data structures, see section 4.3.2). The inverse phrase model probabilities are estimated from phrase counts as follows:

$$p(\tilde{f}|\tilde{e}) = \frac{c(\tilde{f},\tilde{e})}{\sum_{\tilde{f}'} c(\tilde{f}',\tilde{e})}$$

According to the previous equation, the set of sufficient statistics to be stored for the inverse phrase model consists of a set of phrase counts, $c(\tilde{f},\tilde{e})$.

Given the n-th training pair $(\mathbf{f}_n, \mathbf{e}_n)$, the standard phrase-based model estimation method uses a word alignment matrix, A, between $\mathbf{f}_n$ and $\mathbf{e}_n$ to extract the set of phrase pairs that are *consistent* with the word alignment matrix: $\mathcal{BP}(\mathbf{f}_n, \mathbf{e}_n, A)$ (see section 3.2 for more details). Once the consistent phrase pairs have been extracted, the phrase counts are updated as follows:

$$c(\tilde{f},\tilde{e})^{(t)} = c(\tilde{f},\tilde{e})^{(t-1)} + c(\tilde{f},\tilde{e}\,|\,\mathcal{BP}(\mathbf{f}_n,\mathbf{e}_n,A)) \tag{7.16}$$

where $c(\tilde{f},\tilde{e})^{(t)}$ is the current count of the phrase pair $(\tilde{f},\tilde{e})$, $c(\tilde{f},\tilde{e})^{(t-1)}$ is the previous count, and $c(\tilde{f},\tilde{e}\,|\,\mathcal{BP}(\mathbf{f}_n,\mathbf{e}_n,A))$ is the count of $(\tilde{f},\tilde{e})$ in $\mathcal{BP}(\mathbf{f}_n,\mathbf{e}_n,A)$.

After updating the phrase counts, we need to efficiently compute the phrase translation probabilities. For this purpose, we maintain in memory both the current phrase counts and their normalisers.

One problem to be solved when updating the phrase model parameters is the need of generating word alignment matrices. To solve this problem, we use the direct and inverse HMM-based alignment models that are included in the formulation of the IMT system. Specifically, these models are used to obtain word alignments in both translation directions. The resulting direct and inverse word alignment matrices are combined by means of the *symmetrisation* alignment operation [ON03] before extracting the set of consistent phrase pairs.

In order to obtain an IMT system able to robustly learn from user feedback, we also need to incrementally update the HMM-based alignment models. In the following section we show how to efficiently incorporate new knowledge to these models.

7.5.4 Inverse and Direct HMM-Based Alignment Models (h_3 and h_4)

HMM-based alignment models play a crucial role in log-linear components h_3 and h_4, since they are used to smooth phrase-based models and to generate word alignment matrices. HMM-based alignment models were chosen here because, according to [ON03] and [TIM02], they outperform IBM 1 to IBM 4 alignment models while still allowing the exact calculation of the loglikelihood for a given sentence pair. However, our proposal is not restricted to the use of HMM-based alignment models.

The standard estimation procedure for HMM-based alignment models is carried out by means of the EM algorithm. However, the standard EM algorithm is not appropriate to incrementally extend our HMM-based alignment models because it is designed to work in batch training scenarios. To solve this problem, the incremental view of the EM algorithm described in section 7.3.1 can be applied.

Model Definition

HMM-based alignment models are a class of single-word alignment models. Single-word alignment models were introduced in section 1.4.2; for the reader's convenience, we describe again those concepts that are relevant to the definition of HMM-based alignment models.

Single-word alignment models are based on the concept of alignment between word positions of the source and the target sentences f_1^J and e_1^I. Specifically, the alignment is defined as a function $a : \{1 \cdots J\} \rightarrow \{0 \cdots I\}$, where $a_j = i$ if the j'th source position is aligned with the i'th target position. Additionally, $a_j = 0$ notes that the word position j of f_1^J has not been aligned with any word position e_1^I (or that it has been aligned with the *null word* e_0). Let $\mathcal{A}(f_1^J, e_1^I)$ be the set of all possible alignments between e_1^I and f_1^J, we formulate $Pr(f_1^J|e_1^I)$ in terms of the alignment variable as follows (Equation (1.17)):

$$Pr(f_1^J|e_1^I) = \sum_{a_1^J \in \mathcal{A}(f_1^J, e_1^I)} Pr(f_1^J, a_1^J|e_1^I) \tag{7.17}$$

Under a generative point of view, $Pr(f_1^J, a_1^J|e_1^I)$ can be decomposed without loss of generality as follows (Equation (1.18)):

$$Pr(f_1^J, a_1^J|e_1^I) = Pr(J|e_1^I) \cdot \prod_{j=1}^{J} Pr(f_j|f_1^{j-1}, a_1^j, e_1^I) \cdot Pr(a_j|f_1^{j-1}, a_1^{j-1}, e_1^I) \tag{7.18}$$

HMM-based alignment models are very similar to IBM models, specifically, they only differ in the assumptions made over the alignment probabilities. HMM-based alignment models use a first order alignment model $p(a_j|a_{j-1}, I)$ to approximate the distribution $Pr(a_j|f_1^{j-1}, a_1^{j-1}, e_1^I)$ and a word-to-word lexical model $p(f_j|e_{a_j})$ to approximate the distribution $Pr(f_j|f_1^{j-1}, a_1^j, e_1^I)$, resulting in the expression:

$$p(f_1^J, a_1^J|e_1^I, \Theta) = \prod_{j=1}^{J} p(f_j|e_{a_j}) \cdot p(a_j|a_{j-1}, I) \tag{7.19}$$

where we assume that a_0 is equal to zero and

$$\Theta = \begin{cases} p(f|e) & \forall\, f \in \mathcal{F} \text{ and } e \in \mathcal{E} \\ p(i|i', I) & 1 \leq i \leq I,\, 0 \leq i' \leq I \text{ and } \forall\, I \end{cases} \tag{7.20}$$

is the set of hidden parameters. For the sake of simplicity, we do not allow alignments with the null word, i.e. $i \neq 0$. This corresponds to the so-called *homogeneous* HMM-based alignment models defined in [VNT96]. The treatment of the null word can be easily introduced as it is shown in [ON03].

Incremental EM Algorithm

We follow the incremental EM iteration of (7.8). This incremental EM iteration is defined in terms of a set of sufficient statistics summarising the inferential import of the complete data.

As a preliminary step, and following the same derivation strategy that is presented in [Civ08] for the batch EM algorithm, we change the nature of the original alignment variable $a_j \in \{0, \ldots, I\}$ from an integer value to an indicator vector:

$$a_j = (a_{j0}, a_{j1}, \ldots, a_{jI}) \tag{7.21}$$

The vector a_j takes the value of one in the i'th position and zeros elsewhere if the source position j is aligned to the target position i.

Equation (7.19) can be reexpressed in terms of indicator vectors as follows:

$$p(f_1^J, a_1^J | e_1^I, \Theta) = \prod_{j=1}^{J} \prod_{i=1}^{I} p(f_j|e_i)^{a_{ij}} \prod_{i'=1}^{I} p(i|i', I)^{a_{j-1i'}a_{ij}} \tag{7.22}$$

with $a_{00} = 1$.

The complete data set, $\mathcal{Z} = (\mathcal{X}, \mathcal{Y})$ comprises the observed data, $\mathcal{X} = \{(\mathbf{f}_1, \mathbf{e}_1), ..., (\mathbf{f}_N, \mathbf{e}_N)\}$, which is composed of the training sentence pairs; and the hidden data, $\mathcal{Y} = \{\mathbf{a}_1, ..., \mathbf{a}_N\}$, which is composed of the alignment vectors associated with the sentence pairs of $\mathcal{X}$. The loglikelihood function of the complete data, $\mathcal{L}(\Theta, \mathbf{f}, \mathbf{e}, \mathbf{a})$, is defined as follows:

$$\mathcal{L}(\Theta, \mathbf{f}, \mathbf{e}, \mathbf{a}) = \sum_{n=1}^{N} \log p(\mathbf{f}_n, \mathbf{a}_n | \mathbf{e}_n, \Theta) \tag{7.23}$$

It can be demonstrated by means of the Fisher-Neyman factorisation theorem that $s(\mathbf{f}, \mathbf{e}, \mathbf{a}) = \sum_n s_n(\mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n)$ constitutes a vector of sufficient statistics for the model parameters, where $s_n(\mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n)$ is the vector of sufficient statistics for data item n:

$$s_n(\mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n) = \begin{cases} c(f|e; \mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n) & \forall\, f \in \mathcal{F} \text{ and } e \in \mathcal{E} \\ c(i|i', I; \mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n) & 1 \leq i \leq I,\, 0 \leq i' \leq I \text{ and } \forall\, I \end{cases} \tag{7.24}$$

with

$$c(f|e;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n) = \sum_{j:\mathbf{f}_{nj}=f}^{|\mathbf{f}_n|}\;\sum_{i:\mathbf{e}_{ni}=e}^{|\mathbf{e}_n|} \mathbf{a}_{nji} \tag{7.25}$$

$$c(i|i',I;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n) = \delta(I,|\mathbf{e}_n|)\sum_{j=1}^{|\mathbf{f}_n|}(\mathbf{a}_{n(j-1)i'}\,\mathbf{a}_{nji}) \tag{7.26}$$

being $c(f|e;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)$ the number of times that the word e is aligned to the word f for the sentence pair $(\mathbf{f}_n,\mathbf{e}_n)$; and $c(i|i',I;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)$ the number of times that the alignment i has been seen after the previous alignment i' given a source sentence composed of I words for the sentence pair $(\mathbf{f}_n,\mathbf{e}_n)$.

The log-likelihood of the complete data can be expressed in terms of the sufficient statistics as follows:

$$\begin{aligned}\mathcal{L}(\Theta,\mathbf{f},\mathbf{e},\mathbf{a}) = &\sum_{f\in\mathcal{F}}\sum_{e\in\mathcal{E}}\sum_{n=1}^{N} c(f|e;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)\cdot\log p(f|e)+\\ &\sum_{\forall I}\sum_{i=1}^{I}\sum_{i'=0}^{I}\sum_{n=1}^{N} c(i|i',I;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)\cdot\log p(i|i',I)\end{aligned} \tag{7.27}$$

To implement the E step of the incremental EM algorithm, we need to obtain the expected value at instant t of the sufficient statistics given the probability distribution of the hidden alignment variable: $\tilde{s}_n^{(t)} = E_{q_n}[s_n(\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)]$, where $q_n(\mathbf{a}_n) = p(\mathbf{a}_n|\mathbf{f}_n,\mathbf{e}_n,\Theta^{(t-1)})$. For this purpose, the counts given by equations (7.25) and (7.26) are replaced by expected counts:

$$c(f|e;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)^{(t)} = \sum_{j:\mathbf{f}_{nj}=f}^{|\mathbf{f}_n|}\;\sum_{i:\mathbf{e}_{ni}=e}^{|\mathbf{e}_n|} \mathbf{a}_{nji}^{(t)} \tag{7.28}$$

$$c(i|i',I;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)^{(t)} = \delta(I,|\mathbf{e}_n|)\sum_{j=1}^{|\mathbf{f}_n|}(\mathbf{a}_{n(j-1)i'}\,\mathbf{a}_{nji})^{(t)} \tag{7.29}$$

where

$$\mathbf{a}_{nji}^{(t)} = \frac{\alpha_{nji}\cdot\beta_{nji}}{\sum\limits_{\tilde{\imath}=1}^{|\mathbf{e}_n|}\alpha_{nj\tilde{\imath}}\cdot\beta_{nj\tilde{\imath}}} \tag{7.30}$$

$$(\mathbf{a}_{n(j-1)i'}\,\mathbf{a}_{nji})^{(t)} = \frac{\alpha_{n(j-1)i'}\cdot p(i|i',|\mathbf{e}_n|)^{(t-1)}\cdot p(\mathbf{f}_{nj}|\mathbf{e}_{ni})^{(t-1)}\cdot\beta_{nji}}{\sum\limits_{\tilde{\imath}'=1}^{|\mathbf{e}_n|}\sum\limits_{\tilde{\imath}=1}^{|\mathbf{e}_n|}\alpha_{n(j-1)\tilde{\imath}'}\cdot p(\tilde{\imath}\,|\,\tilde{\imath}',|\mathbf{e}_n|)^{(t-1)}\cdot p(\mathbf{f}_{nj}|\mathbf{e}_{n\tilde{\imath}})^{(t-1)}\cdot\beta_{nj\tilde{\imath}}} \tag{7.31}$$

being $\mathbf{a}_{nji}^{(t)}$, the posterior probability of aligning the source position j to the target position i at the current instant t ; and $(\mathbf{a}_{n(j-1)i'}\,\mathbf{a}_{nji})^{(t)}$, the posterior probability of aligning the source position $j-1$ to the target position i' and the position j to the position i for the n'th sample at the current instant t. The recursive functions α and β are defined as follows:

$$\alpha_{nji} = \begin{cases} 1 & j = 0 \\ p(i|0, |\mathbf{e}_n|)^{(t-1)} \cdot p(\mathbf{f}_{nj}|\mathbf{e}_{ni})^{(t-1)} & j = 1 \\ \sum\limits_{\tilde{\imath}=1}^{|\mathbf{e}_n|} \alpha_{n(j-1)\tilde{\imath}} \cdot p(i|\tilde{\imath}, |\mathbf{e}_n|)^{(t-1)} \cdot p(\mathbf{f}_{nj} \,|\, \mathbf{e}_{ni})^{(t-1)} & j > 1 \end{cases} \tag{7.32}$$

$$\beta_{nji} = \begin{cases} 1 & j = |\mathbf{f}_n| \\ \sum\limits_{\tilde{\imath}=1}^{|\mathbf{e}_n|} p(\tilde{\imath}|i, |\mathbf{e}_n|)^{(t-1)} \cdot p(\mathbf{f}_{n(j+1)} \,|\, \mathbf{e}_{n\tilde{\imath}})^{(t-1)} \cdot \beta_{n(j+1)\tilde{\imath}} & j < |\mathbf{f}_n| \end{cases} \tag{7.33}$$

Regarding the M step, we have to obtain the set of parameters that maximises the log-likelihood of the complete data given the expected values of the sufficient statistics. For this purpose, we replace the sufficient statistics in Equation (7.27) by their expected values at instant t, and then maximise the resulting expression (which corresponds to the $Q(\Theta|\Theta^{t-1})$ function expressed in terms of the sufficient statistics), obtaining the following update equations:

$$p(f|e)^{(t)} = \frac{\sum\limits_{n=1}^{N} c(f|e; \mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n)^{(t)}}{\sum\limits_{f' \in \mathcal{F}} \sum\limits_{n=1}^{N} c(f'|e; \mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n)^{(t)}} \tag{7.34}$$

$$p(i|i', I)^{(t)} = \frac{\sum\limits_{n=1}^{N} c(i|i', I; \mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n)^{(t)}}{\sum\limits_{\tilde{\imath}=1}^{I} \sum\limits_{n=1}^{N} c(\tilde{\imath}|i', I; \mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n)^{(t)}} \tag{7.35}$$

In the previous equations, the numerator values constitute the cumulative sufficient statistics $\tilde{s}^{(t)} = \sum_n \tilde{s}_n^{(t)}$ for the model parameters.

Incremental Update Rule

Given the n-th training pair $(\mathbf{f}_n, \mathbf{e}_n)$, the incremental update equation for the cumulative sufficient statistics, $\tilde{s}^{(t)}$, is given by the following expression:

$$\tilde{s}^{(t)} = \tilde{s}^{(t-1)} + \tilde{s}_n^{(t)} \tag{7.36}$$

It is worth mentioning that the sufficient statistics for a given sentence pair are nonzero for a small fraction of its components. As a result of this, the time required to update the parameters of the HMM-based alignment model depends only on the number of nonzero components.

Once the cumulative sufficient statistics, $\tilde{s}^{(t)}$, have been updated, we need to efficiently compute the model parameters. For this purpose, the normaliser factors for $\tilde{s}^{(t)}$ are also maintained.

The parameters of the direct HMM-based alignment model are estimated analogously to those of the inverse model.

7.5.5 Source Phrase Length, Target Phrase Length and Distortion Models (h_5, h_6 and h_7)

The δ parameters of the geometric distributions associated to the feature functions h_5, h_6 and h_7 are left fixed. Because of this, there are no sufficient statistics to store for these feature functions.

7.6 Summary

In this chapter, an IMT system in which the standard batch learning paradigm is replaced by the online learning paradigm has been proposed. Online learning is particularly useful in IMT since it allows to feed the models used by the IMT system with the translations validated by the user.

Our proposed IMT system implements online learning by means of incremental learning algorithms. Incremental learning algorithms are a kind of online learning algorithms that discard each new training sample after updating the learner.

Our online IMT system uses incremental learning algorithms to update the parameters of the statistical models involved in the translation process. For this purpose, we need to maintain a set of sufficient statistics that can be incrementally updated for these models.

If the estimation of a given statistical model does not require the use of the EM algorithm, then it is generally easy to incrementally extend the model given a new training sample. By contrast, if the EM algorithm is required, the estimation procedure has to be modified, since the conventional EM algorithm is designed for its use in batch learning scenarios. To solve this problem, we have applied the incremental view of the EM algorithm described in [NH98].

We provided a complete set of update equations and algorithms that allow us to obtain an incrementally updateable IMT system, breaking technical limitations encountered in other works.

CHAPTER 8
PB-IMT EVALUATION

In this chapter we show the results of the experiments that we carried out to test the three IMT techniques proposed in chapters 6 and 7, namely, IMT based on partial phrase-based alignments, IMT based on stochastic error-correction models and IMT with online learning[a].

8.1 IMT based on Partial Phrase-Based Alignments

We carried out experiments to test our proposed IMT system based on partial phrase-based alignments described in section 6.2. In some experiments, this IMT system uses a monotonic version of the expansion algorithm given by Algorithm 6.1. Monotonic IMT systems are useful because of their lower response times, but this time complexity reduction usually comes at the cost of poorer translation results. The language model used by the IMT system was implemented as a standard backoff language model, which was estimated by means of the SRILM toolkit [Sto02]. Regarding the inverse and direct smoothed phrase-based models, they were obtained by means of the THOT toolkit presented in Appendix B of this thesis.

The experiments were performed using the Xerox and the EU corpora, which were described in section 1.10. We evaluated our proposed techniques by means of the KSMR (key stroke and mouse action ratio) measure described in section 1.9.3. In addition to this, in some experiments we also assessed the performance of our proposed IMT system with respect to using a conventional SMT system followed by human post-editing. This assessment is done by comparing the KSR (key stroke ratio) measure obtained by the IMT system with the CER (character error rate) and PKSR (post-editing key stroke ratio) measures. As was explained in section 1.9.3, the CER measure constitutes a rough estimation of the post-editing effort, since professional translators typically use text editors with autocompletion capabilities to generate the target translations. To solve this problem, the PKSR measure defined in [RTV10] is used.

[a] In some experiments reported in this chapter we show the time cost of the proposed algorithms, all the experiments were executed on a PC with a 2.40 Ghz Intel Xeon processor with 1GB of memory.

8.1.1 Experiments with the Xerox Corpus

In Table 8.1, IMT results with the Xerox corpus using different phrase-to-phrase alignment smoothing techniques are presented, for three different language pairs and both translation directions. Geometric distributions were selected to implement both the h_5 (target phrase length model) and h_6 (source phrase length model) feature functions. The first row of the table shows the baseline, which consists of the results obtained using maximum-likelihood estimation without smoothing (ML). The rows labelled with GT (Good-Turing), AD (absolute-discount), KN (Kneser-Ney) and SD (simple discount) show the results for the phrase-based model estimators presented in section 4.4.2. The rest of the rows corresponds to different estimation techniques combined with the lexical distribution (LEX) by means of linear interpolation (LI), backing-off (BO), and log-linear interpolation (LL). Because of the great number of possible configurations of the IMT system and the high time cost of the MERT algorithm, we used a monotonic IMT system with default values for the weights of the log-linear model to carry out the experiments.

Table 8.1: KSMR results for the three Xerox corpora (for both direct and inverse translation directions separated by the symbol "/") for a monotonic IMT system and different smoothing techniques. Geometric distributions were selected to implement the h_5 and h_6 feature functions. Default weights for the log-linear model were used. Best results are shown in bold.

Smooth.	**Spa-Eng**	**Fre-Eng**	**Ger-Eng**
ML	36.7/32.5	59.4/53.2	63.6/57.2
GT	28.6/29.4	51.9/49.4	57.7/53.0
AD	30.3/28.1	50.4/46.7	58.4/52.5
KN	30.3/28.1	50.4/46.7	58.4/52.4
SD	28.5/29.4	51.6/49.2	57.1/52.5
ML+LEX$_{\mathbf{LI}}$	21.2/21.3	39.9/39.2	**43.9**/42.4
GT+LEX$_{\mathbf{LI}}$	**21.1**/21.3	39.9/39.2	44.2/42.2
AD+LEX$_{\mathbf{LI}}$	21.4/22.2	40.2/40.5	45.1/42.2
KN+LEX$_{\mathbf{LI}}$	21.5/22.2	40.1/40.5	45.0/42.2
SD+LEX$_{\mathbf{LI}}$	21.2/21.2	39.9/**39.0**	44.0/**41.8**
GT+LEX$_{\mathbf{BO}}$	**21.1**/**21.0**	**39.8**/**39.0**	45.3/42.3
SD+LEX$_{\mathbf{BO}}$	21.2/**21.0**	**39.8**/39.2	45.1/42.3
ML+LEX$_{\mathbf{LL}}$	37.5/35.5	59.5/53.7	64.3/58.0
GT+LEX$_{\mathbf{LL}}$	24.0/25.8	43.2/43.3	50.9/46.9
AD+LEX$_{\mathbf{LL}}$	30.8/29.2	51.3/46.9	59.7/52.1
KN+LEX$_{\mathbf{LL}}$	30.9/29.1	51.4/46.9	59.7/52.0
SD+LEX$_{\mathbf{LL}}$	23.6/27.7	43.2/42.7	50.7/45.9

According to the table, the baseline system obtained by far the worst results. In contrast, all those experiments that included the LEX distribution outperformed the others due to improved assignment of probabilities to unseen events. As was expected (see section 4.4.2), linear interpolation and backing-off obtained better results than log-linear interpolation. Additionally, GT and SD statistical estimators worked slightly better than the rest of estimators.

We also carried out experiments to study the impact of the different probability distributions used for the feature functions h_5 (target phrase length model) and h_6 (source phrase length model) in the accuracy of our system. Table 8.2 reports the KSMR results for all possible combinations of the probability distributions used for h_5 (Uniform (U) and Geometric (G)) and for h_6 (Uniform (U), Geometric (G), and Poisson (P)). Only the results obtained for the best smoothing technique (Good-Turing) are reported (the results corresponding to other smoothing techniques were similar). Again, a monotonic IMT system with default log-linear weights were used.

Table 8.2: KSMR results for the three Xerox corpora (for both direct and inverse translation directions separated by the symbol "/") for all possible combinations of the probability distributions for the h_5 and h_6 feature functions when using two different smoothing techniques. A monotonic IMT system with default log-linear model weights were used. Best results are shown in bold.

Smooth.	h_5, h_6	**Spa-Eng**	**Fre-Eng**	**Ger-Eng**
GT	U,U	30.1/29.0	53.8/50.7	58.0/53.9
	U,P	29.5/28.6	52.9/49.7	57.6/53.4
	U,G	28.7/28.0	51.7/48.7	57.3/52.7
	G,U	30.5/29.7	54.6/51.5	58.5/54.4
	G,P	29.7/29.4	53.3/50.5	58.2/53.7
	G,G	28.6/29.4	51.9/49.4	57.7/53.0
GT+ LEX$_{\text{BO}}$	U,U	21.8/21.6	40.4/39.1	44.8/42.2
	U,P	21.5/21.4	40.2/39.0	44.3/42.0
	U,G	21.3/21.4	40.1/**38.8**	**44.0**/**41.8**
	G,U	21.6/21.5	40.3/39.1	44.6/42.1
	G,P	21.4/21.3	40.0/39.0	44.2/41.9
	G,G	**21.1**/**21.0**	**39.8**/39.0	45.3/42.3

As can be seen in the table, slight KSMR differences are obtained. The best results were obtained when U+G distributions were used for the GT estimator, and G+G for the BO combination. As was mentioned in section 6.2.2, the use of a uniform distribution for h_5 penalises the length of the bisegmentation and the use of a geometric distribution penalises the length of the source phrases. Correspondingly, the use of a geometric distribution for h_6 makes it possible to establish a relationship between the length of source and target phrases (the use of a Poisson distribution also worked well).

IMT results for the three considered corpora (for both translation directions) are shown in Table 8.3. MERT for the development corpus was performed to adjust the weights of the log-linear model. In this case, only the GT+LEX$_{\text{BO}}$, the SD+LEX$_{\text{BO}}$ and the SD+LEX$_{\text{LI}}$ smoothing techniques were tested, obtaining very similar results. The last column of Table 8.3 shows the average time in seconds per iteration needed to complete a new translation given a user validated prefix. Clearly, these times allow the system to work on a real time scenario.

We also carried out experiments using a non-monotonic IMT system with GT+LEX$_{\text{BO}}$ smoothing. Table 8.4 shows the KSMR results and the time cost in seconds per each interaction when translating the Xerox corpora from English to Spanish, French and German.

Table 8.3: KSMR results for the three Xerox corpora, using a monotonic IMT system with three different smoothing techniques. Geometric distributions were used to implement the h_5 and h_6 feature functions. MERT was performed. The average time in seconds per interaction is also reported.

Corpus	**Smooth.**	**KSMR**	**s/inter.**
Spa–Eng	**GT+LEX$_{\mathbf{BO}}$**	19.6	0.086
	SD+LEX$_{\mathbf{BO}}$	19.6	0.090
	SD+LEX$_{\mathbf{LI}}$	19.7	0.090
Eng–Spa	**GT+LEX$_{\mathbf{BO}}$**	17.5	0.093
	SD+LEX$_{\mathbf{BO}}$	17.6	0.094
	SD+LEX$_{\mathbf{LI}}$	17.9	0.106
Fre–Eng	**GT+LEX$_{\mathbf{BO}}$**	36.9	0.204
	SD+LEX$_{\mathbf{BO}}$	37.0	0.205
	SD+LEX$_{\mathbf{LI}}$	37.4	0.242
Eng–Fre	**GT+LEX$_{\mathbf{BO}}$**	34.4	0.148
	SD+LEX$_{\mathbf{BO}}$	34.4	0.147
	SD+LEX$_{\mathbf{LI}}$	34.1	0.211
Ger–Eng	**GT+LEX$_{\mathbf{BO}}$**	39.5	0.170
	SD+LEX$_{\mathbf{BO}}$	39.5	0.184
	SD+LEX$_{\mathbf{LI}}$	39.7	0.237
Eng–Ger	**GT+LEX$_{\mathbf{BO}}$**	39.1	0.152
	SD+LEX$_{\mathbf{BO}}$	39.2	0.154
	SD+LEX$_{\mathbf{LI}}$	39.2	0.210

Table 8.4: KSMR results for the three Xerox corpora, using a non-monotonic IMT system with GT+LEX$_{\mathrm{BO}}$ smoothing. Geometric distributions were used to implement the h_5 and h_6 feature functions. MERT was performed. The average time in seconds per interaction is also reported.

Corpus	**KSMR**	**s/inter.**
Eng–Spa	16.7	0.283
Eng–Fre	34.9	0.388
Eng–Ger	38.6	0.405

Geometric distributions were used to implement the h_5 and h_6 feature functions, the weights of the log-linear combination were adjusted by means of the MERT algorithm. As can be seen in the table, the non-monotonic IMT system obtains better KSMR results than those reported in Table 8.3 for the monotonic IMT system. However, these improved results are obtained with higher time costs per each interaction of the IMT process.

Additionally, we performed experiments to estimate the human effort reduction that can be obtained using our proposed IMT system with respect to using the post-editing approach. For this purpose, we compared the KSR measure obtained by our IMT system with the CER and PKSR measures. Table 8.5 shows the obtained results. According to the results, the

Table 8.5: CER and PKSR obtained with the post-editing approach and KSR obtained with our proposed IMT system based on smoothing techniques (a monotonic IMT system with GT+LEX$_{\mathrm{BO}}$ smoothing tuned with MERT was used, geometric distributions were selected to implement the h_5 and h_6 feature functions). The results were obtained for the Xerox corpora.

Corpus	**CER**	**PKSR**	**KSR**
Eng–Spa	22.3	17.3	10.0
Eng–Fre	48.1	35.6	21.7
Eng–Ger	55.3	39.5	25.1

Table 8.6: KSMR results comparison of our IMT system based on partial statistical phrase-based alignments (a monotonic IMT system with GT+LEX$_{\mathrm{BO}}$ smoothing tuned with MERT was used, geometric distributions were selected to implement the h_5 and h_6 feature functions) and three different state-of-the art IMT systems. 95% confidence intervals are shown. The experiments were executed on the Xerox corpora. Best results are shown in bold.

Corpus	**AT**	**PB**	**SFST**	**PSPBA**
Spa–Eng	24.0±1.3	**18.1±1.2**	26.9±1.3	19.6±1.1
Eng–Spa	23.2±1.3	**16.7±1.2**	21.8±1.4	17.6±1.1
Fre–Eng	40.5±1.4	37.2±1.3	45.5±1.3	**37.0±1.4**
Eng–Fre	40.4±1.4	35.8±1.3	43.8±1.6	**34.4±1.2**
Ger–Eng	45.9±1.2	**36.7±1.2**	46.6±1.4	39.5±1.1
Eng–Ger	44.7±1.2	40.1±1.2	45.7±1.4	**39.2±1.1**

estimated human effort to generate correct translations using our proposed IMT system is significantly reduced with respect to using the post-editing approach. As was expected, the values of the CER measure were greater than those of the PKSR measure for the three language pairs. This is due to the autocompletion capabilities that are involved in the calculation of the PKSR measure.

Finally, in Table 8.6 a comparison of the best results obtained by our IMT system based on partial statistical phrase-based alignments (PSPBA) with state-of-the-art IMT systems is reported (95% confidence intervals are shown). We compared our system with those presented in [BBC$^+$09]: the alignment templates (AT), the stochastic finite-state transducer (SFST), and the phrase-based (PB) approaches to IMT. As can be seen, our system obtains similar results and in some cases clearly outperforms the results obtained by these IMT systems. Specifically, our results were better than those obtained by the SFST and the AT systems. By contrast, the KSMR results with respect to the PB approach were similar.

8.1.2 Experiments with the EU Corpus

Additional experiments using the EU corpus were carried out to test the performance of our IMT system based on partial phrase-based alignments. Table 8.7 shows the obtained KSMR

Table 8.7: KSMR results for the three EU corpora, using a monotonic IMT system with GT+LEX$_{\mathrm{BO}}$ smoothing. Geometric distributions were used to implement the h_5 and h_6 feature functions. MERT was performed. The average time in seconds per interaction is also reported.

Corpus	KSMR	s/inter.
Spa–Eng	21.9	0.327
Fre–Eng	19.5	0.326
Ger–Eng	28.3	0.278

Table 8.8: CER and PKSR obtained with the post-editing approach and KSR obtained with our proposed IMT system based on smoothing techniques (a monotonic IMT system with GT+LEX$_{\mathrm{BO}}$ smoothing tuned with MERT was used, geometric distributions were selected to implement the h_5 and h_6 feature functions). The results were obtained for the EU corpora.

Corpus	CER	PKSR	KSR
Spa–Eng	36.6	25.5	13.1
Fre–Eng	32.7	23.2	11.6
Ger–Eng	45.0	30.6	17.4

results when translating from Spanish, French and German to the English language. All the results were obtained by means of a monotonic IMT system with GT+LEX$_{\mathrm{BO}}$ smoothing. The h_5 and h_6 feature functions were implemented by means of geometric distributions. The weights of the log-linear combination were obtained by using the MERT algorithm. The table also shows the average time in seconds required by each new interaction of the IMT process.

As can be seen in the table, the interactive translation from English to Spanish obtained the lowest KSMR measure. The time costs per interaction for the three language pairs were higher than those obtained when translating the Xerox corpora (see Table 8.3). This is due to the fact that the EU corpora have greater training sets which produce substantially larger translation and language models.

Additional experiments were carried out to compare the performance of our proposed system with respect to that of the post-editing approach. Results are shown in Table 8.8. According to the table, post-editing significantly increased the required human effort with respect to the IMT system. This is the same situation that was observed for the Xerox corpora (see Table 8.5).

Finally, Table 8.9 shows a comparison of the KSMR results (95% confidence intervals are shown) that were obtained by our proposed IMT system based on partial phrase-based alignments (PSPBA), with respect to those obtained by state-of-the-art IMT systems. As was explained in section 8.1.1, these state-of-the-art IMT systems are based on different translation technologies, including alignment templates (AT), stochastic finite-state transducers (SFST) and phrase-based models (PB). In the table we show the results obtained by a monotonic IMT system with GT+LEX$_{\mathrm{BO}}$ smoothing tuned with MERT. As can be seen, our system

Table 8.9: KSMR results comparison of our IMT system based on partial statistical phrase-based alignments (a monotonic IMT system with GT+LEX$_{\mathrm{BO}}$ smoothing tuned with MERT was used, geometric distributions were selected to implement the h_5 and h_6 feature functions) and three different state-of-the art IMT systems. 95% confidence intervals are shown. The experiments were executed on the EU corpora. Best results are shown in bold.

Corpus	**AT**	**PB**	**SFST**	**PSPBA**
Spa–Eng	33.3±1.3	23.8±1.0	31.1±1.3	**21.9±1.0**
Fre–Eng	28.6±1.2	21.5±1.0	28.0±1.2	**19.5±0.9**
Ger–Eng	38.1±1.4	31.7±1.0	39.1±1.5	**28.2±1.2**

outperforms the results obtained by these IMT systems in all cases.

8.2 IMT based on Stochastic Error-Correction Models

We carried out experiments to test our IMT system based on stochastic error-correction models described in section 6.3. The initial word graph for each source sentence was generated using a regular SMT system. This SMT system uses the log-linear model described in section 3.5.3. The components of the log-linear combination where instantiated as follows: a standard backoff language model estimated by means of the SRILM toolkit was used to implement h_1; h_2 was implemented by means of a set of normal distributions; inverse and direct phrase-based models without smoothing generated by means of the THOT toolkit were used to implement h_3 and h_4, respectively; the target phrase length model, h_5, was implemented by means of a geometric distribution; finally, geometric distributions were used to implement the source phrase length and the distortion models, h_6 and h_7, respectively.

The experiments were performed using the Xerox corpora and the EU corpora as well as in the previous section. We also used the same evaluation measures, including KSMR and KSR compared with CER and PKSR.

8.2.1 Experiments with the Xerox Corpus

In Table 8.10 the IMT results for the Xerox corpora (for the three language pairs and both translation directions) using our proposed IMT system based on stochastic error-corrections models are shown. A monotonic SMT system was used to generate the word graphs that are required during the IMT process. MERT for the development corpus was performed to adjust the weights of the log-linear model. The last column of Table 8.10 shows the average time in seconds per iteration needed to complete a new translation given a user validated prefix. These times allow the system to work on a real time scenario, and they are even lower than those obtained by the IMT system based on partial phrase-based alignments given in Table 8.3. By contrast, the obtained KSMR results are slightly worse. This is because the greater simplicity of the word-graph based IMT system, which carries out only one translation process at the first interaction of the interactive translation of each source sentence.

Table 8.10: KSMR results for the three Xerox corpora, using an IMT system based on stochastic error-correction models. Word graphs were generated by means of a monotonic SMT system. MERT was performed. The average time in seconds per interaction is also reported.

Corpus	**KSMR**	**s/inter.**
Spa–Eng	21.2	0.010
Eng–Spa	19.8	0.010
Fre–Eng	41.2	0.012
Eng–Fre	37.5	0.013
Ger–Eng	41.0	0.012
Eng–Ger	42.7	0.012

Table 8.11: KSMR results for the three Xerox corpora, using an IMT system based on stochastic error-correction models. Word graphs were generated by means of a non-monotonic SMT system. MERT was performed. The average time in seconds per interaction is also reported.

Corpus	**KSMR**	**s/inter.**
Eng–Spa	19.3	0.048
Eng–Fre	36.9	0.100
Eng–Ger	42.2	0.084

We also performed non-monotonic experiments. Table 8.11 shows the KSMR results when translating the Xerox test corpora from English to the other three languages. Word graphs were generated by means of a non-monotonic SMT system. The weights of the IMT system were tuned using the MERT algorithm. The average time costs per each interaction are also shown. As can be seen, slight improvements with respect to the monotonic IMT system can be obtained, at the cost of higher interaction times.

We carried out experiments to compare the performance of our proposed IMT system based on stochastic error-correction models with that of the post-editing approach. Table 8.12 shows the obtained results. According to the table, our proposed IMT system allowed us to significantly reduce the required human effort with respect to post-editing the output of an SMT system.

Finally, in Table 8.13 a comparison of the results obtained by our phrase-based IMT system based on stochastic error-correction models (PB-SECM) with state-of-the-art IMT systems is reported (95% confidence intervals are shown). As in previous sections, the comparison includes the KSMR results obtained by IMT systems using different translation technologies, namely, alignment templates (AT), stochastic finite-state transducer (SFST), and phrase-based models (PB). Additionally, we also show the results of the IMT system based on partial phrase-based alignments (PSPBA) that were reported in the previous section. As can be seen, our system is competitive with the SFST and the AT systems but underperforms the results obtained by the PB and PSPBA IMT systems. It is worth mentioning that the

Table 8.12: CER and PKSR obtained with the post-editing approach and KSR obtained with our proposed IMT system based on error-correction models (word graphs were generated by means of a monotonic SMT system). The results were obtained for the Xerox corpora.

Corpus	**CER**	**PKSR**	**KSR**
Eng–Spa	21.6	16.8	11.8
Eng–Fre	47.9	35.6	24.5
Eng–Ger	55.2	39.1	28.3

AT and the SFST systems are also based on word graphs and error-correction techniques to generate the suffixes required in IMT, as well as our proposed IMT system; specifically, these systems obtain the translation of minimum edit distance to the given prefix. By contrast, the PB and PSPBA IMT systems generate a new translation of the source sentence at each interaction of the IMT process instead of generating a word graph at the beginning. This allows such IMT systems to obtain better results but generally with higher time costs per interaction.

Table 8.13: KSMR results comparison of our IMT system based on stochastic error-correction models and four different state-of-the art IMT systems (word graphs were generated by means of a monotonic SMT system). 95% confidence intervals are shown. The experiments were executed on the Xerox corpora. Best results are shown in bold.

Corpus	**AT**	**PB**	**SFST**	**PSPBA**	**PB-SECM**
Spa–Eng	24.0±1.3	**18.1**±**1.2**	26.9±1.3	19.6±1.1	21.2±1.2
Eng–Spa	23.2±1.3	**16.7**±**1.2**	21.8±1.4	17.6±1.1	19.8±1.3
Fre–Eng	40.5±1.4	37.2±1.3	45.5±1.3	**37.0**±**1.4**	41.1±1.4
Eng–Fre	40.4±1.4	35.8±1.3	43.8±1.6	**34.4**±**1.2**	37.5±1.2
Ger–Eng	45.9±1.2	**36.7**±**1.2**	46.6±1.4	39.5±1.1	41.0±1.2
Eng–Ger	44.7±1.2	40.1±1.2	45.7±1.4	**39.2**±**1.1**	42.7±1.1

8.2.2 Experiments with the EU Corpus

We executed experiments on the EU corpora using our IMT system based on stochastic error-correction models. Table 8.14 shows the obtained KSMR results when translating from Spanish, French and German to the English language. The word graphs required in the IMT process were generated by means of a monotonic SMT system. The weights of the log-linear models were tuned via MERT. The table also shows the average time costs per each interaction of the interactive translation. Again, we observe worse KSMR results and lower time costs per interaction of the IMT system based on stochastic error-correction models with respect to the results obtained by the IMT system based on partial phrase-based alignments (see Table 8.7).

We also performed experiments to compare the results of our proposed IMT system with those of the post-editing approach. Table 8.15 shows the CER, PKSR and KSR results for the

Table 8.14: KSMR results for the three EU corpora, using an IMT system based on stochastic error-correction models. Word graphs were generated by means of a monotonic SMT system. MERT was performed. The average time in seconds per iteration is also reported.

Corpus	KSMR	s/inter.
Spa–Eng	26.9	0.021
Fre–Eng	23.2	0.027
Ger–Eng	31.9	0.024

Table 8.15: CER and PKSR obtained with the post-editing approach and KSR obtained with our proposed IMT system based on error-correction models (word graphs were generated by means of a monotonic SMT system). The results were obtained for the EU corpora.

Corpus	CER	PKSR	KSR
Spa–Eng	37.4	26.7	16.7
Fre–Eng	32.8	23.4	14.3
Ger–Eng	43.5	30.7	20.1

Table 8.16: KSMR results comparison of our IMT system based on stochastic error-correction models and four different state-of-the art IMT systems (word graphs were generated by means of a monotonic SMT system). 95% confidence intervals are shown. The experiments were executed on the EU corpora. Best results are shown in bold.

Corpus	AT	PB	SFST	PSPBA	PB-SECM
Spa–Eng	33.3±1.3	23.8±1.0	31.1±1.3	**21.9±1.0**	26.9±1.0
Fre–Eng	28.6±1.2	21.5±1.0	28.0±1.2	**19.5±0.9**	23.2±1.0
Ger–Eng	38.1±1.4	31.7±1.0	39.1±1.5	**28.2±1.2**	31.9±1.1

three EU test corpora. Again, the required human effort was lower for the interactive system.

As in previous sections, we also present a comparison between the results obtained by our IMT system and those obtained by state-of-the-art IMT systems following different translation approaches, including the alignment templates (AT), the stochastic finite-state transducers (SFST) and the phrase-based (PB) approaches to IMT. Again, we have also included the results obtained by our proposed IMT system based on partial phrase-based alignments (PSPBA). Table 8.16 shows the obtained KSMR results (95% confidence intervals are shown). Again, the PB and PSPBA IMT systems obtained the best results, and our PB-SECM IMT system outperformed the results of the AT and the SFST IMT systems, which are also based on error-correction techniques to generate the suffixes required in IMT.

8.3 IMT with Online Learning

This section describes the experiments that we carried out to test our proposed online IMT system presented in Chapter 7. In our experiments, the basic IMT system is restricted to obtain monotonic alignments between the source and the target sentences. The incremental language and phrase-based models involved in the interactive translation process were generated and accessed by means of a yet unpublished extension of the THOT toolkit presented in this thesis.

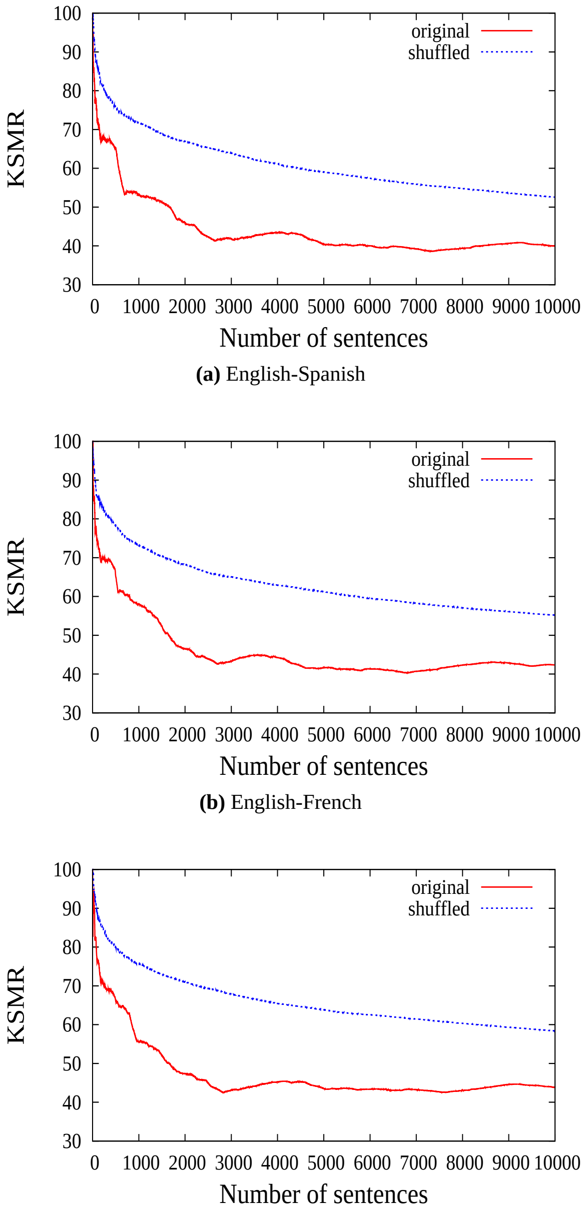
We evaluated our IMT system with online learning by means of the KSMR measure described in section 1.9.3. In addition to this, we also used the well-known BLEU score to measure the translation quality of the first translation hypothesis produced by the IMT system for each source sentence (which is automatically generated without user intervention).

8.3.1 Experiments with the Xerox Corpus

To test our proposed techniques, we carried out experiments with the Xerox corpora in two different scenarios. In the first one, the first 10 000 sentences extracted from the training corpora were interactively translated by means of an IMT system without any preexistent model stored in memory. Each time a new sentence pair was validated, it was used to incrementally train the system. To save computation time, monotonic search was used in all cases. Default values for the weights of the log-linear model were adopted. Figures 8.1a, 8.1b and 8.1c show the evolution of the KSMR with respect to the number of sentence pairs processed by the IMT system; the results correspond to the translation from English to Spanish, French and German, respectively. In addition to this, for each language pair we interactively translated the original portion of the training corpus and the same portion of the original corpus after being randomly shuffled.

As the above mentioned figures show, the results clearly demonstrate that the IMT system is able to learn from scratch. The results were similar for the three languages. It is also worthy of note that the obtained results were better in all cases for the original corpora than for the shuffled ones. This is because, in the original corpora, similar sentences appear more or less contiguously (due to the organisation of the contents of the printer manuals). This circumstance increases the accuracy of the online learning, since with the original corpora the number of *lateral effects* occurred between the translation of similar sentences is decreased. The online learning of a new sentence pair produces a lateral effect when the changes in the probability given by the models not only affect the newly trained sentence pair but also other sentence pairs. A lateral effect can cause that the system generates a wrong translation for a given source sentence due to undesired changes in the statistical models.

The accuracy were worse for shuffled corpora, since shuffling increases the number of lateral effects that may occur between the translation of similar sentences (because they no longer appear contiguously). These results illustrate the importance of the order in which knowledge is acquired when executing incremental learning algorithms that was mentioned in section 7.3. A good way to compare the quality of different online IMT systems is to determine their robustness in relation to sentence ordering. However, it can generally be expected that the sentences to be translated in an interactive translation session will be in a non-random order.



(a) English-Spanish

(b) English-French

(c) English-German

Figure 8.1: KSMR evolution translating a portion of the Xerox training corpora. A monotonic online IMT system with log-linear weights tuned via MERT was used.

Alternatively, we carried out experiments in a different learning scenario. Specifically, the Xerox test corpora were interactively translated from the English language to the other three languages, comparing the performance of a batch IMT system with that of an online IMT system. The batch IMT system is a conventional IMT system which is not able to take advantage of user feedback after each translation while the online IMT system uses the new sentence pairs provided by the user to revise the statistical models. Both systems were initialised with a log-linear model trained in batch mode by means of the Xerox training corpora. The weights of the log-linear combination were adjusted for the development corpora by means of the MERT algorithm.

Table 8.17 shows the obtained results. The table shows the BLEU score and the KSMR for the batch and the online IMT systems (95% confidence intervals are shown). Both systems used monotonic search. The log-linear weights were adjusted by means of the MERT algorithm. The BLEU score was calculated from the first translation hypothesis produced by the IMT system for each source sentence (see, for example, the initial interaction in Figure 1.3). The table also shows the average online learning time (LT) for each new sample presented to the system. All the improvements obtained with the online IMT system were statistically significant. Also, the average learning times clearly allow the system to be used in a real-time scenario.

Table 8.17: BLEU and KSMR results for the Xerox test corpora using the batch and the online IMT systems. Both systems used monotonic search with log-linear weights tuned via MERT. The average online learning time (LT) in seconds is shown for the online system.

Corpus	**IMT system**	**BLEU**	**KSMR**	**LT (s)**
Eng-Spa	batch	55.1± 2.3	18.2± 1.1	-
	online	60.6± 2.3	15.8± 1.0	0.04
Eng-Fre	batch	33.7± 2.0	33.9± 1.3	-
	online	42.2± 2.2	27.9± 1.3	0.09
Eng-Ger	batch	20.4± 1.8	40.3± 1.2	-
	online	28.0± 2.0	35.0± 1.3	0.07

Finally, as in previous sections, a comparison of the KSMR results (95% confidence intervals are shown) obtained by the online IMT system with state-of-the-art IMT systems is reported in Table 8.18. These IMT systems are based on different translation approaches, including the alignment templates (AT), the stochastic finite-state transducer (SFST), and the phrase-based (PB) approaches to IMT. Our system outperformed the results obtained by these systems.

8.3.2 Experiments with the EU Corpus

We executed additional experiments on the EU corpus to test the learning capabilities of our proposed online IMT system. In this case, the experimentation was restricted to the French-English language pair. In addition to this, only the second experimentation scenario described in section 8.3.1 was considered here.

Table 8.18: KSMR results comparison of our system and three different state-of-the-art batch systems. The experiments were executed on the Xerox corpora. Best results are shown in bold.

Corpus	**AT**	**PB**	**SFST**	**Online**
Eng-Spa	23.2±1.3	16.7±1.2	21.8±1.4	**15.8± 1.0**
Eng-Fre	40.4±1.4	35.8±1.3	43.8±1.6	**27.9± 1.3**
Eng-Ger	44.7±1.2	40.1±1.2	45.7±1.4	**35.0± 1.3**

Table 8.19 shows a comparison of the performance of a batch IMT system with that of an online IMT system when translating the French-English test set of the EU corpus. Both systems used monotonic search and were initialised with a log-linear model trained in batch mode by means of the EU training corpus. The weights of the log-linear model were adjusted via MERT. The table shows the BLEU score (calculated from the first translation hypothesis produced by the IMT system) and the KSMR measure for the batch and the online IMT systems (95% confidence intervals are shown). The table also shows the average online learning time (LT) for each new sample presented to the system. As can be seen in the table, the online IMT system is not able to improve the results obtained by the batch IMT system. The accuracy of online learning may depend of a series of factors, including the ordering of the test sentences, the presence or not of similar sentences that cannot be correctly translated by the IMT system or the size of the test set (online learning techniques are to be evaluated in the long term). We think that one possible reason to explain the results can be the small size of the EU test set, which is composed of only 800 sentences (below we show the results of an additional experiment that tries to validate this hypothesis). Regarding the learning time per sample, a small increase can be observed with respect to the learning times that were reported for the Xerox corpora (see Table 8.17), despite this, the proposed techniques can still be applied in a real time scenario.

Table 8.19: BLEU and KSMR results for the French-English EU test corpus using the batch and the online IMT systems. Both IMT systems used monotonic search. MERT was performed. The average online learning time (LT) in seconds is shown for the online system.

Corpus	**IMT system**	**BLEU**	**KSMR**	**LT (s)**
Fre-Eng	batch	47.6±2.0	21.5±1.3	-
	online	47.6±1.9	21.5±1.0	0.342

In order to determine the influence of the size of the test set in the performance of the online IMT system, we repeated the same experiment using an alternative partition of the French-English EU corpus. Specifically, the last 5 000 sentence pairs of the French-English EU training corpus were used as the new test set and all the previous ones as the new training set. Table 8.20 shows the BLEU score, the KSMR measure and the average learning time (LT) in seconds that were obtained during the interactive translation of the alternative test set using a batch and an online IMT system. Both systems used monotonic search and their log-linear weights were adjusted using the MERT algorithm.

Table 8.20: BLEU and KSMR results for an alternative partition of the French-English EU corpus using the batch and the online IMT systems. Both systems used monotonic search. The log-linear weights were tuned by means of the MERT algorithm. The average online learning time (LT) in seconds is shown for the online system.

Corpus	IMT system	BLEU	KSMR	LT (s)
Fre-Eng	batch	45.1±0.8	21.6±0.4	-
	online	46.5±0.8	20.9±0.4	0.380

As can be seen in the table, the online IMT system is now able to obtain slight improvements both in terms of BLEU and KSMR with respect to the batch IMT system. There are no significant differences between the obtained average learning time and that obtained while interactively translating the standard test corpus.

8.4 Summary

In this chapter we have empirically demonstrated that the concept of partial phrase-based alignment can be successfully used to implement IMT systems. The details of the proposal were described in section 6.2. The experiments we carried out show the great impact of the smoothing techniques in the accuracy of our system. The combination of a phrase-based model estimator with a lexical distribution yielded the best results. Three different combination techniques were tested: backing-off, linear interpolation and log-linear interpolation. As we expected, backing-off and linear interpolation worked better.

We have also compared the results obtained by our system with those obtained by state-of-the-art IMT systems. Our system obtained similar results and in some cases clearly outperformed the results obtained by the state-of-the-art systems.

In addition to this, we also carried out experiments to test the IMT system based on error-correction techniques described in section 6.3. The results of the experiments show that the IMT system based on error-correction techniques obtains worse results but is faster than the IMT system based on partial phrase-based alignments that was also proposed in this thesis. Again, we compared the proposed IMT system with other state-of-the-art IMT systems. Our IMT system outperformed the results of those state-of-the-start IMT systems that are based on word graphs.

We empirically demonstrated that the two IMT systems proposed in this thesis were able to reduce the user effort that is required to generate correct translations with respect to using a conventional SMT system followed by human post-editing. For this purpose, we compared the KSR results obtained by our proposed systems with the CER and PKSR results calculated from the fully-automatic translations.

Finally, we also performed experiments to test the IMT system with online learning techniques proposed in Chapter 7. The results of the experiments show that our techniques allow the IMT system to learn from scratch or from previously estimated models. In addition to this, the online learning techniques proposed in this thesis allowed us to significantly outperform the results obtained by other state-of-the-art IMT systems described in the literature.

Part IV

Conclusions and Bibliography

CHAPTER 9
CONCLUSIONS

In this chapter we summarise the achievements of this thesis and provide a list of publications related with these achievements. Additionally, the chapter is concluded with a list of directions for future work.

9.1 Summary

In Chapter 2, we defined the list of scientific ([SC]) and technologic ([TC]) goals of this thesis. These goals were classified into fully automatic and interactive phrase-based SMT goals. In this section we summarise the achievements of this thesis with respect to the list of scientific and technologic goals:

1. **Fully-automatic phrase-based SMT achievements**

    - **Improved phrase-based model estimation** [SC]

      We proposed an alternative estimation technique for phrase-based models which we have called BRF (bisegmentation-based relative frequency) estimation. BRF estimation tries to reduce the strong heuristic component of the standard estimation method by considering the extracted phrase pairs as part of complete bisegmentations of the source and target sentences. We theoretically studied the computational complexity of the estimation algorithm, finding that the problem is tractable under some conditions that are commonly met by existing corpora. This was empirically demonstrated by means of a series of experiments. It is also important to stress that the proposed technique did not require additional constraints to be executed.

      Additionally, we also carried out translation quality experiments. The standard estimation technique slightly outperformed our proposed technique, but the differences were not statistically significant. In spite of this, BRF estimation obtained higher likelihood values than the standard estimation technique for corpora of different complexity. One possible explanation for the negative results may be that our estimation technique overfits the training data. Anyway, we think that the acceptable time cost of BRF estimation makes it interesting as a starting point to implement more sophisticated estimation techniques for phrase-based models.

- **Phrase-based model estimation from very large corpora** [TC]

  We proposed a specific training procedure that allows us to train phrase-based models from corpora of an arbitrary size without introducing a significant time overhead. The proposed training procedure works by transforming memory requirements into hard disk requirements. One advantage of the proposed estimation technique is its ability to collect the information that is required to generate direct and inverse probabilities in only one iteration over the set of training samples. The direct and inverse probabilities can be efficiently obtained from the collected information using appropriate data structures. Experimental results obtained on the Europarl corpus shows that the proposed techniques efficiently estimate the parameters of phrase-based models, in some cases even outperforming the efficiency of the standard estimation algorithm. In addition to this, the proposed technique can be easily parallelised.

- **Development of open-source software for phrase-based SMT** [TC]

  We developed the open-source THOT toolkit for SMT (see Appendix B). The THOT toolkit allows to estimate phrase-based models using two different estimation techniques, namely, the well-known, standard phrase-based model estimation technique and the BRF estimation technique proposed in this thesis. The THOT toolkit has been successfully used throughout this thesis to execute SMT and IMT experiments. The development of the THOT toolkit also aimed at offering a publicly available resource for the research community. The toolkit is hosted by SourceForge[a] and released under GPL license. According to information provided by the SourceForge web site, THOT has been downloaded more than one thousand times since its first release. In addition to this, the toolkit has been cited in different research papers.

- **Specific phrase-based model derivation** [SC]

  We proposed a specific derivation for phrase-based models that allows us to obtain a set of statistical submodels governing different aspects of the translation process. In addition to this, these submodels can be introduced as individual components of a log-linear model. Such log-linear model was successfully used throughout this thesis to define and implement different tools, including a phrase-based SMT decoder, a tool to generate alignments at phrase level and a set of IMT systems using different technologies.

- **Branch-and-bound search for phrase-based SMT** [SC]

  We described a search algorithm for SMT which is based on the well-known branch-and-bound search paradigm. The computational complexity of the proposed algorithm can be bounded by the complexity of a well-known dynamic programming algorithm described in the literature. The proposed algorithm incorporates different pruning techniques. Among such pruning techniques, the most important one is the maximum stack size limitation. The pruning efficiency

[a] http://sourceforge.net/

is improved by using multiple stacks. Specifically, those hypotheses with the same number of aligned source words are stored into the same stack.

The initial search algorithm can be modified to obtain new algorithms with different properties, including the breadth-first search algorithm and the generalised multiple-stack search algorithms. Such generalised algorithms can explore the search space either in a best-first or a breadth-first manner.

The search space can be explored in a breadth-first fashion by modifying the scoring function of the initial search algorithm. We theoretically demonstrated that the computational complexity of best-first search cannot be bounded by the complexity of breadth-first search when a maximum stack size limitation is imposed. Nevertheless, empirical results show that best-first search is less time consuming than breadth-first search when translating simple corpora. In addition to this, best-first search executes a less aggressive pruning of the search space. As a result, best-first search allowed us to slightly improve the average score per sentence for a given test corpus with respect to breadth-first search.

Generalised best-first search algorithms determine the number of stacks that will be used during the search process by means of the so-called granularity parameter. This parameter allows us to make a tradeoff between the advantages of single- and multiple-stack algorithms. Empirical results were not positive, obtaining a worse average score per sentence with respect to the initial algorithm. One possible reason for these results may be that the proposed algorithms stores hypotheses with different number of aligned words in the same stack.

Generalised breadth-first search algorithms improve the pruning efficiency by defining equivalence classes for the partial hypotheses. Such equivalence classes are used to map partial hypotheses to stacks. Empirical results show that these algorithms outperform conventional breadth-first search algorithms in terms of average score and time cost per sentence.

Finally, we compared the translation quality obtained by a decoder using generalised breadth-first search with that obtained with the Moses decoder. The obtained results were similar for both decoders when translating the test sets of the Xerox and EU corpora. Regarding the Europarl corpora, Moses obtained similar results to those obtained by our proposed decoder when translating from French and German to the English language, and significantly outperformed our decoder for the Spanish-English language pair. Nevertheless, if the lexical log-linear components of the Moses decoder are removed (our proposed decoder does not include them), then the observed differences between the two decoders were not statistically significant.

- **Efficient decoding with large phrase-based models** [TC]

  We proposed a technique to efficiently handle phrase-based models composed of millions of parameters. The proposed technique is strongly inspired by a classic concept of computer architecture: cache memory. The proposed technique allows to transform main memory requirements into disk requirements without introducing significant time overhead. In addition to this, we also proposed a specific data structure with very low memory requirements to represent the phrase pairs that compose the phrase models. Experiments carried out on the Europarl corpus show that the proposed cache memory architecture has a extremely low rate of cache misses, allowing a very efficient access to phrase-based model parameters. Regarding the proposed data structure, we empirically demonstrated that it greatly reduces the memory requirements with respect to the requirements of standard representation techniques.

- **Generation of phrase-based alignments** [SC]

  We studied the problem of generating alignments at phrase level. The main difficulty that may be encountered during the generation of phrase-level alignments are those situations in which the phrase pairs that are required to compose the phrase alignments are not contained in the phrase table. The proposed solution is based on the use of a phrase-based statistical alignment model together with a set of smoothing techniques. The smoothing techniques consists of different statistical phrase-based model estimators and a lexical distribution which can be combined by means of backoff techniques, linear interpolation or log-linear interpolation. In addition to this, a specific search algorithm able to efficiently explore the space of possible phrase alignments was proposed (specifically, only the hypothesis expansion algorithm has to be modified).

  Although we were interested in evaluating the quality of the phrase-to-phrase alignments, there is not a gold standard for them. As a result of this, we needed to refine the obtained phrase alignments to word alignments in order to compare them with other existing word alignment techniques. Experimental results for a well-known shared task on word alignment evaluation were obtained. The results show the great impact of the smoothing techniques on alignment quality. As we expected, backing-off and linear interpolation worked better than log-linear interpolation.

2. **Interactive phrase-based SMT achievements**

- **Alternative IMT techniques** [SC]

  A common problem in IMT arises when the user sets a prefix which cannot be explained by the statistical models used by the IMT system. By this reason, existing systems use specific techniques to robustly generate the suffixes required in IMT. In this thesis we presented two novel IMT techniques which tackle this problem in different ways. The first one constitutes an application of the techniques to generate alignments at phrase level that were also presented in this thesis. In this technique, robustness is ensured via the application of smoothing techniques over the phrase-based models as well as by means of a specific search algorithm.

  The second IMT technique proposed in this thesis is based on the application of error-correction techniques over the target sentences generated by the SMT system, allowing us to obtain the translation that better explains the user prefix. In contrast with other similar IMT systems described in the literature, we modify the statistical formalisation of the IMT process to justify the use of error-correction techniques. As it is explained in Chapter 6, this new IMT formalisation can be generalised for its use in other pattern recognition applications.

Empirical results obtained on the Xerox and the EU corpora demonstrate that the two IMT systems proposed in this thesis are competitive with state-of-the-art IMT systems. The IMT system based on the application of error-correction techniques was faster than the IMT system based on phrase-level alignments, but obtained worse results.

Additionally, we also empirically demonstrated that the two proposed IMT systems reduced the user effort that is required to generate correct translations with respect to using a conventional SMT system followed by human post-editing.

Finally, the IMT techniques proposed in this thesis have been implemented into an IMT prototype. Such prototype is described in Appendix C.

- **Online learning for IMT** [SC]

  We presented an IMT system that is able to learn from user feedback by means of online learning techniques. This contrasts with existing IMT systems, which are based on the well-known batch learning paradigm. The proposed system is able to incrementally extend the statistical models involved in the translation process, breaking technical limitations encountered in other works. Empirical results obtained on the Xerox and the EU corpora show that our techniques allow the IMT system to learn from scratch or from previously estimated models. One key aspect of the proposed system is the use of HMM-based alignment models trained by means of the incremental EM algorithm.

9.2 Scientific Publications

In this section we summarise the list of publications derived from the work presented in this thesis.

The BRF phrase-based model estimation procedure along with the THOT toolkit presented in Chapter 3, were described in an international conference:

- D. Ortiz, I. García-Varea, and F. Casacuberta. Thot: a toolkit to train phrase-based statistical translation models. In *Proceedings of the Machine Translation Summit X*, pages 141–148. Asia-Pacific Association for Machine Translation, Phuket, Thailand, September 2005. **CORE B**

The THOT toolkit also yielded a publication in an international workshop as an invited talk as well as in a national and an international conference:

- D. Ortiz, I. García-Varea, F. Casacuberta, L. Rodríguez, and J. Tomás. Thot. New features to deal with larger corpora and long sentences. In *TC-STAR OpenLab on Speech Translation Workshop*, Trento, Italy, 30th March - 1st April 2006. http://tc-star.itc.it/openlab2006/day1/GarciaVarea.pdf. **invited talk**.

- D. Ortiz, I. García-Varea, and F. Casacuberta. Estimación de modelos de traducción de secuencias de palabras a partir de corpus muy grandes mediante thot. In *Actas de las IV Jornadas en Tecnologías del Habla*, pages 65-70, Zaragoza, Spain, November 2006.

- R. San-Segundo, A. Pérez, D. Ortiz, L. F. D'Haro, I. Torres, and F. Casacuberta. Evaluation of alternatives on speech to sign language translation. In *Proceedings of the Interspeech conference*, pages 2529–2532, Antwerp, Belgium, August 2007. **CORE A**

The techniques to estimate phrase-based models from very large corpora presented in Chapter 3; as well as the techniques to access the resulting models during the decoding stage presented in Chapter 4, were described in a national and an international conference:

- D. Ortiz, I. García-Varea, and F. Casacuberta. Algunas soluciones al problema del escalado en traducción automática estadística. In *Actas del Campus Multidisciplinar en Percepción e Inteligencia*, pages 830–842, Albacete, Spain, July 2006.

- D. Ortiz, I. García-Varea, and F. Casacuberta. A general framework to deal with the scaling problem in phrase-based statistical machine translation. In *Proceedings of the 3rd Iberian Conference on Pattern Recognition and Image Analysis*, volume 4478 of *Lecture Notes in Computer Science*, pages 314–322. Springer Verlag, Girona (Spain), June 2007. **CORE C**

These techniques also yielded a publication in an international journal:

- D. Ortiz-Martínez, I. García-Varea, and F. Casacuberta. The scaling problem in the pattern recognition approach to machine translation. *Pattern Recognition Letters*, 29:1145–1153, 2008. **JCR**

A predecessor of the work on decoding with branch-and-bound search algorithms presented in Chapter 4 was published in an international conference. In this work, single-word alignment models instead of phrase-based models were used:

- D. Ortiz, I. García-Varea, and F. Casacuberta. An empirical comparison of stack-based decoding algorithms for statistical machine translation. In *Proceedings of the 1st Iberian Conference on Pattern Recognition and Image Analysis*, volume 2652 of *Lecture Notes in Computer Science*, pages 654–663. Springer Verlag, Mallorca, Spain, June 2003. **CORE C**

Some of the features of the generalised branch-and-bound search algorithm for phrase-based models were presented in an international workshop:

- D. Ortiz, I. García-Varea, and F. Casacuberta. Generalized stack decoding algorithms for statistical machine translation. In *Proceedings of the HLT-NAACL Workshop on Statistical Machine Translation*, pages 64–71, New York City, USA, June 4–9 2006.

The proposed branch-and-bound search algorithms proposed in this thesis were used to implement an improved IMT system in the following international conference:

- G. Sanchís-Trilles, D. Ortiz-Martínez, J. Civera, F. Casacuberta, E. Vidal, H. Hoang. Improving interactive machine translation via mouse actions. In *Proceedings of the Empirical Methods in Natural Language Processing conference*, pages 485–494, Honolulu, Hawaii, November 2008. **CORE A**

Additionally, The proposed branch-and-bound search algorithms were also used to empirically demonstrate theoretical results related to loss functions in SMT. This work was presented in an international journal:

- J. Andrés Ferrer, D. Ortiz Martínez, I. Garca Varea, F. Casacuberta Nolla. On the use of different loss functions in statistical pattern recognition applied to machine translation. In *Pattern Recognition Letters*, 29(8):1072-1181, 2008. **JCR**

The phrase-level alignment generation technique described in Chapter 4 was published in an international conference:

- D. Ortiz-Martínez, I. García-Varea, and F. Casacuberta. Phrase-level alignment generation using a smoothed loglinear phrase-based statistical alignment model. In *Proceedings of the Conference of the European Association for Machine Translation*, pages 160–169, Hamburg, Germany, September 2008. **best paper award**. **CORE B**

  This contribution received the best paper award sponsored by the Springer Verlag editorial to a paper accepted for publication in the EAMT conference.

A predecessor of the work on phrase-level alignment generation mentioned above was presented in an international conference:

- I. García-Varea, D. Ortiz, F. Nevado, P. A. Gómez, and F. Casacuberta. Automatic segmentation of bilingual corpora: A comparison of different techniques. In *Proceedings of the 2nd Iberian Conference on Pattern Recognition and Image Analysis*, volume 3523 of *Lecture Notes in Computer Science*, pages 614–621. Springer Verlag, Estoril, Portugal, June 2005. **CORE C**

The generation of phrase-level alignments was applied to implement a technique to prune the parameters of phrase-based models. Such technique was presented in an international conference:

- G. Sanchís-Trilles, D. Ortiz-Martínez, J. González-Rubio, J. González, F. Casacuberta. Bilingual segmentation for phrasetable pruning in statistical machine translation. In *Proceedings of the European Association for Machine Translation*, pages 257–264, Leuven, Belgium, May, 2011. **CORE B**

The IMT system based on partial phrase-based alignments described in Chapter 6 was published in an international conference:

- D. Ortiz-Martínez, I. García-Varea, and F. Casacuberta. Interactive machine translation based on partial statistical phrase-based alignments. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 330–336, Borovets, Bulgaria, September 2009. (**paper selected to appear in the volume "Best RANLP papers"**). **CORE C**

  This contribution has been selected to appear in the volume "Best RANLP papers" published by John Benjamins, due to the excellent reviews received at this conference. In addition to this, the authors have been invited to submit an extended version of the paper to the JNLE (Journal of Natural Language Engineering).

The IMT techniques presented in Chapter 6 were used to build different prototypes that were presented in international conferences:

- V. Alabau, D. Ortiz, V. Romero, and J. Ocampo. A multimodal predictive-interactive application for computer assisted transcription and translation. In *Proceedings of the International Conference on Multimodal interfaces*, pages 227–228, New York, NY, USA, 2009. ACM. **CORE B**
- D. Ortiz-Martínez, L. A. Leiva, V. Alabau, and F. Casacuberta. Interactive machine translation using a web-based architecture. In *Procedings of the International Conference on Intelligent User Interfaces*, pages 423–425. Hong Kong, China, February 2010. **CORE A**
- V. Alabau, D. Ortiz-Martínez, A. Sanchís, F. Casacuberta. Multimodal interactive machine translation. In *Proceedings of the International Conference on Multimodal Interfaces*, Beijing, China, November 2010. **CORE B**

Additionally, the IMT system based on error-correction techniques was the basis of two new IMT system proposals which were presented in international conferences:

- J. González-Rubio, D. Ortiz-Martínez, F. Casacuberta. On the use of confidence measures within an interactive-predictive machine translation system. In *Proceedings of the European Association for Machine Translation conference*, Saint Raphael, France, May, 2010. **CORE B**
- J. González-Rubio, D. Ortiz-Martínez, F. Casacuberta. Balancing user effort and translation error in interactive machine translation via confidence measures. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 173–177, Uppsala, Sweden, July, 2010. **CORE A**

The online learning techniques for IMT presented in Chapter 7, as well as a prototype of an IMT system with online learning capabilities were published in two international conferences:

- D. Ortiz-Martínez, I. García-Varea, F. Casacuberta. Online learning for interactive statistical machine translation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies*, pages 546–554, Los Angeles, USA, June 2010. **CORE A**
- D. Ortiz-Martínez, L. A. Leiva, V. Alabau, I. García-Varea, and F. Casacuberta. An interactive machine translation system with online learning. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics - Human Language Technologies*, pages 68–73, companion volume, system demonstrations, Portland, USA, June 2011. **CORE A**

Finally, part of the work on IMT presented in this thesis was also published in three book chapters:

- J. Civera, J. González-Rubio, D. Ortiz-Martínez. Interactive machine translation. In *Multimodal Interactive Pattern Recognition and Applications*, pages 135-152, Springer, June, 2011.

- D. Ortiz-Martínez, I. García-Varea. Incremental and adaptive learning for interactive machine translation. In *Multimodal Interactive Pattern Recognition and Applications*, pages 169-178, Springer, June, 2011.

- L. A. Leiva, V. Alabau, V. Romero, F. M. Segarra, R. Sánchez Sáez, D. Ortiz-Martínez, L. Rodríguez. Prototypes and demonstrators. In *Multimodal Interactive Pattern Recognition and Applications*, pages 227-266, Springer, June, 2011.

9.3 Future Work

In this section we outline future directions for further developments of the work presented in this thesis.

- **Richer phrase-based models using BRF estimation**

  In this thesis, an alternative estimation technique for phrase-based models called BRF estimation has been proposed. As was explained in Chapter 3, the proposed technique can be straightforwardly modified to obtain more detailed information about the bisegmentation process, including information about bisegmentation lengths, about source and target phrase lengths or about reorderings. This contrasts with the vast majority of phrase-based models described in the literature, where only the phrase-to-phrase probabilities and (in some cases) the reordering probabilities are estimated from training data.

- **Improved phrase-based model estimation using EM algorithm**

  As was explained in Chapter 3, the BRF estimation algorithm can be modified to efficiently compute the sum of the probability for each possible bisegmentation composed of consistent phrase pairs. This extension can be used to partially compute the E step of the EM algorithm. This partial computation can be justified by means of the sparse version of the EM algorithm proposed in [NH98]. The resulting estimation procedure would be similar to that proposed in [DGZK06]. However, in that work the maximum phrase length is limited to three words and only lexical and distortion parameters are estimated. We think that our proposed computation of the E step will allow us to remove the maximum phrase length limitation. In addition to this, our proposed specific phrase-based model derivation can be applied to determine a complete set of distributions to be estimated.

  Additionally, the information generated by our proposed BRF estimation algorithm can be used to generate random samples from the set of bisegmentations that are composed of consistent phrase pairs. For the future we plan to study if this sampling technique can be useful to estimate phrase-based models by means of the Monte-Carlo EM algorithm [WT90].

- **Further development of open-source software**

  The THOT toolkit constitutes a publicly available resource for the SMT scientific community and has been successfully used to carry out experiments presented in this thesis. In addition to this, THOT has also been the starting point to develop new code and tools for SMT and IMT also used in this thesis, including a phrase-based translation decoder, a tool to generate alignments at phrase level, IMT engines, tools to incrementally estimate statistical translation models, etc. In the near future, we plan to study the interest of releasing public versions of this software.

- **Further applications of statistical phrase-level alignments**

  We studied the problem of generating alignments at phrase level, which can be seen as a slightly modified version of the search problem. One possible application of the generation of phrase-level alignments is in the area of multi-source SMT [ON01]. In that paper, the PROD ranking technique is described. This technique requires the generation of phrase-level alignments between two languages, but their authors report coverage problems that make the technique impractical. These problems can be solved by means of the techniques proposed here. In addition to this, the best phrase alignments for each sentence pair can be used to perform a Viterbi-like estimation of phrase-based models as it is proposed in [WMN10]. Phrase-level alignments have been also used in discriminative training [LBCKT06], training of phrase segmentation models [SDAS08], etc.

- **Development of more complex error-correction models for IMT**

  In this thesis we proposed an IMT system based on stochastic error-correction models. These stochastic error-correction models allow us to find the target translation that better explains the prefix given by the user. We have used probabilistic finite state machines (PFSMs) with ad-hoc parameters as error-correction models. One possible continuation of the presented work is to estimate the parameters of the PFSMs by means of the EM algorithm. In addition to this, PFSMs can be replaced by more complex models, such as the IBM or the HMM-based alignment models. One advantage of these models with respect to models based on PFSMs is that they can represent non-monotonic alignments between the target translation and the user prefix.

- **Implementation of interactive systems using the proposed generalised formalisation**

  One of the IMT techniques presented in this thesis uses an alternative formalisation of the IMT process in which the target sentence generated by the system and the user prefix constitute separated entities. As it was explained, this alternative formalisation can be generalised for its use in other pattern recognition applications, including multi-source translation, computer assisted speech transcription, multimodal computer assisted translation and computer assisted transcription of text images. For the future, we plan to implement the pattern recognition applications mentioned above following our general formalisation.

- **Further applications of incremental learning in SMT and IMT**

  We implemented an IMT system with online learning which is based on the application of incremental learning techniques. It is worth noting that the incremental techniques proposed here can also be exploited to extend SMT systems (in fact, our proposed IMT system is based on an incrementally updateable SMT system). For the near future we plan to study possible applications of our techniques both in the SMT and IMT frameworks. One example of these applications is active learning (the interested reader can find a survey on active learning in [Set09]). In the active learning paradigm, the learner pose queries, usually in the form of unlabelled data instances to be labelled by an oracle. Incremental learning can help in those active learning scenarios in which the time required to process a newly labelled sample is slow due to the necessity of performing a complete retraining of the set of labelled samples.

- **Apply online learning techniques in other interactive applications**

  In this thesis we presented an IMT system with online learning. Online learning fits nicely into the IMT framework, since the user generates new training samples as a by-product of the use of the IMT system. The proposed online learning techniques require the definition of incremental versions of the statistical models involved in the interactive translation process. These online learning techniques can be exported to other interactive applications, such as computer assisted speech transcription, interactive image retrieval, etc (see [VRCGV07] for a complete list).

BIBLIOGRAPHY

[AB92] M. Anthony and N. Biggs. *Computational learning theory: an introduction*. Cambridge University Press, New York, NY, USA, 1992.

[ABC$^+$00] J. C. Amengual, J. M. Benedí, F. Casacuberta, M. A. Castaño, A. Castellanos, V. M. Jiménez, D. Llorens, A. Marzal, M. Pastor, F. Prat, E. Vidal, and J. M. Vilar. The EuTrans-I speech translation system. *Machine Translation*, 1, 2000.

[ABD00] H. Alshawi, S. Bangalore, and S. Douglas. Learning dependency translation models as collections of finite state head transducers. *Computational Linguistics*, 26(1):45–60, 2000.

[AFJC07] J. Andrés-Ferrer and A. Juan-Císcar. A phrase-based hidden markov model approach to machine translation. In *Proceedings of New Approaches to Machine Translation*, pages 57–62, January 2007.

[AFOMGVC08] J. Andrés-Ferrer, D. Ortiz-Martínez, I. García-Varea, and F. Casacuberta. On the use of different loss functions in statistical pattern recognition applied to machine translation. *Pattern Recognition Letters*, 29(8):1072–1181, 2008.

[AK07] A. Arun and P. Koehn. Online learning methods for discriminative training of phrase based statistical machine translation. In *Proceedings of the Machine Translation Summit XI*, pages 15–20, Copenhagen, Denmark, September 2007.

[Als96a] H. Alshawi. Head automata and bilingual tiling: translation with minimal representations. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, ACL '96, pages 167–176, Stroudsburg, PA, USA, 1996. Association for Computational Linguistics.

[Als96b] H. Alshawi. Head automata for speech translation. In *Proceedings of the International Conference on Spoken Language Processing*, volume 4, pages 2360–2363, Philadelphia, PA, 1996.

[AV98] J.C. Amengual and E. Vidal. Efficient error-correcting viterbi parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.PAMI-20, No.10:1109–1116, October 1998.

[AX97] H. Alshawi and F. Xiang. English-to-Mandarin speech translation with head transducers. In *Spoken Language Translation Workshop (SLT-97)*, pages 54–60, Madrid (SPAIN), July 1997.

[BBC+09] S. Barrachina, O. Bender, F. Casacuberta, J. Civera, E. Cubel, S. Khadivi, A. L. Lagarda, H. Ney, J. Tomás, E. Vidal, and J. M. Vilar. Statistical approaches to computer-assisted translation. *Computational Linguistics*, 35(1):3–28, 2009.

[BBD+94] A. L. Berger, P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, J. R. Gillett, J. D. Lafferty, H. Printz, and L. Ureš. The Candide system for machine translation. In *Proceedings of the ARPA Workshop on Human Language Technology*, pages 157–162, Plainsboro, NJ, March 1994.

[BBD+96] A. L. Berger, P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, J. R. Gillett, A. S. Kehler, and R. L. Mercer. Language translation apparatus and method of using context-based translation models. United States Patent, No. 5510981, April 1996.

[BCBOK06] A. Birch, C. Callison-Burch, M. Osborne, and P. Koehn. Constraining the phrase-based, joint probability statistical translation model. In *Proceedings of the ACL Workshop on Statistical Machine Translation*, pages 154–157, New York City, June 2006.

[BDDM93] P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, 1993.

[Bel57] R. Bellman. *Dynamic programming*. Princeton University Press, Princeton, NJ, 1957.

[BHV+05] O. Bender, S. Hasan, D. Vilar, R. Zens, and H. Ney. Comparison of generation strategies for interactive machine translation. In *Conference of the European Association for Machine Translation*, pages 33–40, Budapest, Hungary, May 2005.

[BJ75] L. R. Bahl and F. Jelinek. Decoding for channels with insertions, deletions, and substitutions wi th applications to speech recognition. *IEEE Transactions on Information Theory,* IT-21(4):404–411, 1975.

[BL05] S. Banerjee and A. Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization,* pages 65–72, Ann Arbor, Michigan, USA, June 2005.

[BRW00] S. Bangalore, O. Rambow, and S. Whittaker. Evaluation metrics for generation. In *Proceedings of the First International Natural Language Generation Conference, Mitzpe,* pages 1–8, 2000.

[CBBS05] C. Callison-Burch, C. Bannard, and J. Schroeder. Scaling phrase-based statistical machine translation to larger corpora and longer sentences. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics,* pages 255–262, Ann Arbor, June 2005.

[CBFK$^+$07] C. Callison-Burch, C. Fordyce, P. Koehn, C. Monz, and J. Schroeder. (meta-) evaluation of machine translation. In *Proceedings of the ACL Workshop on Statistical Machine Translation*, pages 136–158, Prague, Czech Republic, June 2007.

[CBRS08] N. Cesa-Bianchi, G. Reverberi, and S. Szedmak. Online learning algorithms for computer-assisted translation. Deliverable D4.2, SMART: Stat. Multilingual Analysis for Retrieval and Translation, Mar. 2008.

[CG91] K. W. Church and W. A. Gale. A comparison of the enhanced good-turing and deleted estimation methods for estimating probalities of english bigrams. *Computer Speech and Language*, 5:19–54, 1991.

[CG96] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. In Arivind Joshi and Martha Palmer, editors, *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, pages 310–318, San Francisco, 1996. Morgan Kaufmann Publishers.

[CGV94] A. Castellanos, I. Galiano, and Enrique Vidal. Application of OSTIA to machine translation tasks. In R. C. Carrasco and J. Oncina, editors, *Grammatical Inference and Applications, Proc. of 2nd ICGI*, volume 862 of *Lecture Notes in Computer Science*, pages 93–105. Springer-Verlag, Alicante, Spain, 1994.

[Chi05] D. Chiang. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 263–270, Morristown, NJ, USA, 2005.

[Chi07] D. Chiang. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228, 2007.

[Civ08] J. Civera. *Novel statistical approaches to text classification, machine translation and computer-assisted translation*. PhD thesis, Universidad Politécnica de Valencia, Valencia (Spain), Juny 2008. Advisors: A. Juan and F. Casacuberta.

[CM09] O. Cappé and E. Moulines. On-line expectation-maximization algorithm for latent data models. *Journal of the Royal Statistical Society Ser. B*, 71(1):593–613, 2009.

[CMR08] D. Chiang, Y. Marton, and P. Resnik. Online large-margin training of syntactic and structural translation features. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2008.

[CNO$^+$04] F. Casacuberta, H. Ney, F. J. Och, E. Vidal, J. M. Vilar, S. Barrachina, I. García-Varea, D. Llorens, C. Martínez, S. Molau, F. Nevado, M. Pastor, D. Picó, A. Sanchis, and C. Tillmann. Some approaches to statistical and finite-state speech-to-speech translation. *Computer Speech and Language*, 18:25–47, January 2004.

[Com08] European Communities. Directorate-general for translation. http://ec.europa.eu/dgs/translation/index_en.htm, 2008.

[CV04] F. Casacuberta and E. Vidal. Machine translation with inferred stochastic finite-state transducers. *Computational Linguistics*, 30(2):205–225, 2004.

[CVC$^+$04] J. Civera, J. M. Vilar, E. Cubel, A. L. Lagarda, S. Barrachina, E. Vidal, F. Casacuberta, D. Picó, and J. González. From machine translation to computer assisted translation using finite-state models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Barcelona, 2004.

[DB05] Y. Deng and W. Byrne. HMM word and phrase alignment for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 169–176, October 2005.

[DBCK08] J. DeNero, A. Bouchard-Côté, and D. Klein. Sampling alignment structure under a bayesian translation model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 314–323, Morristown, NJ, USA, 2008.

[DCML08] C. Dyer, A. Cordova, A. Mont, and J. Lin. Fast, easy, and cheap: Construction of statistical machine translation models with MapReduce. In *Proceedings of the ACL Workshop on Statistical Machine Translation*, pages 199–207, Columbus, Ohio, 2008.

[DG04] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. In *Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation*, pages 137–150, Berkeley, CA, USA, 2004. USENIX Association.

[DGZK06] J. DeNero, D. Gillick, J. Zhang, and D. Klein. Why generative phrase models underperform surface heuristics. In *Proceedings of the HLT-NAACL Workshop on Statistical Machine Translation*, pages 31–38, Morristown, NJ, USA, 2006.

[DK08] J. DeNero and D. Klein. The complexity of phrase alignment problems. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies*, pages 25–28, Morristown, NJ, USA, 2008.

[DLR77] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society Ser. B*, 39(1):1–22, 1977.

[Dod02] G. Doddington. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings ARPA Workshop on Human Language Technology*, 2002.

[DR72] J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43:1470–1480, 1972.

[Epp99] D. Eppstein. Finding the k shortest paths. *SIAM J. Comput.*, 28(2):652–673, 1999.

[FC07] Marcello Federico and Mauro Cettolo. Efficient handling of n-gram language models for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 88–95, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics.

[FIP97] G. Foster, P. Isabelle, and P. Plamondon. Target-text mediated interactive machine translation. *Machine Translation*, 12(1):175–194, 1997.

[FKJ06] G. Foster, R. Kuhn, and H. Johnson. Phrasetable smoothing for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 53–61, Sydney, Australia, July 2006.

[FLL02] G. Foster, P. Langlais, and G. Lapalme. User-friendly text prediction for translators. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 148–155, 2002.

[Fos02] G. Foster. *Text Prediction for Translators*. PhD thesis, Université de Montréal, 2002.

[GC00] C. Giraud-Carrier. A note on the utility of incremental learning. *AI Communications*, 13(4):215–223, December 2000.

[Ger01] U. Germann. Aligned hansards of the 36th parliament of canada, 2001. `http://www.isi.edu/natural-language/download/hansard/`.

[GGK$^+$06] M. Galley, J. Graehl, K. Knight, D. Marcu, S. DeNeefe, W. Wang, and I. Thayer. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 961–968, Morristown, NJ, USA, 2006.

[GHKM04] M. Galley, M. Hopkins, K. Knight, and D. Marcu. What's in a translation rule? In *Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference*, Boston, USA, May 2004.

[GJK$^+$01] U. Germann, M. Jahr, K. Knight, D. Marcu, and K. Yamada. Fast decoding and optimal decoding for machine translation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 228–235, Toulouse, France, July 2001.

[GSC08] J. González, G. Sanchis, and F. Casacuberta. Learning finite state transducers using bilingual phrases. In *9th International Conference on Intelligent Text Processing and Computational Linguistics. Lecture Notes in Computer Science*, Haifa, Israel, February 2008.

[GV03] I. García-Varea. *Traducción automática estadística: Modelos de traducción basados en máxima entropía y algoritmos de búsqueda*. PhD thesis, Departamento de Sistemas Informáticos y Computación. Universidad Politécnica de Valencia, Valencia, España, Diciembre 2003.

[GV08] Q. Gao and S. Vogel. Parallel implementations of word alignment tool. In *Proceedings of the ACL Workshop on Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, pages 49–57, Columbus, Ohio, June 2008.

[GVON$^+$05] I. García-Varea, D. Ortiz, F. Nevado, P. A. Gómez, and F. Casacuberta. Automatic segmentation of bilingual corpora: A comparison of different techniques. In *Proceedings of the Second Iberian Conference on Pattern Recognition and Image Analysis*, volume 3523 of *Lecture Notes in Computer Science*, pages 614–621. Springer Verlag, Estoril (Portugal), June 2005.

[HNR68] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems, Science, and Cybernetics*, SSC-4(2):100–107, 1968.

[Hob92] J. R. Hobbs. Machine translation. Technical report, DARPA. Software and Intelligent Systems Technology Office, February 1992.

[HP03] J. L. Hennessy and D. A. Patterson. *Computer architecture: a quantitative approach*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[HS92] W. John Hutchins and Harold L. Somers. *An introduction to machine translation*. Academic Press, Cambridge, MA, 1992.

[HZN07] S. Hasan, R. Zens, and H. Ney. Are very large n-best lists useful for smt? In *Proceedings of the North American Chapter of the Association for Computational Linguistics conference; Companion Volume, Short Papers on XX*, pages 57–60, Morristown, NJ, USA, 2007.

[IC97] P. Isabelle and K. Church. Special issue on new tools for human translators. *Machine Translation*, 12(1–2), 1997.

[Ima02] K. Imamura. Application of translation knowledge acquired by hierarchical phrase alignment for pattern-based MT. In *Proceedings of the 9th Conference on Theoretical and Methodological Issues in Machine Translation*, pages 74–84, 2002.

[Jel98] F. Jelinek. *Statistical methods for speech recognition*. MIT Press, Cambridge, MA, 1998.

[JM03] V. M. Jiménez and A. Marzal. A lazy version of eppstein's k shortest paths algorithm. In *Proceedings of the 2nd international conference on Experimental and efficient algorithms*, pages 179–191, Berlin, Heidelberg, 2003. Springer-Verlag.

[JMB75] F. Jelinek, R. L. Mercer, and L. R. Bahl. Design of a linguistic statistical decoder for the recognition of continuous speech. *IEEE Transactions on Information Theory*, IT-21(3):250–256, May 1975.

[KAO98] K. Knight and Y. Al-Onaizan. Translation with finite-state devices. In *Proceedings of the 4th Conference of the Association for Machine Translation in the Americas*, pages 421–437, Langorne, PA, USA, October 1998.

[KG03] S. Khadivi and C. Goutte. Tools for corpus alignment and evaluation of the alignments (deliverable d4.9). Technical report, TransType 2 (IST-2001-32091), 2003.

[KG05] K. Knight and J. Graehl. An overview of probabilistic tree transducers for natural language processing. In *Proceedings of the 5th International Conference on Intelligent Text Processing and Computational Linguistics*, pages 1–24, 2005.

[KH07] P. Koehn and H. Hoang. Factored translation models. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 868–876, Prague, Czech Republic, June 2007.

[KHB$^+$07] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 177–180, Prague, Czech Republic, June 2007.

[KM06] P. Koehn and C. Monz. Manual and automatic evaluation of machine translation between european languages. In *Proceedings of the HLT-NAACL Workshop on Statistical Machine Translation*, pages 102–121, New York City, June 2006.

[KN95] R. Kneser and H. Ney. Improved backing-off for m-gram language modeling. In *IEEE International Conference on Acoustics, Speech and Signal Processing, volume 1*, pages 181–184, Trier, Germany, 1995.

[Kni99] K. Knight. Decoding complexity in word-replacement translation models. *Computational Linguistics*, 25(4):607–615, 1999.

[Knu73] D. E. Knuth. *Sorting and searching*, volume 3 of *The Art of Computer Programming*. Addison-Wesley, Reading, Massachusetts, second edition, 10 January 1973.

[Knu81] D. E. Knuth. *Seminumerical algorithms*, volume 2 of *The Art of Computer Programming*. Addison-Wesley, Massachusetts, 2nd edition, 1981.

[Koe03] P. Koehn. *Noun phrase translation*. PhD thesis, Los Angeles, CA, USA, 2003.

[Koe05] P. Koehn. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the Machine Translation Summit X*, pages 79–86, September 2005.

[KOM03] P. Koehn, F. J. Och, and D. Marcu. Statistical phrase-based translation. In *Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference*, pages 48–54, Edmonton, Canada, May 2003.

[LBCKT06] P. Liang, A. Bouchard-Côté, D. Klein, and B. Taskar. An end-to-end discriminative approach to machine translation. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics*, pages 761–768, Morristown, NJ, USA, 2006.

[LCBO10] A. Levenberg, C. Callison-Burch, and M. Osborne. Stream-based translation models for statistical machine translation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies*, pages 394–402, Los Angeles, California, June 2010.

[LD60] A. H. Land and A. G Doig. An automatic method of solving discrete programming problems. *Econometrica*, 28(3):497–520, 1960.

[Lev66] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady.*, 10(8):707–710, February 1966.

[LFL00] P. Langlais, G. Foster, and G. Lapalme. Unit completion for a computer-aided translation typing system. *Machine Translation*, 15(4):267–294, 2000.

[Lit88] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. In *Machine Learning*, pages 285–318, 1988.

[LJS$^+$95] D. Llorens, V. Jiménez, J. A. Sánchez, E. Vidal, and H. Rulot. ATROS, an automatically trainable continuous-speech recognition system for limited-domain tasks. In *Proceedings of the VI National Symposium on Pattern Recognition and Image Analysis*, pages 478–483, Barcelona, Spain, May 1995.

[LLL02] P. Langlais, G. Lapalme, and M. Loranger. Transtype: Development-evaluation cycles to boost translator's productivity. *Machine Translation*, 15(4):77–98, 2002.

[Lop08] A. Lopez. Statistical machine translation. *ACM Comput. Surv.*, 40(3), 2008.

[MDM91] E. Mays, F. J. Damerau, and R. L. Mercer. Context based spelling correction. *Information Processing and Management: an International Journal.*, 27(5):517–522, 1991.

[Mel04] I. D. Melamed. Statistical machine translation by parsing. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 653, Morristown, NJ, USA, 2004.

[MP03] R. Mihalcea and T. Pedersen. An evaluation exercise for word alignment. In *Proceedings of the HLT-NAACL Workshop on Building and Using Parallel Texts: Data Driven Machine Translation and Beyond*, pages 1–10, Edmonton, Alberta, Canada, May 2003.

[MS01] C. D. Manning and H. Schütze. *Foundations of statistical natural language Processing*. MIT Press, Cambridge, Massachusetts 02142, 2001.

[MW02] D. Marcu and W. Wong. A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1408–1414, Philadelphia, USA, July 2002.

[NEK94] H. Ney, U. Essen, and R. Kneser. On structuring probabilistic dependencies in stochastic language modelling. *Computer Speech and Language*, 8:1–38, 1994.

[Ney95] H. Ney. On the probabilistic-interpretation of neural-network classifiers and discriminative training criteria. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):107–119, February 1995.

[Ney01] H. Ney. Stochastic modelling: From pattern classification to language translation. In *Proceedings of the Data-Driven Machine Translation Workshop, 39th Annual Meeting of the Association for Computational Linguistics,* pages 33–37, Toulouse, France, July 2001.

[NH98] R.M. Neal and G.E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Proceedings of the NATO-ASI on Learning in graphical models*, pages 355–368, Norwell, MA, USA, 1998.

[NIS06] Nist 2006 machine translation evaluation official results. http://www.nist.gov/speech/tests/mt/mt06eval_official_results.html, November 2006.

[NLLF04] L. Nepveu, G. Lapalme, P. Langlais, and G. Foster. Adaptive language and translation models for interactive machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 190–197, Barcelona, Spain, July 2004.

[NM65] J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, January 1965.

[NNO$^+$00] H. Ney, S. Nießen, F. J. Och, H. Sawaf, C. Tillmann, and S. Vogel. Algorithms for statistical translation of spoken language. *IEEE Transactions on Speech and Audio Processing*, 8(1):24–36, January 2000.

[NOL00] S. Nießen, F. J. Och, and H. Leusch, G.and Ney. An evaluation tool for machine translation: Fast evaluation for MT research. In *International Conference on Language Resources and Evaluation*, pages 39–45, Athens, Greece, May 2000.

[Och00] F. J. Och. GIZA++: Training of statistical translation models, 2000. http://www-i6.informatik.rwth-aachen.de/~och/software/GIZA++.html.

[Och02] F. J. Och. *Statistical machine translation: From single-word models to alignment templates*. PhD thesis, Computer Science Department, RWTH Aachen, Germany, October 2002.

[Och03] F. J. Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 160–167, Morristown, NJ, USA, 2003.

[OGV93] J. Oncina, P. García, and E. Vidal. Learning subsequential transducers for pattern recognition interpretation tasks. *IEEE Transactions on PAMI*, 15(5):448–458, 1993.

[OGVC03] D. Ortiz, I. García-Varea, and F. Casacuberta. An empirical comparison of stack-based decoding algorithms for statistical machine translation. In *New Advance in Computer Vision*, Lecture Notes in Computer Science. Springer-Verlag, 2003. 1st Iberian Conference on Pattern Recognition and Image Analysis Mallorca. Spain. June.

[OGVC05] D. Ortiz, I. Garca-Varea, and F. Casacuberta. Thot: a toolkit to train phrase-based statistical translation models. In *Proceedings of the Machine Translation Summit X*, pages 141–148. Asia-Pacific Association for Machine Translation, Phuket, Thailand, September 2005.

[OMGVC08] D. Ortiz-Martnez, I. García-Varea, and F. Casacuberta. The scaling problem in the pattern recognition approach to machine translation. *Pattern Recognition Letters*, 29(8):1145–1153, 2008.

[ON00] F. J. Och and H. Ney. A comparison of alignment models for statistical machine translation. In *Proceedings of the 18th conference on Computational linguistics*, pages 1086–1090, Morristown, NJ, USA, 2000.

[ON01] F. J. Och and H. Ney. Statistical multi-source translation. In *Proceedings of the Machine Translation Summit VIII*, pages 253–258, Santiago de Compostela, Spain, September 2001.

[ON02] F. J. Och and H. Ney. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, PA, July 2002.

[ON03] F. J. Och and H. Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, March 2003.

[OTN99] F. J. Och, C. Tillmann, and H. Ney. Improved alignment models for statistical machine translation. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 20–28, University of Maryland, College Park, MD, June 1999.

[OUN01] F. J. Och, N. Ueffing, and H. Ney. An efficient A* search algorithm for statistical machine translation. In *Data-Driven Machine Translation Workshop*, pages 55–62, Toulouse, France, July 2001.

[OZN03] F. J. Och, R. Zens, and H. Ney. Efficient search for interactive statistical machine translation. In *Tenth Conference of the European Chapter of the Association for Computational Linguistics*, pages 387–393, Budapest, Hungary, April 2003.

[Pow64] M. J. D. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, 7(2):155–162, February 1964.

[PRW98] K. A. Papineni, S. Roukos, and R. T. Ward. Maximum likelihood and discriminative training of direct translation models. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 189–192, Seattle, WA, May 1998.

[PRWZ01] K. A. Papineni, S. Roukos, T. Ward, and W. Zhu. Bleu: a method for automatic evaluation of machine translation. Technical Report RC22176 (W0109-022), IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY, September 2001.

[RCV07] L. Rodríguez, F. Casacuberta, and E. Vidal. Computer assisted transcription of speech. In *Proceedings of the 3rd Iberian Conference on Pattern Recognition and Image Analysis, Volume 4477 of LNCS*, pages 241–248, Girona (Spain), June 2007.

[Ros58] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

[Ros00] R. Rosenfeld. Two decades of statistical language modeling: where do we go from here? 2000. School of Computer Science, Carnegie Melon University, Pittsburgh, PA 15213.

[RTV10] V. Romero, A. H. Toselli, and E. Vidal. Character-level interaction in computer-assisted transcription of text images. In *International Conference on Frontiers in Handwritten Recognition*, pages 539–544. Kolkata, India, November 2010.

[RY97] E. S. Ristad and P. N. Yianilos. Learning string-edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5):522–532, 1997.

[SDAS08] W. Shen, B. Delaney, T. Anderson, and R. Slyh. The MIT-LL/AFRL IWSLT-2008 MT System. In *Proceedings of the International Workshop on Spoken Language Translation*, pages 69–76, Hawaii, USA, 2008.

[SdIfIV$^+$01] SchlumbergerSema S.A., Instituto Tecnológico de Informática, Rheinisch Westfälische Technische Hochschule Aachen Lehrstul für Informatik VI, Recherche Appliquée en Linguistique Informatique Laboratory University of Montreal, Celer Soluciones, Société Gamma, and Xerox Research Centre Europe. TT2. TransType2 - computer assisted translation. Project technical annex., 2001.

[SDS$^+$06] M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas*, pages 223–231, Boston, Massachusetts, USA, August 2006.

[Set09] B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[Som98] H. L. Somers. New paradigms in MT: the state of play now that the dust has settled. In *Proceedings of the ESSLI'98, Workshop on Machine Translation*, pages 22–33, Saarbrcken, Germany, August 1998.

[Sto02] A. Stolcke. Srilm-an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, pages 257–286, Menlo Park, CA, USA, November 2002.

[TC01] J. Tomás and F. Casacuberta. Monotone statistical translation using word groups. In *Proceedings of the Machine Translation Summit VIII*, pages 357–361, Santiago de Compostela, Spain, 2001.

[TC06] J. Tomás and F. Casacuberta. Statistical phrase-based models for interactive computer-assisted translation. In *Proceedings of the Coling/ACL joint conference*, pages 835–841, Sydney, Australia, 17th-21th July 2006.

[Til01] C. Tillmann. *Word re-ordering and dynamic programming based search algorithms for statistical machine translation*. PhD thesis, Computer Science Department, RWTH Aachen, Germany, May 2001.

[Til03] C. Tillmann. A projection extension algorithm for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1–8, July 2003.

[TIM02] K. Toutanova, H. T. Ilhan, and C. Manning. Extensions to hmm-based statistical word alignment models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2002.

[Tom03] J. Tomás. *Traducción automática de textos entre lenguas similares utilizando métodos estadísticos*. PhD thesis, Universidad Politécnica de Valencia, Valencia (Spain), July 2003. (In spanish).

[Tru99] A. Trujillo. *Translation engines: techniques for machine translation*. Springer-Verlag, London, 1 edition, 1999.

[TVRV07] A. H. Toselli, V.Romero, L. Rodríguez, and E. Vidal. Computer assisted transcription of handwritten text. In *9th International Conference on Document Analysis and Recognition (ICDAR 2007)*, pages 944–948. IEEE Computer Society, Curitiba, Paraná (Brazil), September 2007.

[UM06] R. Udupa and H. K. Maji. Computational complexity of statistical machine translation. In *In Proceedings of the 11th Conference of the European Association for Machine Translation*, April 2006.

[UON02] N. Ueffing, F. Och, and H. Ney. Generation of word graphs in statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 156–163, 2002.

[Vau75] B. Vauquois. *La traduction automatique á Grenoble*. Dunod, Paris, 1 edition, 1975.

[Vid97] E. Vidal. Finite-state speech-to-speech translation. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 111–114, Munich, Germany, April 1997.

[Vil00] J. M. Vilar. Improve the learning of subsequential transducers by using alignments and dictionaries. In *ICGI '00*, pages 298–311. Springer-Verlag, 2000.

[Vit67] A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory*, 13:260–269, 1967.

[VNT96] S. Vogel, H. Ney, and C. Tillmann. HMM-based word alignment in statistical translation. In *Proceedings of The 16th International Conference on Computational Linguistics*, pages 836–841, Copenhagen, Denmark, August 1996.

[VP92] E. Vidal and N. Prieto. Learning language models through the ECGI method. *Speech Communication*, 11:299–309, 1992.

[VRCGV07] E. Vidal, L. Rodríguez, F. Casacuberta, and I. García-Varea. Interactive pattern recognition. In *Proceedings of the 4th Workshop on Machine Learning for Multimodal Interaction*, pages 60–71. Brno, Czech Republic, 28-30 June 2007.

[VTdlH$^+$05a] E. Vidal, F. Thollard, C. de la Higuera, F. Casacuberta, and R. Carrasco. Probabilistic finite-state machines - part I. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1013–1025, 2005.

[VTdlH$^+$05b] E. Vidal, F. Thollard, C. de la Higuera, F. Casacuberta, and R. Carrasco. Probabilistic finite-state machines - part II. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1025–1039, 2005.

[VV05] J. M. Vilar and E. Vidal. A recursive statistical translation model. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pages 199–207, Morristown, NJ, USA, 2005.

[VVW03] A. Venugopal, S. Vogel, and A. Waibel. Effective phrase translation extraction from alignment models. In *Proceedings of the 41th Annual Meeting of the Association for Computational Linguistics*, pages 319–326, Sapporo, Japan, July 2003.

[VZH$^+$03] S. Vogel, Y. Zhang, F. Huang, A. Tribble, A. Venugopal, B. Zhao, and A. Waibel. The CMU statistical machine translation system. In *Proceedings of the Machine Translation Summit IX*, pages 115–120, New Orleans, USA, September 2003.

[Wan98] Y. Wang. *Grammar inference and statistical machine translation*. PhD thesis, School of Computer Science, Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA, 1998.

[WMN10] J. Wuebker, A. Mauser, and H. Ney. Training phrase translation models with leaving-one-out. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 475–484, Uppsala, Sweden, July 2010.

[WSTI07] T. Watanabe, J. Suzuki, H. Tsukada, and H. Isozaki. Online large-margin training for statistical machine translation. In *Proceedings of the EMNLP-CONLL joint conference*, pages 764–733, Prage, Czeck Republic, 2007.

[WT90] G. C. G. Wei and M. A. Tanner. A monte carlo implementation of the em algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association*, 85(411):699–704, September 1990.

[Wu83] C. F. Jeff Wu. On the convergence properties of the EM algorithm. *The Annals of Statistics*, 11(1):95–103, 1983.

[Wu96] D. Wu. A polynomial-time algorithm for statistical machine translation. In *Proc. of the 34th Annual Conf. of the Association for Computational Linguistics*, pages 152–158, Santa Cruz, CA, June 1996.

[Wu97] D. Wu. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403, 1997.

[WW97] Y. Wang and A. Waibel. Decoding algorithm in statistical translation. In *Proc. 35th Annual Conference of the Association for Computational Linguistics*, pages 366–372, Madrid, Spain, July 1997.

[WW98] D. Wu and H. Wong. Machine translation with a stochastic grammatical channel. In *Proceedings of the COLING-ACL joint conference*, pages 1408–1415, 1998.

[XLL06] D. Xiong, Q. Liu, and S. Lin. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics*, 2006.

[YK01] K. Yamada and K. Knight. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 523–530, Toulouse, France, July 2001.

[Zen07] R. Zens. *Phrase-based statistical machine translation: models, search, training*. PhD thesis, Computer Science Department, RWTH Aachen, Germany, 2007.

[ZG05] H. Zhang and D. Gildea. Stochastic lexicalized inversion transduction grammar for alignment. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, 2005.

[ZN07] R. Zens and H. Ney. Efficient phrase-table representation for machine translation with applications to online MT and speech translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, pages 492–499, Rochester, New York, April 2007.

[ZON02] R. Zens, F. J. Och, and H. Ney. Phrase-based statistical machine translation. In *Advances in artificial intelligence. 25. Annual German Conference on AI*, volume 2479 of *LNCS*, pages 18–32. Springer Verlag, September 2002.

[ZV05] Y. Zhang and S. Vogel. An efficient phrase-to-phrase alignment model for arbitrarily long phrase and large corpora. In *Proceedings of the 10th Conference of the European Association for Machine Translation*, pages 30–31, 2005.

Part V

Appendices

Appendix A
Incremental EM Algorithm for HMM Alignment Model

This appendix shows the details of the derivation of the incremental EM algorithm for HMM-based alignment models. Specifically, we have applied the EM iteration given by (7.8). The general details of such derivation were presented in Chapter 7.

A.1 Sufficient Statistics

The application of the incremental view of the EM algorithm given by iteration (7.8) requires the definition of a vector of sufficient statistics for HMM-based alignment model. The set of parameters Θ for HMM-based alignment models is defined by Equation (7.20).

The vector of sufficient statistics for Θ, $s(\mathbf{f}, \mathbf{e}, \mathbf{a}) = \sum_n s_n(\mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n)$, is obtained as the sum of a set of counts for each training sample, $s_n(\mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n)$. Such set of counts for the sample n is given by Equation (7.24) and includes counts of aligned words, $c(f|e; \mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n)$, and counts of the width of alignment jumps, $c(i|i', I; \mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n)$. It can be demonstrated that $s(\mathbf{f}, \mathbf{e}, \mathbf{a})$ constitutes a sufficient statistic by means of the Fisher-Neyman factorisation theorem.

Theorem 1 (Fisher-Neyman factorization theorem) *Let $f(\Theta, \mathbf{x})$ be the density or mass function for the random vector $\mathbf{x}$, parametrised by the vector Θ. The statistic $s(\mathbf{x})$ is sufficient for Θ if and only if there exist functions $a(\mathbf{x})$ (not depending on Θ) and $b(\Theta, s(\mathbf{x}))$ such that*

$$f(\Theta, \mathbf{x}) = a(\mathbf{x}) \cdot b(\Theta, s(\mathbf{x}))$$

for all possible values of $\mathbf{x}$.

We use the Fisher-Neyman factorisation theorem to demonstrate that $s(\mathbf{f}, \mathbf{e}, \mathbf{a})$ constitutes a vector of sufficient statistics for the HMM-based alignment model. For this purpose, the log-likelihood function of the complete data that can be obtained by combining equations (7.22) and (7.23) is represented in terms of the sufficient statistics:

$$\begin{aligned}\mathcal{L}(\Theta,\mathbf{f},\mathbf{e},\mathbf{a}) = &\sum_{n=1}^{N}\sum_{j=1}^{|\mathbf{f}_n|}\sum_{i=1}^{|\mathbf{e}_n|}\mathbf{a}_{nji}\cdot\log p(\mathbf{f}_{nj}|\mathbf{e}_{ni})+\\ &\sum_{i'=1}^{|\mathbf{e}_n|}(\mathbf{a}_{n(j-1)i'}\,\mathbf{a}_{nji})\cdot\log p(i|i',|\mathbf{e}_n|) \qquad (A.1)\\ =&\sum_{f\in\mathcal{F}}\sum_{e\in\mathcal{E}}\sum_{n=1}^{N}c(f|e;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)\cdot\log p(f|e)+\\ &\sum_{\forall I}\sum_{i=1}^{I}\sum_{i'=0}^{I}\sum_{n=1}^{N}c(i|i',I;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)\cdot\log p(i|i',I) \qquad (A.2)\\ &= a(\mathbf{f},\mathbf{e},\mathbf{a})\cdot b(\Theta,s(\mathbf{f},\mathbf{e},\mathbf{a})) \qquad (A.3)\end{aligned}$$

where $a(\mathbf{f},\mathbf{e},\mathbf{a}) = 1$ and $b(\Theta,s(\mathbf{f},\mathbf{e},\mathbf{a}))$ is equal to the right-hand side of Equation (A.2). By the Fisher-Neyman factorisation theorem, $s(\mathbf{f},\mathbf{e},\mathbf{a})$ is sufficient for Θ.

A.2 E step

The E step of the incremental EM algorithm requires the computation of the expected values of the sufficient statistics for each data item, $\tilde{s}_n^{(t)} = E_{q_n}[s_n(\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)]$, where $q_n(\mathbf{a}_n) = p(\mathbf{a}_n|\mathbf{f}_n,\mathbf{e}_n,\Theta^{(t-1)})$; $\tilde{s}_n^{(t)}$ includes expected counts of the form $c(f|e;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)^{(t)}$ and $c(i|i',I;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)^{(t)}$.

To compute $c(f|e;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)^{(t)}$ and $c(i|i',I;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)^{(t)}$, the expected values of $\mathbf{a}_{nji}$ and $(\mathbf{a}_{n(j-1)i'}\,\mathbf{a}_{nji})$, respectively, are to be calculated.

The term $\mathbf{a}_{nji}^{(t)}$ is calculated as follows:

$$\begin{aligned}\mathbf{a}_{nji}^{(t)} &= p(\mathbf{a}_{nji}=1\,|\,\mathbf{f}_n,\mathbf{e}_n,\Theta^{(t)})\\ &= \frac{p(\mathbf{f}_n,\mathbf{a}_{nji}=1\,|\,\mathbf{e}_n,\Theta^{(t)})}{\sum_{\tilde{\imath}=1}^{|\mathbf{e}_n|}p(\mathbf{f}_n,\mathbf{a}_{nj\tilde{\imath}}=1\,|\,\mathbf{e}_n,\Theta^{(t)})}\\ &= \frac{p(\mathbf{f}_{n1}^{\,j},\mathbf{a}_{nji}=1\,|\,\mathbf{e}_n,\Theta^{(t)})\cdot p(\mathbf{f}_{n(j+1)}^{|\mathbf{f}_n|}\,|\,\mathbf{f}_{n1}^{\,j},\mathbf{a}_{nji}=1,\mathbf{e}_n,\Theta^{(t)})}{\sum_{\tilde{\imath}=1}^{|\mathbf{e}_n|}p(\mathbf{f}_{n1}^{\,j},\mathbf{a}_{nj\tilde{\imath}}=1\,|\,\mathbf{e}_n,\Theta^{(t)})\cdot p(\mathbf{f}_{n(j+1)}^{|\mathbf{f}_n|}\,|\,\mathbf{f}_{n1}^{\,j},\mathbf{a}_{nj\tilde{\imath}}=1,\mathbf{e}_n,\Theta^{(t)})}\\ &= \frac{\alpha_{nji}\cdot\beta_{nji}}{\sum_{\tilde{\imath}=1}^{|\mathbf{e}_n|}\alpha_{nj\tilde{\imath}}\cdot\beta_{nj\tilde{\imath}}} \qquad (A.4)\end{aligned}$$

where the α and β recursive functions are given by Equations (7.32) and (7.33), respectively.

The term $(\mathbf{a}_{n(j-1)i'}\,\mathbf{a}_{nji})^{(t)}$ is given by the following expression:

$$\begin{aligned}(\mathbf{a}_{n(j-1)i'}\mathbf{a}_{nji})^{(t)} &= p(\mathbf{a}_{n(j-1)i'} = 1, \mathbf{a}_{nji} = 1 \,|\, \mathbf{f}_n, \mathbf{e}_n, \Theta^{(t)}) \\ &= \frac{p(\mathbf{f}_n, \mathbf{a}_{n(j-1)i'} = 1, \mathbf{a}_{nji} = 1 \,|\, \mathbf{e}_n, \Theta^{(t)})}{\sum_{\tilde{\imath}'=1}^{|\mathbf{e}_n|}\sum_{\tilde{\imath}=1}^{|\mathbf{e}_n|} p(\mathbf{f}_n, \mathbf{a}_{n(j-1)\tilde{\imath}'} = 1, \mathbf{a}_{nj\tilde{\imath}} = 1 \,|\, \mathbf{e}_n, \Theta^{(t)})}\end{aligned} \tag{A.5}$$

where

$$\begin{aligned}p(\mathbf{f}_n, \mathbf{a}_{n(j-1)i'} = 1, \mathbf{a}_{nji} = 1 \,|\, \mathbf{e}_n;\Theta^{(t)}) = p(&\mathbf{f}_{n1}^{(j-1)}, \mathbf{a}_{n(j-1)i'} = 1 \,|\, \mathbf{e}_n, \Theta^{(t)})\cdot \\ &p(\mathbf{a}_{nji} = 1 \,|\, \mathbf{f}_{n1}^{(j-1)}, \mathbf{a}_{n(j-1)i'} = 1, \mathbf{e}_n, \Theta^{(t)})\cdot \\ &p(\mathbf{f}_{nj} \,|\, \mathbf{f}_{n1}^{(j-1)}, \mathbf{a}_{n(j-1)i'} = 1, \mathbf{a}_{nji} = 1, \mathbf{e}_n, \Theta^{(t)})\cdot \\ &p(\mathbf{f}_{n(j+1)}^{|\mathbf{f}_n|} \,|\, \mathbf{f}_{n1}^{j}, \mathbf{a}_{n(j-1)i'} = 1, \mathbf{a}_{nji} = 1, \mathbf{e}_n, \Theta^{(t)}) \\ =&\alpha_{n(j-1)i'}\, p(i|i', |\mathbf{e}_n|)^{(t)}\, p(\mathbf{f}_{nj}|\mathbf{e}_{ni})^{(t)}\, \beta_{nji}\end{aligned} \tag{A.6}$$

Then,

$$(\mathbf{a}_{n(j-1)i'}\mathbf{a}_{nji})^{(t)} = \frac{\alpha_{n(j-1)i'} \cdot p(i|i', |\mathbf{e}_n|)^{(t)} \cdot p(\mathbf{f}_{nj}|\mathbf{e}_{ni})^{(t)} \cdot \beta_{nji}}{\sum_{\tilde{\imath}'=1}^{|\mathbf{e}_n|}\sum_{\tilde{\imath}=1}^{|\mathbf{e}_n|} \alpha_{n(j-1)\tilde{\imath}'} \cdot p(\tilde{\imath} \,|\, \tilde{\imath}', |\mathbf{e}_n|)^{(t)} \cdot p(\mathbf{f}_{nj}|\mathbf{e}_{n\tilde{\imath}})^{(t)} \cdot \beta_{nj\tilde{\imath}}} \tag{A.7}$$

A.3 M step

The M step of the incremental EM algorithm obtains the model parameters, $\Theta^{(t)}$, that maximises the log-likelihood of the complete data given the expected values of the sufficient statistics, $\tilde{s}^{(t)}$.

If we replace s_n by $\tilde{s}_n^{(t)}$ in Equation A.2, we obtain the function $Q(\Theta|\Theta^{(t-1)})$ expressed in terms of the sufficient statistics:

$$\begin{aligned}Q(\Theta|\Theta^{(t-1)}) = &\sum_{f\in\mathcal{F}}\sum_{e\in\mathcal{E}}\sum_{n=1}^{N} c(f|e; \mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n)^{(t)} \cdot \log p(f|e) \\ &+ \sum_{\forall I}\sum_{i=1}^{I}\sum_{i'=0}^{I}\sum_{n=1}^{N} c(i|i', I; \mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n)^{(t)} \cdot \log p(i|i', I)\end{aligned} \tag{A.8}$$

(A.9)

The maximum-likelihood parameters are given by:

$$\Theta = \arg\max_{\Theta}\{Q(\Theta|\Theta^{(t-1)})\} \tag{A.10}$$

where Θ includes lexical and alignment parameters that are subject to the following constraints:

$$\begin{array}{ll} \sum\limits_{f \in \mathcal{F}} p(f|e) = 1 & \forall\, e \\ \sum\limits_{i=1}^{I} p(i|i', I) = 1 & \forall\, 1 \leq i' \leq I \text{ and } I \end{array} \tag{A.11}$$

To ensure that the previous constraints are satisfied, we define Lagrange multipliers:

$$\Lambda = \left\{ \begin{array}{l} -\sum\limits_{e \in \mathcal{E}} \lambda_e \left(\sum\limits_{f \in \mathcal{F}} p(f|e) - 1 \right) \\ -\sum\limits_{i'=1}^{I} \sum\limits_{I} \lambda_{i'I} \left(\sum\limits_{i=1}^{I} p(i|i', I) - 1 \right) \end{array} \right. \tag{A.12}$$

which are introduced in Equation (A.10), resulting in the following expression:

$$\Theta = \arg\max_{\Theta} \{ \max_{\lambda} Q(\Theta|\Theta^{(t-1)}) + \Lambda \} \tag{A.13}$$

The maximum-likelihood parameters are obtained by taking derivatives of Equation (A.13) with respect to Θ and Λ and equating to zero.

According to this maximisation procedure, the lexical parameters, $p(f|e)$, are updated as follows:

$$\frac{\partial Q(\Theta \,|\, \Theta^{(t-1)}) + \Lambda}{\partial p(f|e)} = \sum_{n=1}^{N} c(f|e; \mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n)^{(t)} \cdot p(f|e)^{-1} - \lambda_e = 0 \tag{A.14}$$

$$\frac{\partial Q(\Theta \,|\, \Theta^{(t-1)}) + \Lambda}{\partial \lambda_e} = \sum_{f \in \mathcal{F}} p(f|e) - 1 = 0 \tag{A.15}$$

Reorganising terms

$$p(f|e) = \lambda_e^{-1} \sum_{n=1}^{N} c(f|e; \mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n)^{(t)} \tag{A.16}$$

$$\sum_{f \in \mathcal{F}} p(f|e) = 1 \tag{A.17}$$

substituting $p(f|e)$ by $\lambda_e^{-1} \sum\limits_{n=1}^{N} c(f|e; \mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n)^{(t)}$ in Equation (A.17), we get:

$$\lambda_e^{-1} \sum_{f \in \mathcal{F}} \sum_{n=1}^{N} c(f|e; \mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n)^{(t)} = 1 \tag{A.18}$$

where

$$\lambda_e = \sum_{f \in \mathcal{F}} \sum_{n=1}^{N} c(f|e; \mathbf{f}_n, \mathbf{e}_n, \mathbf{a}_n)^{(t)} \tag{A.19}$$

Replacing λ_e into Equation (A.16) we obtain

$$p(f|e)^{(t)} = \frac{\sum\limits_{n=1}^{N} c(f|e;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)^{(t)}}{\sum\limits_{f'\in\mathcal{F}}\sum\limits_{n=1}^{N} c(f'|e;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)^{(t)}} \tag{A.20}$$

Finally, the update equations for the alignment parameters, $p(i|i',I)$, are given by:

$$\frac{\partial Q(\Theta\,|\,\Theta^{(t-1)})+\Lambda}{\partial p(i|i',I)} = \sum_{n=1}^{N} c(i|i',I;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)^{(t)}\cdot p(i|i',I)^{-1} - \lambda_{i'I} = 0 \tag{A.21}$$

$$\frac{\partial Q(\Theta\,|\,\Theta^{(t-1)})+\Lambda}{\partial \lambda_{i'I}} = \sum_{i=1}^{I} p(i|i',I) - 1 = 0 \tag{A.22}$$

so

$$p(i|i',I) = \lambda_{i'I}^{-1}\sum_{n=1}^{N} c(i|i',I;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)^{(t)} \tag{A.23}$$

$$\sum_{i=1}^{I} p(i|i',I) = 1 \tag{A.24}$$

substituting $p(i|i',I)$ by $\lambda_{i'I}^{-1}\sum\limits_{n=1}^{N} c(i|i',I;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)^{(t)}$ in Equation (A.24), we obtain

$$\lambda_{i'I}^{-1}\sum_{i=1}^{I}\sum_{n=1}^{N} c(i|i',I;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)^{(t)} = 1 \tag{A.25}$$

where

$$\lambda_{i'I} = \sum_{i=1}^{I}\sum_{n=1}^{N} c(i|i',I;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)^{(t)} \tag{A.26}$$

and replacing $\lambda_{i'I}$ into Equation (A.23), we obtain

$$p(i|i',I)^{(t)} = \frac{\sum\limits_{n=1}^{N} c(i|i',I;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)^{(t)}}{\sum\limits_{\tilde{\imath}=1}^{I}\sum\limits_{n=1}^{N} c(\tilde{\imath}|i',I;\mathbf{f}_n,\mathbf{e}_n,\mathbf{a}_n)^{(t)}} \tag{A.27}$$

APPENDIX B
THE OPEN-SOURCE THOT TOOLKIT

Open-source software constitutes a valuable resource for the researchers. We have developed the open-source THOT toolkit for PB-SMT, which is freely and publicly available and it has been extensively used throughout this thesis.

B.1 Design Principles

The open-source THOT toolkit has been developed using the C++ and the AWK programming languages. The design principles that have led the development process were:

- **Modularity and extensibility**: The THOT code is organised into classes for each aspect of its main functionality. Abstract classes are used when appropriate to define the basic behaviour of the functional components of the toolkit. In addition to this, the use of abstract classes also allows to easily extend the toolkit functionality by means of the well-known object-oriented programming mechanism of inheritance.

- **Flexibility**: it works with different and well-known data formats, including data formats used by the well-known GIZA++, Pharaoh or Moses toolkits.

- **Usability**: the toolkit functionality is easy to use, the code is easy to incorporate to new code.

- **Portability**: It is known to compile in the following architectures: Linux (tested for different kernel versions), Windows (using cygwin), Sun Sparc, Sun Solaris, MacOS X, DEC Alpha and FreeBSD.

B.2 Toolkit Functionality

The THOT toolkit implements the following functionality:

- **Operations between alignments**: As stated in section 3.2, it is common to apply operations between alignments in order to make them better. The toolkit provides the following operations:
  - **Union**: Obtains the union of two matrices.
  - **Intersection**: Obtains the intersection of two matrices.
  - **Sum**: Obtains the sum of two or more matrices.
  - **Symmetrisation**: Obtains *something* between the union and the intersection of two matrices. It was defined in [Och02] for the first time, and there exist different versions.
- **RF and BRF estimation**: The THOT toolkit implements both the relative frequency (RF) and the bisegmentation-based RF (BRF) estimation techniques described in sections 3.2 and 3.3 respectively.
- **Scalable estimation**: The estimation techniques implemented by the THOT toolkit can be applied on very large corpora by means of the techniques described in section 3.4. The maximum size of the corpus is only restricted by available disk space.
- **Parallel estimation**: The THOT toolkit incorporates a specific utility that allows the parallel execution of the main functionality of the toolkit on multiprocessors or PBS (portable batch system) clusters.
- **Phrase-based models library**: The toolkit provides a library of functions that allows us to generate phrase-based models as well as to access the statistical parameters contained in them.

B.3 Documentation

The THOT toolkit includes the following documentation resources:

- A tutorial in pdf format.
- README file and man pages.
- Class diagrams showing the software architecture.

B.4 Public Availability and License

The THOT toolkit is released under the GNU General Public License (GPL)[a]. The toolkit can be downloaded at http://sourceforge.net/projects/thot/.

[a] For more information on the GPL, see http://www.gnu.org/copyleft/gpl.html

Appendix C
Web-based Interactive Machine Translation Prototype

This appendix describes the main features of a web-based IMT prototype that has been developed following the techniques proposed in this thesis. This prototype is not intended to be a production-ready application. Rather, it provides an intuitive interface which aims at showing that the IMT approaches presented in this thesis can work in practise. The prototype is publicly available at http://cat.iti.upv.es/imt/.

It is important to stress here that this prototype has been developed with the collaboration of other persons. Specifically, the web interface has been developed by Luis A. Leiva and the application programming interface (API) that allows client and server applications to communicate through sockets has been developed by Vicent Alabau.

The rest of the appendix is organised as follows: first, the system architecture is introduced. Second, we describe the protocol that rules the interaction process. Next, the prototype functionalities are enumerated, and finally we describe the interface of the prototype.

C.1 System Architecture

The system architecture has been built on two main aspects, namely, accessibility and flexibility. The former is necessary to reach a larger number of potential users. The latter allows the researchers to test different techniques and interaction protocols reducing the implementation effort.

For that reason, an application programming interface for CAT tools was developed. This API allows a neat separation between the client interface and the actual translation system by using a network communication protocol and by exposing a well-defined set of functions. Furthermore, it allows the customisation of professional tools to use the IMT system with minimal implementation effort.

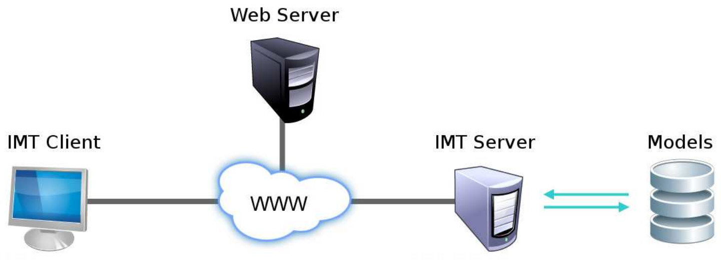


Figure C.1: IMT system architecture.

A diagram of the architecture is shown in Figure C.1. On the one hand, the IMT client provides a user interface (UI) which uses the API to communicate with the IMT server through the Web. The hardware requirements in the client are very low, as the translation process is carried out remotely on the server, so virtually any computer (including netbooks, tablets or 3G mobile phones) should be enough. On the other hand, the server, which is unaware of the implementation details of the IMT client, uses the models and the statistical methods described in this thesis to perform the translation.

C.2 User Interaction Protocol

The protocol that rules the IMT process has the following steps:

1. The system proposes a full translation of the selected text segment.
2. The user validates the longest prefix of the translation which is error-free and/or corrects the first error in the suffix. Corrections are entered by amendment keystrokes or mouse-clicks operations.
3. In this way, a new extended consolidated prefix is produced based on the previous validated prefix and the interaction amendments. Using this new prefix, the system suggests a suitable continuation of it.
4. Steps 2 and 3 are iterated until the user-desired translation is produced.

C.3 Prototype Functionality

The following is a list of the features that the prototype supports:

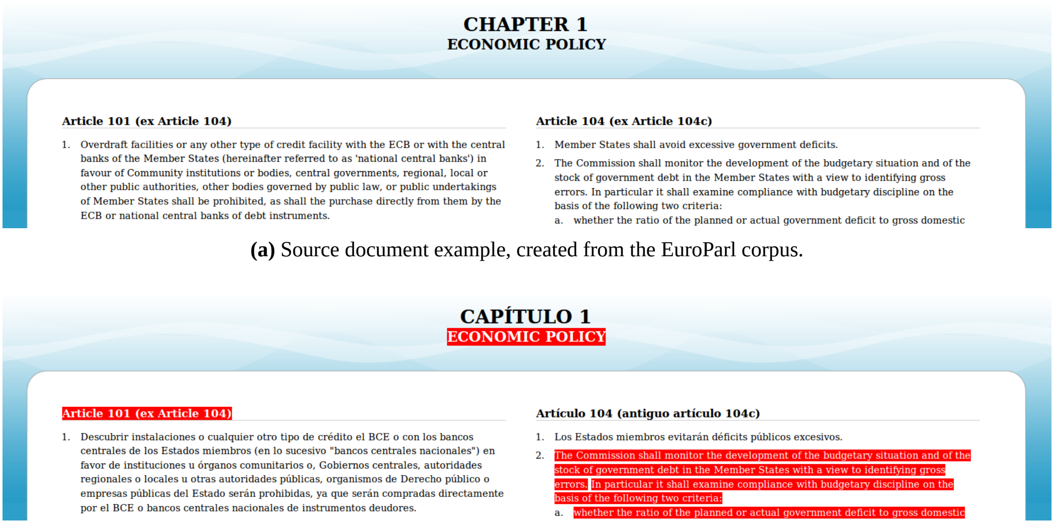
- When the user corrects the solution proposed by the system, a new improved suffix is proposed.
- The user is able to perform *actions* by means of keyboard shortcuts or mouse gestures. The supported actions on the proposed suffix are:

Substitution Substitute the first word or character of the suffix.

Deletion Delete the first word or character of the suffix.

Insertion Insert a word before the suffix.

- At any time, the user is able to visualise the original document (Figure C.2a), as well as a draft of the current translation properly formatted (Figure C.2b).
- A list of documents is presented to the user (Figure C.3) so that she can test the prototype under different conditions, i.e. corpora and language pairs.



CHAPTER 1
ECONOMIC POLICY

Article 101 (ex Article 104)

1. Overdraft facilities or any other type of credit facility with the ECB or with the central banks of the Member States (hereinafter referred to as 'national central banks') in favour of Community institutions or bodies, central governments, regional, local or other public authorities, other bodies governed by public law, or public undertakings of Member States shall be prohibited, as shall the purchase directly from them by the ECB or national central banks of debt instruments.

Article 104 (ex Article 104c)

1. Member States shall avoid excessive government deficits.
2. The Commission shall monitor the development of the budgetary situation and of the stock of government debt in the Member States with a view to identifying gross errors. In particular it shall examine compliance with budgetary discipline on the basis of the following two criteria:
   a. whether the ratio of the planned or actual government deficit to gross domestic

(a) Source document example, created from the EuroParl corpus.

CAPÍTULO 1
ECONOMIC POLICY

Article 101 (ex Article 104)

1. Descubrir instalaciones o cualquier otro tipo de crédito el BCE o con los bancos centrales de los Estados miembros (en lo sucesivo "bancos centrales nacionales") en favor de instituciones u órganos comunitarios o, Gobiernos centrales, autoridades regionales o locales u otras autoridades públicas, organismos de Derecho público o empresas públicas del Estado serán prohibidas, ya que serán compradas directamente por el BCE o bancos centrales nacionales de instrumentos deudores.

Artículo 104 (antiguo artículo 104c)

1. Los Estados miembros evitarán déficits públicos excesivos.
2. The Commission shall monitor the development of the budgetary situation and of the stock of government debt in the Member States with a view to identifying gross errors. In particular it shall examine compliance with budgetary discipline on the basis of the following two criteria:
   a. whether the ratio of the planned or actual government deficit to gross domestic

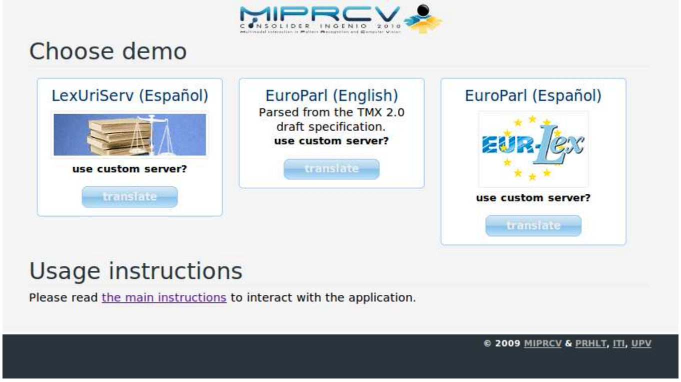
(b) Translated example document, preserving original format and highlighting non-translated sentences.

Figure C.2: Translating documents with the proposed system.

C.4 Prototype Interface

This prototype exploits the WWW to enable the connection of simultaneous accesses across the globe, coordinating client-side scripting with server-side technologies. The interface is built by using Web technologies such as HTML, JavaScript and ActionScript; while the IMT engine is written in C++ using the THOT toolkit.

To begin with, the Web UI (WUI) loads an index of all available translation corpora (Figure C.3). The user chooses a corpus and navigates to the main interface page (Figure C.4), where she starts translating the text segments one by one. User's feedback is then processed by the IMT server. Predictive interaction is approached in such a way that both the main and the feedback data streams help each-other to optimise overall performance and usability. All corrections are stored in plain text logs on the server, so the user can retake them in any moment, also allowing other users to help to translate the full documents.



Choose demo

LexUriServ (Español)

use custom server?

EuroParl (English)

Parsed from the TMX 2.0 draft specification.

use custom server?

EuroParl (Español)

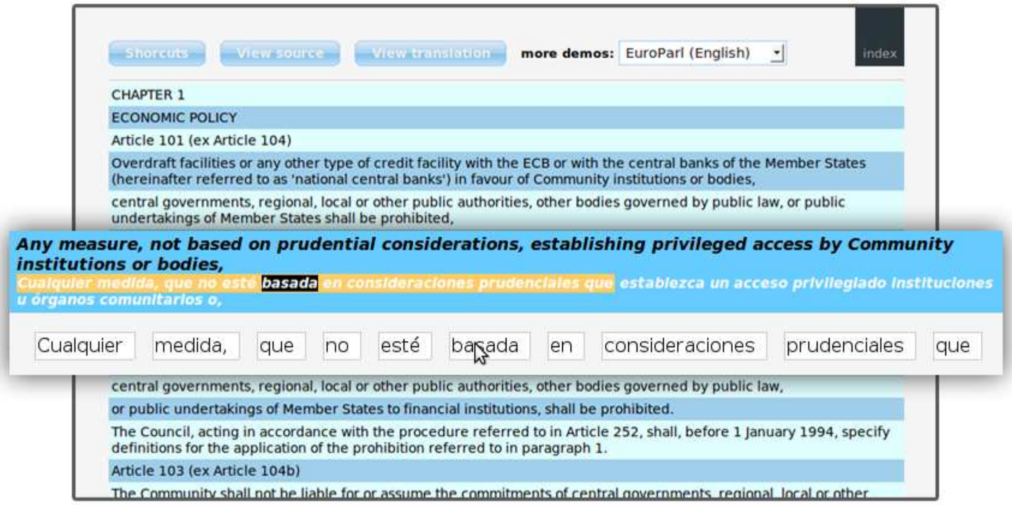
use custom server?

Usage instructions

Please read the main instructions to interact with the application.

© 2009 MIPRCV & PRHLT, ITI, UPV

Figure C.3: Index of available corpora.



more demos: EuroParl (English)

CHAPTER 1

ECONOMIC POLICY

Article 101 (ex Article 104)

Overdraft facilities or any other type of credit facility with the ECB or with the central banks of the Member States (hereinafter referred to as 'national central banks') in favour of Community institutions or bodies,

central governments, regional, local or other public authorities, other bodies governed by public law, or public undertakings of Member States shall be prohibited,

Any measure, not based on prudential considerations, establishing privileged access by Community institutions or bodies,

Cualquier medida, que no esté basada en consideraciones prudenciales que establezca un acceso privilegiado instituciones u órganos comunitarios o,

Cualquier | medida, | que | no | esté | basada | en | consideraciones | prudenciales | que

central governments, regional, local or other public authorities, other bodies governed by public law,

or public undertakings of Member States to financial institutions, shall be prohibited.

The Council, acting in accordance with the procedure referred to in Article 252, shall, before 1 January 1994, specify definitions for the application of the prohibition referred to in paragraph 1.

Article 103 (ex Article 104b)

The Community shall not be liable for or assume the commitments of central governments, regional, local or other

Figure C.4: Prototype interface. The source text segments are automatically extracted from source document.

Since the users operate within a Web browser, the system also provides crossplatform compatibility and requires neither computational power nor disk space on the client's machine. Client-Web server communication is based on asynchronous HTTP connections, providing thus a richer interactive experience (no page refreshes are required, only for changing the corpus to translate). Moreover, the Web server communicates with the IMT engine through binary TCP sockets. Thus, response times are quite slow (a desired requirement for the user's solace). Additionally, cross-domain requests are possible. In this way, it is possible to switch between different IMT engines from the same WUI.

APPENDIX D

SYMBOLS AND ACRONYMS

D.1 Mathematical Symbols

$\lvert \cdot \rvert$	cardinal of a set or word sequence
BOS	begin of sentence symbol
EOS	end of sentence symbol
$\mathbf{x} \equiv x_1 \dots x_i \dots x_{\lvert\mathbf{x}\rvert}$	data vector composed of $\lvert\mathbf{x}\rvert$ elements
x_i	i'th element of the data vector
$f_1^J \equiv f_1 \dots f_j \dots f_J$	source sentence composed of J words
$f_{j_1}^{j_2} \equiv f_{j_1} \dots f_{j_2}$	phrase of f_1^J, where $1 \leq j_1 \leq j_2 \leq J$
$\mathcal{F}$	source vocabulary
f	source word
J	length of the source sentence
f_j	j'th word of the source sentence
$e_1^I \equiv e_1 \dots e_i \dots e_I$	target sentence composed of I words
$e_{i_1}^{i_2} \equiv e_{i_1} \dots e_{i_2}$	phrase of e_1^I, where $1 \leq i_1 \leq i_2 \leq I$
$\mathcal{E}$	target vocabulary
e	target word
I	length of the target sentence
e_i	i'th word of the target sentence
e_0	null word of the target sentence
$a_1^J \equiv a_1 \dots a_j \dots a_J$	word alignment vector
a_j	j'th position of the word alignment vector
K	length of a phrase sequence
k	index for a phrase sequence
$\tilde{f}_1^K \equiv \tilde{f}_1 \dots \tilde{f}_k \dots \tilde{f}_K$	source phrase sequence
$\tilde{f}_k$	k'th source phrase
$\tilde{f}$	source phrase of an arbitrary length
$\tilde{e}_1^K \equiv \tilde{e}_1 \dots \tilde{e}_k \dots \tilde{e}_K$	target phrase sequence
$\tilde{e}_k$	k'th target phrase

$\tilde{e}$	target phrase of an arbitrary length
$\tilde{a}_1^K \equiv \tilde{a}_1 \ldots \tilde{a}_k \ldots \tilde{a}_K$	phrase alignment vector
$\tilde{a}_k$	k'th position of the phrase alignment vector
$\tilde{A}(f_1^J, e_1^I)$	bisegmentation or phrase-based alignment between f_1^J and e_1^I
$\tilde{A}_V(f_1^J, e_1^I)$	Viterbi phrase-based alignment between f_1^J and e_1^I
A	word alignment matrix
$\mathcal{BP}(f_1^J, e_1^I, A)$	set of consistent phrase pairs for f_1^J, e_1^I and A
$\mathcal{MS}_{f_1^J, e_1^I}$	set of monotonic bisegmentations for f_1^J, e_1^I
$\mathcal{S}_{f_1^J, e_1^I}$	set of bisegmentations for f_1^J, e_1^I
$\mathcal{S}_{\mathcal{BP}(f_1^J, e_1^I, A)}$	set of bisegmentations for f_1^J, e_1^I constrained to $\mathcal{BP}(f_1^J, e_1^I, A)$
$\mathcal{SP}$	set of unaligned source positions
$\mathcal{TP}$	set of unaligned target positions
$\mathbf{e}_p$	prefix of the target sentence
$\mathbf{e}_s$	suffix of the target sentence
$\mathcal{X}$	set of training samples
$(\mathbf{f}_n, \mathbf{e}_n)$	n'th training sentence pair
$\mathbf{f}_{nj}$	j'th word of $\mathbf{f}_n$
$\mathbf{e}_{ni}$	i'th word of $\mathbf{e}_n$
$\mathbf{a}_n$	alignment variable for the n'th training sentence pair
$\mathbf{a}_{nji}$	indicator variable for alignment of source position j with target position i corresponding to the n'th training pair
$Pr(\cdot)$	real probability distribution
$p(\cdot)$	model probability distribution
Θ	parameter vector for a model
$\mathcal{L}(\mathcal{X} \mid \Theta)$	likelihood for the set of training samples $\mathcal{X}$ given Θ
$c(\cdot)$	count of a given event

D.2 Acronyms

AER	alignment error rate
BLEU	bilingual evaluation understudy
BRF	bisegmentation-based relative frequency
CAT	computer-assisted translation
CER	character error rate
EM	expectation-maximisation
EU	European Union
HMM	hidden Markov model
GIS	generalized iterative scaling
IBM	international business machines
IMT	interactive machine translation
KSR	key-stroke ratio
KSMR	key-stroke and mouse-action ratio
MAR	mouse-action ratio
MERT	minimum error rate training
ML	maximum likelihood
MT	machine translation
NLP	natural language processing
PB-SMT	phrase-based statistical machine translation
PSPBA	partial statistical phrase-based alignment
PFSM	probabilistic finite state machine
PKSR	post-editing key stroke ratio
RF	relative frequency
SMT	statistical machine translation
SFST	stochastic finite state transducer
SCFG	synchronous context free grammar
WER	word error rate