A Modular Architecture for Separating Hypothesis Formation from Hypothesis Evaluation in Data-driven Machine Translation

Michael Carl and Paul Schmidt
Institut für Angewandte Informationsforschung
66111 Saarbrücken, Germany
{carl,paul}@iai.uni-sb.de

Abstract

Recent research in statistical- and example-based machine translation integrates rule-induced structured representations with statistics and lexicalised exceptions. While rule-based approaches concentrate on the formation of partial translation hypotheses, probabilistic approaches are concerned with the evaluation and selection of the best hypotheses.

Within the METIS-II framework, we propose a machine translation system which uses transfer and expander rules to build an AND/OR graph of partial translations and a statistical ranker to find the best path through the graph. The paper gives an overview of the architecture and an evaluation of the system for several languages.

1 Introduction

Recent machine translation techniques integrate rule-based knowledge and statistics: (Groves and Way, 2006) integrate rule-induced chunk translations with a statistical decoder; for (Richardson et al., 2001; Gamon et al., 2002), or (Ringger et al., 2004), linguistic rules describe what possible transformations a parse tree can undergo, but statistics decides under which conditions a particular rule is applied and (Quirk and Menezes, 2006) decide the combination of derivation trees by statistical means.

This paper outlines an MT architecture which uses rule-based devices to generate sets of partial translation hypotheses and a statistical *Ranker* to evaluate and retrieve the best hypotheses in their context.

The rule-based device generates an acyclic AND/OR graph which allows for compact representation of many different translations while the *Ranker* is a beam search algorithm which tries to find most likely paths in the AND/OR graph.

Unlike a usual statistical decoder (Germann et al., 2001; Koehn, 2004), our *Ranker* traverses the search graph to grade alternative paths and outputs a list of the n-best translations. The *Ranker* itself does not modify the graph. It does not permute chunks or items and it does not generate additional paths which are not already contained in the graph. The construction of the search graph and its evaluation are thus separated as two distinct tasks.

Starting from a SL sentence, the graph is incrementally constructed in three rule-based steps then evaluated and only for the n-best translations word tokens are generated:

1. The *Analyser* lemmatises and morphologically analyses the SL sentence. It produces a (flat) grammatical analysis of the sentence, detecting phrases and clauses and potential subject candidates. An outline of the *Analyser* is given in section 2.
2. The *Matcher* matches the analysed SL sentence on the transfer lexicon and retrieves TL equivalences. The *Matcher* has been described in (Carl and Rascu, 2006). We will give a review of the essential features in section 3.
3. The *Expander* inserts, deletes, moves and permutes items or chunks according to TL syntax. It is called *Expander* because it expands the search space through the word- and phrase translations retrieved from the lexicon. The *Expander* relies on a rule-based device. We give some examples in section 4
4. The *Ranker* relies on a beam search algorithm that iteratively traverses the graph and computes the most likely translations in a log-linear fashion (Och and Ney, 2002). The *Ranker* is explained in section 5.
5. Since the *Ranker* outputs lemmatised forms and PoS tags, we need a *Token Generator* to generate surface word-forms from the lemmas and PoS tags. The Token Generator has been described in (Carl et al., 2005) and will be omitted here.

2 The Analyser

The Analyser reads the SL sentence and produces a flat sequence of feature bundles which contains chunking and topological information of the sentence similar to the one described in (Becker and Frank, 2002; Müller, 2004). For instance, from the German

1a Das Haus wurde von Hans gekauft (The house was bought by Hans)
The house was from Hans bought

```
    {lu=das,      wnrr=1,  c=w,sc=art,         mark=np;subjF,    markcl=hs;vf}
    ,{lu=haus,    wnrr=2,  c=noun,             mark=np;subj,     markcl=hs;vf}
    ,{lu=werden,  wnra=3,  c=verb,vtyp=fiv,    mark=vg_fiv,      markcl=hs;lk}
1b  ,{lu=von,     wnrr=4,  c=w,sc=p,           mark=np;nosubjF,  markcl=hs;mf}
    ,{lu=Hans,    wnrr=5,  c=noun,             mark=np;nosubj,   markcl=hs;mf}
    ,{lu=kaufen,  wnra=6,  c=verb,vtyp=ptc2,   mark=vg_ptc,      markcl=hs;rk}
    .
```

Figure 1: A German sentence (1a) and the Output of the *Analyser* (1b). See text for explanation

SL sentence 1a the representation would be generated in figure 1 1b.

The analysis in 1b comprising among other things the lemma `lu` of the word, part-of-speech information `c`, `sc` and `vtyp`, as well as morphological and syntactic information. Chunking information is provided by the `mark` feature while the topological parser produces a linguistically motivated, flat macro structure of German, as coded by the `markcl` feature. The values indicate: `hs` main clause `vf` (Vorfeld) `mf` (Mittelfeld) `lk` (linke Klammer) and `rk` (rechte Klammer) (see (Müller, 2004) for in depth discussion).

3 The Matcher

The input of the matcher is a sequence of annotated SL words as generated from the *Analyser* (see example 1b in figure 1). The sequence of feature bundles is then matched on the lexicon as described in (Carl and Rascu, 2006).

When matching the SL representation on the dictionary and retrieving translation options a number of translation divergencies have to be coped with.

While the dictionary entries are coded as sequences of words, these words may actually be non-sequential in the sentence to be translated. The dictionary and the *Matcher* may regroup words in a sentence into coherent meaning entities which are distributed over several parts of the sentence and retrieve sets of translations for them. On the other hand, the dictionary provides many possible translation options and leaves it to the *Ranker* to decide which fits best the context. We give examples of these two mechanisms.

The *Matcher* retrieves all possible translation options for the sentence. It returns a structure as shown in figure 2. Each SL node is transformed into a node of an AND/OR graph which consists of a set of translation options. Each retrieved translation option is, in turn a flat tree. Translation options are clustered into translation units which are assigned to one SL node.

A translation option consists of a "mother" node

```
 {lu=das,wnrr=1,c=w,sc=art, ... }
    @{c=art,n=146471}@{lu=the,c=AT0}.
    .
,{lu=Haus,wnrr=2,c=noun, ...}
    @{c=noun,n=268244}@{lu=company,c=NN1}.
    ,{c=noun,n=268246}@{lu=home,c=NN1}.
    ,{c=noun,n=268247}@{lu=house,c=NN1}.
    ,{c=noun,n=268249}@{lu=site,c=NN1}.
    .
,{lu=werden,wnrr=3,c=verb,vtyp=fiv, ...}
    @{c=verb,n=604071}@{lu=be,c=VBD} .
    ,{c=verb,n=604076}@{lu=will,c=VM0} .
    .
,{ori=von,wnrr=4,c=w,sc=p, ...}
    @{c=w,sc=p,n=587268}@{lu=by,c=PRP}.
    ,{c=w,sc=p,n=587269}@{lu=from,c=PRP}.
    ,{c=w,sc=p,n=587270}@{lu=of,c=PRF}.
    .
,{ori=Hans,wnrr=5,c=noun, ...}
    @{c=noun,n=265524}@{lu=hans,c=NP0}.
    .
,{lu=kaufen,wnrr=6,c=verb,vtyp=ptc2, ...}
    @{c=verb,n=307263}@{lu=buy,c=VVN}.
    ,{c=verb,n=307265}@{lu=purchase,c=VVN}.
    .
.
```

Figure 2: Output of the *Matcher*: The SL analysis of figure 1 (example 1b) is transformed into an AND/OR graph enriched with retrieved translation options.

preceding the `@` and a sequence of leave nodes, following `@`. The "mother" node of a retrieved translation option contains information concerning the entry as a whole while the feature bundle(s) following `@` represent tagged and analysed words of the the actual entry.

3.1 Discontinuous matching

In this section we give some examples of discontinuous entries as they occur frequently in German (and Dutch). Note that the complexity for matching discontinuous phrases is much higher than for match-

ing continuous phrases. Matching a discontinuous phrase of length m on a sentence of length n may lead to a huge number of retrieved entries in the order of $O\left(\begin{array}{c} n \\ m \end{array}\right)$, while for continuous phrases there is a maximum of $(n-m)$ matches. Thus, there are more than 3000 possible ways to match a discontinuous phrase of 5 words on a 15-word sentence while a continuous phrase may lead to only 10 possible matches.

In addition, various permutations of the phrases are possible, according to whether the phrase is matched on a subordinate clause or a main clause. In (Carl and Rascu, 2006) we have described various strategies to reject matched entries if they don't obey a predefined set of criteria. Among other things, the *Matcher* tackles the following phenomena.

3.1.1 Detached prefixes

Detached verbal prefix as e.g. "ab" in example 2a occur at the end of the main clause as in 2b, while it stays with the verb in subordinate clauses (2c).

2a ablehnen ↔ reject
2b Hans **lehnt** das Angebot **ab**.
≈ *Hans rejects the offer* `prefix`.

2c dass Hans das Angebot **ablehnt**
≈ *that Hans the offer rejects.*
⇒ that Hans rejects the offer.

The matching algorithm must take into account these different realisations of the same word.

3.1.2 Reflexive verbs

The reflexive pronoun "sich" in the dictionary entry 3a may be dislocated from the finite verb as in the subordinate clauses 3b or it may be adjacent to it as in the main clause 3c.

3a sich beeilen ↔ hurry up
3b dass Hans **sich** immer **beeilt**.
≈ *that Hans himself always hurries up.*
⇒ that Hans always hurries up.

3c Hans **beeilt sich** morgens
⇒ Hans hurries up in the morning.

3.1.3 Light verbs and idioms

The predicate "vom Mund" and the verb "ablesen" may be non-adjacent in 4b or continuous in a subordinate clause 4c. Note that "ablesen" has a detachable prefix so that two discontinuous phenomena overlap here.

4a vom Mund ablesen ↔ lip-read
4b Hans **liest** ihr Wünsche **vom Mund ab**.
≈ *Hans read her wishes from mouth* `prefix`.
⇒ Hans lip-reads her wishes.

4c dass Hans ihr Wünsche **vom Mund abliest**.
≈ *that Hans her wishes from mouth reads.*
⇒ that Hans lip-reads her wishes.

3.2 Lexical Overgeneration

To account for a maximum number of different contexts, the dictionary over-generates translation hypotheses which are then filtered and graded by the *Ranker* in the context of the generated sentence. This section outlines some of the divergences tackled in the lexicon and its interaction with the *Ranker*.

3.2.1 Lexical semantic ambiguities

In examples 5a and 5b German "Bank" has two different translations "bank" and "bench". From the dictionary both translation options are retrieved and left to the *Ranker* to choose the appropriate translation in 5A and 5B.

5a Bank ↔ bank
5b Bank ↔ bench
5A Die **Bank** hat den Kredit gekündigt.
⇔ The **bank** has recalled the loan.
5B Hans sitzt auf der **Bank**.
⇔ Hans sits on the **bench**.

3.2.2 Negation

In order to translate negation from German to English the finite verb "do" has to be inserted and the German finite verb ("kommt") becomes an infinite English one. We tackle this by means of a multi-word transfer rule 6a.

6a nicht ↔ do not
6b nicht ↔ not
6A Hans kommt **nicht** ⇔ Hans does not come
≈ *Hans comes not.*

3.2.3 Magnifiers / Intensifiers

A specific problem of lexical ambiguities are 'intensifiers' or 'magnifiers'. For instance, the word "stark" (basically 'strong') can translate into many different adjectives depending on the context and the noun it modifies. Examples 7a-f show some of the possible translations and 7A-F provide contexts in which **stark** is translated differently.

3.2.4 Preposition

In some cases the choice of the preposition collocates with the verb as e.g. in "wait for", "depend on" "turn off", "fade out", etc. These are prepositions that are strongly bound to the verb and are lexicalised

7a stark ↔ strong
7b stark ↔ good
7c stark ↔ high
7d stark ↔ bad
7e stark ↔ heavy
7f stark ↔ big

7A Das ist ein **starker** Mann
⇔ This is a **strong** man
7B Es war sein **stärkstes** Theaterstück
⇔ It has been his **best** play
7C Paul hat **starkes** Fieber
⇔ Paul has **high** temperature
7D Das Auto war **stark** beschädigt
⇔ The car was **badly** damaged
7E Hans ist ein **starker** Raucher
⇔ John is a **heavy** smoker
7F Es gab es eine **starke** Nachfrage
⇔ There was a **big** demand

To cope with semantically bound prepositions, the dictionary generates various possible translation as in 8a-e. The choice of the 'correct' preposition is left to the *Ranker* to decide.

8e auf ↔ on;in;up;onto;upon;...
8A **auf** dem Tisch ⇔ **on** the table
8B **auf** dem Hof ⇔ **in** the yard

4 The expander

The *Expander* adds further translation hypotheses to AND/OR graph. It is a rule-based device, which takes as its input the output of the *Matcher*. The *Expander* essentially inserts, deletes and moves translation units in the graph. It also produces alternative partial translations.

A rule for reordering the verbal group in German main clauses is shown below. The *Expander* rule `ReorderFinVerb_hs` moves the translation of the participle "gekauft" in figure 2 in a main clause (`mark=hs`) behind the finite verb "wurde".

```
ReorderFinVerb_hs =
    Ve{mark=hs}e{mark=vg_fiv},
    *e{mark=hs}a{mark~=vg_ptc;vg_inf},
   ^Ie{mark=hs}e{mark=vg_inf},
    Pe{mark=hs}e{mark=vg_ptc}
:   p(move=V->VIP).
```

This is a rule of a formal framework FRED described in (Carl and Schmidt-Wigger, 1998). The rule maps on a sequence of translation units (TUs) starting with the finite verb `mark=vg_fiv` and followed by an optional infinitive verb (`mark=vg_inf`) and a participle `mark=vg_ptc`. Between the finite verb and the optional infinitive can be number of

```
 {lu=das,wnrr=1,c=w,sc=art, ... }
    @{c=art,n=146471}@{lu=the,c=AT0}.
    .
,{lu=Haus,wnrr=2,c=noun, ...}
    @{c=noun,n=268244}@{lu=company,c=NN1}.
    ,{c=noun,n=268246}@{lu=home,c=NN1}.
    ,{c=noun,n=268247}@{lu=house,c=NN1}.
    ,{c=noun,n=268249}@{lu=site,c=NN1}.
    .
,{lu=werden,wnrr=3,c=verb,vtyp=fiv, ...}
    @{c=verb,n=604071}@{lu=be,c=VBD} .
    ,{c=verb,n=604076}@{lu=will,c=VM0} .
    .
,{lu=kaufen,wnrr=6,c=verb,vtyp=ptc2, ...}
    @{c=verb,n=307263}@{lu=buy,c=VVN}.
    ,{c=verb,n=307265}@{lu=purchase,c=VVN}.
    .
,{ori=von,wnrr=4,c=w,sc=p, ...}
    @{c=w,sc=p,n=587268}@{lu=by,c=PRP}.
    ,{c=w,sc=p,n=587269}@{lu=from,c=PRP}.
    ,{c=w,sc=p,n=587270}@{lu=of,c=PRF}.
    .
,{ori=Hans,wnrr=5,c=noun, ...}
    @{c=noun,n=265524}@{lu=hans,c=NP0}.
    .
.
```

Figure 3: Output of the *Expander*: The participle (kaufen) is moved behind the finite verb (werden). A translation equals the concatenation of the leaves of the graph. Each path through the graph can only accommodate one translation option for each translation unit.

'non-verbs'. All nodes need to occur in the same main clause (`mark=hs`). The existential quantifier `e` requires that at least one reading of the TU must be compatible with the test, while the universal quantifier `a` requires there is no other reading of the TU than the one given in the test. The finite verb, the infinite verb and the participle are marked by the marker `V`, `I` and `P` respectively. The action part of the rule — following the colon — moves the marked nodes into the desired word order, so that the verbs are grouped together in their right order. While this operation deterministically moves the nodes in the graph, the formalism also allows operations to produce alternative permutations of sequences of TUs.

The rule `ReorderFinVerb_hs` transforms the *Matcher* output in figure 2 into the graph in figure 3 which yields the (correct) word order "The house was bought by Hans".

We have tested the system on four languages (Dutch, German, Greek and Spanish) into English. A separate set of *Expander* rules is used for each source language, consisting of five rules for Greek up to approx. 20 rules for German. The results of this experiment are discussed in section 6

The *Expander* rules roughly classify into the following sets.

4.1 Inserting and deleting spurious words

In some cases additional function words have to be inserted or deleted. In 9a an article is required while in 9b the preposition **of** is needed to mark the genitive.

9a Hans ist Lehrer ⇔ Hans is **a** teacher
9b der Wagen meines Bruders
⇔ the car **of** my brother

4.2 Adjusting the verbal group

Parts of the German verbal group appear in the "linke Klammer" and other parts in the "rechte Klammer". In main clauses, the "Mittelfeld" may intervene between these two parts. For English these parts have to be re-joined. The discussion of the rule `ReorderFinVerb_hs` shows how we tackle this. More examples are given here.

4.2.1 Passive voice

The finite verb (wurde) and the participle (gekauft) are detached in the main clause. For English these have to be joined as shown in example10b.

10a Das Haus **wurde** von Hans **gekauft**.
≈ *The house* **was** *by Hans* **bought**.
10b The house **was bought** by Hans.

4.2.2 Composed tense

Similarly, for composed tenses in the German main clause, the finite verb and the participle are detached (11a) and need to be joined for the English translation (11b).

11a Hans **hat** das Haus **gekauft**.
≈ *Hans* **has** *the house* **bought**.
11b Hans **has bought** the house.

4.2.3 Sentences with modal verbs

For modal sentences, the infinite verb has to be moved right behind the finite verb:

12a Hans **will** ein Haus **kaufen**.
≈ *Hans* **wants to** *a house* **buy**.
12b Hans **wants to buy** a house.

4.3 Adjusting the subject

In contrast to English, German allows one phrasal element to precede the finite verb, which may or may not be the subject of the sentence. In some cases we know the subject from the German analysis. In these cases we can deterministically move the subject to its right position. In other cases the German analysis provides several subject candidates. We generate a translation hypothesis for each possible permutation and let the *Ranker* decide which is the more likely subject.

4.3.1 Moving the subject

In example 13a the subject "Hans" follows the finite verb "kam". In English the subject needs to precede the finite verb, so that a re-ordering of the TUs are required. In order to adjust the sentence to English syntax, the subject needs to be moved in front of the verb, as in 13b.

13a Gestern kam **Hans** in das Büro.
Yesterday came **Hans** *into the office.*
13b Yesterday **Hans** came into the office.

4.3.2 Ambiguous subjects

In cases a sentence is tagged with several subject candidates, we generate multiple word orders and let the *Ranker* decide which is the more likely one. While 14a is the usual subject-verb-order, the accusative object may be topicalised as in 14b. Due to homonymy of some German accusative and nominative nouns we may not be able to disambiguate the correct subject.

14a Die Katze trinkt die Milch.
The cat drinks the milk.
14b Die Milch trinkt die Katze.
The milk drinks the cat.

We generate two translation hypotheses as in 14c with both subject candidates and leave it to the statistical *Ranker* to decide which is the more likely English sentence.

14c (the milk drink the cat.
| the cat drink the milk.)

5 The ranker

The *Ranker* works similar to a decoder as used in statistical machine translation. Och and Ney (2002) extend the noisy channel model of Brown et al. (1993) by adding weighing coefficients with feature functions and combining them in a log linear fashion. As a statistical decoder, the *Ranker* is a search procedure which seeks to find the target sentence $\hat{e}$ with the highest probability:

$$\hat{e} = \texttt{argmax} \sum_{m}^{M} w_m h_m(\cdot)$$

where h_m is a feature function and w_m is a weighing coefficient. The feature functions h_m can be independent and trained on separate data while the weighing coefficients w_m are used to fine-tune the system.

The *Ranker* is a beam-search algorithm which traverses the AND/OR graph in a breadth first manner. At each step the nodes are weighted by the feature functions and all expanded sentence prefixes are

```
<s id=3-0 lp="-9.227912">
the     AT0     146471
company NN1     268244
was     VBD     604071 PermFinVerb_hs
bought  VVN     307263 PermFinVerb_hs
by      PRP     587268 PermFinVerb_hs
hans    NP0     265524 PermFinVerb_hs
.       PUN     367491
</s>
<s id=3-1 lp="-9.682535">
the     AT0     146471
house   NN1     268247
was     VBD     604071 PermFinVerb_hs
purchased VVN   307265 PermFinVerb_hs
by      PRP     587268 PermFinVerb_hs
hans    NP0     265524 PermFinVerb_hs
.       PUN     367491
</s>
```

Figure 4: Output of the *Ranker*: the two best scored translations.

stored in the beam until its maximum width (currently 1000) is achieved. From there on only the highest weighted sentence are further expanded. We have experimented with various feature functions to weight the nodes and describe their settings in this section. An evaluation is given in section 6.

Output of the *Ranker* are the n-best graded translation paths through the graph. For the example in figures 3 the two best translations (word forms) are shown in figure 4. The output also indicates the resources used to generate the translations, among other things, the number of the translation entries and the *Expander* rules.

5.1 Language model

We have tested various language models, all of them making use of the BNC[1] and all are generated using the CMU language modelling toolkit[2]. The BNC is a tagged collection of texts making use of the CLAWS5 tag set which comprises roughly 70 different tags. The CMU language modelling toolkit generates n-gram language models (LMs) from tokenised texts. These LMs are then used as a feature function of the *Ranker*.

The CMU toolkit generates a vocabulary of up to 65535 words which occur most frequently in the training material. It supports open LMs which account for unknown words and closed LMs which assume all tokens to be known in the training material. A LM made up of CLAWS5 tags would be a closed language model since there are less than 70 different tags in this tag set and all tags are likely to occur in the training material.

The closed LMs assume that only items in the training data will occur in the test data, while open LMs save some of the probability mass for (unknown) words in the test data which did not occur in the training set. These words will be mapped on the item UNK.

We did not experiment with the various discounting strategies provided with the CMU toolkit; all our experiments used good-turing discounting.

To find suited LMs for our application, we have experimented with the following parameters:

- number of sentences: 100K, 1M and 6M
- different ways of preprocessing the BNC:
  - open token-based LM
  - closed mixed lemma-tag LM
  - closed mixed token-tag LM
  - orthogonal lemma-tag LM
- 3-gram and 4-gram token LMs and 4-gram, 5-gram and 6-gram PoS-tag LMs

5.1.1 Open token-based LM

The open token-based LM assumes (lower-cased) surface word-forms as the input to the *Ranker*. This requires token generation to take place on the output of the *Expander* previous to the *Ranker*.

5.1.2 Closed mixed token-tag model

The vocabulary of the closed mixed token-tag model consists of word tokens (thus the un-lemmatised BNC) but unknown words will be mapped on their CLAWS5 tag. That is, assume the reference set contains the sentence "John likes strawberries" but "strawberries" does not occur in the vocabulary of the 60000 most frequent tokens. Instead of letting the CMU toolkit map "strawberries" on the tag UNK, we would replace it by the CLAWS5 tag <NN2>. In this way we can generate and assume a finite number of different tokens (the 69 CLAWS5 tags plus the 60000 most frequent tokens in the reference set). Analogically at runtime, previous to the *Ranker*, we would generate tokens from the lemmas. The *Ranker* would consult the LM's vocabulary and map any unknown word on the CLAWS5 tag. (Manning and Schütze, 1999) suggest to map unknown words on two tags: one for numbers and all other unknown words on one other tag. With our strategy unknown tokens are mapped on many more tags. In this way we can make sure that any sentence contains only known tokens.

[1] The British National Corpus (BNC) consists of more than 100 million words in more than 6 million sentences `http://www.natcorp.ox.ac.uk/`

[2] which can be downloaded from `http://www.speech.cs.cmu.edu/SLM_info.html`

5.1.3 Closed mixed lemma-tag model

The closed mixed lemma-tag model works essentially similar to the Closed mixed token-tag model but makes use of the 60000 most frequent lemmas. Thus the above reference sentence would be lemmatised into "John like strawberry", and - given "strawberry" is not among the 60000 most frequent lemmas in the training corpus - it would be preprocessed into "John like <NN2>". At runtime, lemmas would be transformed into word tokens on the output of the *Ranker*.

5.1.4 Orthogonal lemma-tag model

In the orthogonal lemma-tag model we compute two LMs: a CLAWS5 tag n-gram model (LM_tag) and a lemma m-gram model (LM_lem). In addition we compute a cooccurrence weight of the lemmas given their tag according to the following equation:

$$w(lem, tag) = \frac{NL}{NL + C(lem)} * (C(lem, tag) + 1)$$

Where NL is half the number of different tags (i.e. 69/2), $C(lem)$ is the number of occurrences of the token in the BNC and $C(lem, tag)$ is the number of cooccurrences of a lemma and a tag. For instance the lemma "tape-recorder" has 103 occurrences in the BNC. The weights for "tape-recorder" given their tag are shown in the table below, where <*> accounts for the possibility that a lemma/tag occurs in the test translations but did not occur in the training set:

lemma	tag	#	w(lem, tag)
tape-recorder	AJ0	3	1.00363636
tape-recorder	NN1	87	22.08000000
tape-recorder	NN2	13	3.51272727
tape-recorder	<*>	0	0.25090909

The orthogonal lemma-tag model consists thus of three feature functions which are computed for each node in the beam:

$$w_1 * log(p(tag)) + w_2 * log(p(lem)) + w_3 * log(w(lem, tag))$$

In a first experiment we have compared these four LMs. The orthogonal lemma-tag model consistently showed the best results so that we gave up further experiments with the open token and the closed token-tag and lemma-tag models.

5.2 Lexical weights

In a set of further experiments we have included and tested feature functions for lexicon weights and for *Expander* rules. A widely used method to compute weights for lexicon entries is based on counting their relative frequencies. However, it is known that relative frequencies overestimate low occurrences and also, since we do not make use of translated texts which could give a clue for their "real" probabilities, counting token frequencies in a hand-made lexicon would even more heavily overestimate rare translation relations.

The idea is, therefore, to 'incrementally' modify the weights of lexicon entries based on the amendments of a posteditor (Och, 2003). This approach is similar to (Watanabe et al., 2003)[p 398], who enhance an MT system "by comparing a wrong translation and its correction". While Watanabe et al. (2003) compare the dependency structures of the translation and its correction, we compute a NIST score ($NIST_{trans}$) for the n-best translations and assign a relative NIST score as the weight $w(r)$ for the *Expander* rules and lexicon rules:

$$w(r) = 1/|T| \sum_{r \in T} \frac{NIST_{Trans}}{NIST_{Max}}$$

Thus, consider the above output of the *Ranker* in section 5 on page 5. Assume the first translation with `id=3-0` has a NIST score of 2.6447 and the second translations with `id=3-1` has a NIST score of 3.0949. Since 3.0949 is in the same time the maximum NIST score for this translation, `house` occurs only in the better scored translation, `company` occurs in the worse scored translation and `hans` occurs in both translations, the lexicon rules will be assigned the following weight:

r	tag	id	$w(r)$
company	NN1	268244	0.854
house	NN1	268247	1.0
hans	NP0	265524	0.927

6 Evaluation

We have experimented with various lemma- and tag-LMs. These models were generated based on sets of 100K, 1M and 6M sentences of the BNC. For all lemma models we have used 3-grams. For the 100K and 1M set we have also used a 4-gram model. Unfortunately, due to space restrictions, it was not possible to generate a 6M, 4-gram lemma model. For the tag models we have used 4, 5 and 6-grams.

We have tested the system on 50 Dutch, 50 Spanish and 50 Greek sentences with the 6M-n3 lemma- and the 6M-n6 tag models with the following results:

Language	BLEU	NIST
Dutch	0,4034	6,4489
Spanish	0,3701	5,7304
Greek	0.2138	5.1220

For a German test set of 200 sentences, various combinations of these models were run with several settings of weight combinations.

Preliminary results are resumed as follows. The more data is used for the lemma LM, the better is the

result. Strangely, this does not seem to hold true for the tag LM. Thus, the 100K tag model performs best in almost every combination. The best results are obtained with the parameters $w_1 = 0.5$, $w_2 = 0.06$ and $w_3 = 1.0$ using the largest 6M 3-gram lemma model. As can be seen in the table below, in this combination, the 100K 4-gram tag model produces best NIST and BLEU results while larger and higher n-gram tag models produce worse.

NIST	BLEU	lemma LM	tag LM
5.4801	0.1861	6M-n3	100K-n4
5.4450	0.1859	6M-n3	100K-n6
5.4645	0.1856	6M-n3	1M-n4
5.4683	0.1855	6M-n3	100K-n5
5.4526	0.1855	6M-n3	6M-n4
5.4509	0.1849	6M-n3	1M-n5
5.4359	0.1844	6M-n3	6M-n5
5.4165	0.1836	6M-n3	6M-n6

As of now we do not have an explanation for this behaviour and we are not aware of similar findings from other groups.

The differences in BLEU and NIST scores for the four languages is — besides their similarity to English and the length of the test sentences — also due to the quality, coverage and ambiguity of the lexicon. The table below shows that conditions are worst for German: the German test set has the longest sentences and the highest lexical ambiguity. There are on average 3.6 translation options per word. Note that this is, compared to a statistical MT system very little, where a word can have up to 100 or more translations. However, dictionaries for Greek, Dutch and Spanish produce on average less than 2 TOs per TU. On the other hand, with more than 1.3 tokens per TO, the German lexicon is almost a phrase-lexicon.

language	Tok/TO	TO/TU	∅ len.
Greek	1.0208	1.9959	9.5
Dutch	1.1258	1.9796	10.8
Spanish	1.1930	1.9506	7.8
German	1.3282	3.6352	13,2

7 Related work and conclusion

We have described a machine translation system within the METIS-II project which joins a transfer dictionary with simple reordering rules and a statistical ranker. A general overview of the METIS-II project is given in (Dirix et al., 2005) and in (Vandeghinste et al., 2006). More detailed descriptions of the various realisations of METIS-II are in (Badia et al., 2005; Markantonatou et al., 2006; Vandeghinste et al., 2007).

Our approach is similar to (Badia et al., 2005) in that both of us use n-gram language models to rank translation candidates. However, while we use a rule-based, heuristic *Expander* to generate sets of hypotheses on TL word order, (Badia et al., 2005) proposes a two layered algorithmic permutation of words and chunks: The first layer permutes words within a chunk; the second layer permutes chunks within a sentence. The ranker selects the most likely permutations. Also (Vandeghinste et al., 2007) assumes a kind of isomorphism between SL chunks and TL chunks. The translations of recursively embedded SL chunks are considered to be bags of bags. These structure bags are the input of the ranker which evaluates all permutations and outputs the 'best' serialisations of the structure.

Since, depending on how the SL is chunked, for many translations it is required to merge various bags, as described in sections 3 and 4, (Vandeghinste et al., 2007) also introduce a rule-based device to overcome cross-chunk translation divergences. The question then arises whether in addition to this heuristic mechanism, an algorithmic permutation is required which generates all possible sequences of the bags, or whether a heuristic permutation of some probable permutations, as we suggest in this paper, is already sufficient. While we control the generation of partial translations by means of rules and thus hope to produce only 'high quality' hypotheses, many of the algorithmic permutations are actually misleading. Much of research (Schack, 2004; Schack and Mechsner, 2006) and the history of word-based SMT (Brown et al., 1990) has shown that too many dimensions of freedom are in fact a hindrance to learning. Learning can only take place by restricting the number of free variables (Schack, 2004). Only when the the basic concepts are learned, the degree of free variables can be increased again. Yet, in Machine Translation it is still unclear which the optimum number of free variables is (Wu, 2006).

An approach related to METIS-II has also been suggested by (Carbonell et al., 2006). Like METIS-II their so-called "Context-based machine translation" also makes use of a transfer dictionary and a target language corpus. The dictionary provides basic word (and phrase) translations which are used to retrieve chunks from the target language corpus. From the best sequence of overlapping chunks the translation is generated. However, like phrase-based MT this approach does not adequately model discontinuous translations.

The core idea of our work is also similar to (Brown and Frederking, 1995) who use a statistical English Language Model (ELM) to select between alternate partial translations produced by three symbolic MT system from the PANGLOSS Mark III system. In contrast to their approach we build a search graph with flat reordering rules.

As in so-called "generation-heavy" translation (Habash, 2004), our expander rules tackle some of

the translation divergences thereby producing numerous partial translation hypotheses. This "symbolic overgeneration" is then constrained by a statistical ranker making use of several statistical feature functions.

Still another line of research related to ours is reported in (Liu et al., 2005). In their "Tree-to-String" approach they use a pre-processed bilingual corpus to match parsed SL sentences and retrieve sets of partial translation hypotheses. As in our work, the best path through these translation hypotheses is determined by a 'statistical generator', a log-linear combination of feature functions.

The common characteristics among these different systems is that basic target language configurations are generated and their best combination is determined by a statistical device.

This is consistent with a number of cognitive and linguistic theories: According to (Harris, 1988), linguistic events are generated through a set of 'basic configurations' "... whose structure is determined by the partial order constraint and whose distribution follows the probabilities associated with the likelihood constraint." (Pereira, 2000)

While, for Harris the 'basic configurations' correspond to elementary clauses, (Pereira, 2000) assumes that they also include 'hidden parameters' such as elements of a grammar. He states that: "a common characteristic of most of [information-theoretic and machine-learning] tasks is [to seek] a decision among a finite set of alternatives, or a ranking of alternatives." This is also in line with (Flach and Kakas, 1997) who point out that rule-based approaches are suited to generate hypotheses, while probabilistic approaches are concerned with evaluation and the selection of the best hypotheses.

This paper outlines the architecture and implementation of an MT system putting into practice this premise. The aim is to clearly separate the two tasks, provide a maximum number of resources to each of them and enable their fruitful complementation.

References

Toni Badia, Gemma Boleda, Maite Melero, and Antonio Oliver. 2005. An n-gram approach to exploiting monolingual corpus for MT. In *Proceedings of the Example-Based Machine Translation Workshop held in conjunction with the 10th Machine Translation Summit*, pages 1–8, Phuket, Tailand.

Markus Becker and Anette Frank. 2002. A stochastic topological parser for german. In *Proceedings of the 19th international conference on Computational linguistics*, pages 1–7.

Ralf Brown and Robert Frederking. 1995. Applying statistical English language modelling to symbolic machine translation. In *Proceedings of the 6th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI)*, pages 221–239, Leuven, Belgium.

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Della Pietra Vincent J., Frederick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16 (2):79–85.

Jaime Carbonell, Steve Klein, David Miller, Michael Steinbaum, Tomer Grassiany, and Jochen Frei. 2006. Context-based machine translation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, pages 19–28, Cambridge, Massachusetts.

Michael Carl and Ecaterina Rascu. 2006. A dictionary lookup strategy for translating discontinuous phrases. In *Proceedings of the European Association for Machine Translation*, pages 49–58, Oslo, Norway.

Michael Carl and Antje Schmidt-Wigger. 1998. Shallow Postmorphological Processing with KURD. In *Proceedings of NeMLaP3/CoNLL98*, pages 257–267, Sydney.

Michael Carl, Paul Schmidt, and Jörg Schütz. 2005. Reversible Template-based Shake & Bake Generation. In *In Proceedings of the Example-Based Machine Translation Workshop held in conjunction with the 10th Machine Translation Smmit*, pages 17–26, Phuket, Tailand.

Peter Dirix, Ineke Schuurman, and Vincent Vandeghinste. 2005. Metis II: Example-based machine translation using monolingual corpora - System description. In *Proceedings of the Example-Based Machine Translation Workshop held in conjunction with the 10th Machine Translation Summit*, pages 43–50, Phuket, Tailand.

Peter Flach and Antonis Kakas, 1997. *Abduction and Induction in AI, Workshop report.* `http://www.cs.bris.ac.uk/~flach/IJCAI97/IJCAI97report.html`.

Michael Gamon, Eric Ringger, Simon Corston-Oliver, and Robert Moore. 2002. Machine-learned contexts for linguistic operations in German sentence realization. In *Proceedings of the 40th annual ACL Conference*, pages 25–32, Philadelphia.

Ulrich Germann, Michael Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. 2001. Fast Decoding and Optimal Decoding for Machine Translation. In *Proceedings of the 39th ACL and 10th Conference of the European Chapter*, pages 228–235, Toulouse, France.

Declan Groves and Andy Way. 2006. Hybrid Data Driven Models of MT. *Machine Translation*, 19.

Nizar Habash. 2004. The use of a structural n-

gram language model in generation-heavy hybrid machine translation. In *Proceeding 3rd International Conference on Natural Language Generation (INLG '04), volume 3123 of LNAI, Springer*, pages 61–69.

Z. S. Harris. 1988. *Language and information.* Columbia University Press, New York.

Philipp Koehn. 2004. Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In *AMTA*, pages 115–124.

Zhanyi Liu, Haifeng Wang, and Hua Wu. 2005. Example-based Machine Translation Based on TSC and Statistical Generation. In *In Proceedings of the 10th Machine Translation Smmit*, pages 25–32, Phuket, Tailand.

Christopher Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing.* MIT, Cambridge, Massachusetts.

Stella Markantonatou, S Sofianopoulos, V. Spilioti, G. Tambouratzis, M. Vassiliou, and O. Yannoutsou. 2006. Using Patterns for Machine Translation. In *Proceedings of the 11th EAMT conference*, pages 239–246, Oslo, Norway.

Frank Henrik Müller, 2004. *Stylebook for the Tübingen Partially Parsed Corpus of Written German (TüPP-D/Z).* `http://www.sfb441.uni-tuebingen.de/a1/pub.html`.

Franz J. Och and Hermann Ney. 2002. Discriminative Training and Maximum Entropy Models for Statistical Machine Translation. In *Proceedings of the 40th annual ACL Conference*, pages 295–302, Philadelphia, PA.

Franz J. Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceeding of ACL.*

Fernando Pereira. 2000. Formal grammar and information theory: Together again? *Philosophical Transactions of the Royal Society*, 358(1769):1239–1253.

Christopher Quirk and Arul Menezes. 2006. Dependency Treelet Translation: The convergence of statistical and example-based machine translation? *Machine Translation*, 20(1).

Stephen Richardson, William Dolan, Arul Menezes, and Jessie Pinkham. 2001. Achieving commercial-quality translation with example-based methods. In *Proceedings of Machine Translation Summit VIII*, pages 293–298, Santiago de Compostela, Spain.

Eric Ringger, Michael Gamon, Robert C. Moore, David Rojas, Martine Smets, and Simon Corston-Oliver. 2004. Lexically Informed Statistical Models of Constituent Structure for Ordering in Sentence Realization. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*, pages 673–680, Geneva.

Thomas Schack and Franz Mechsner. 2006. Representation of motor skills in human long-term memory. *Neuroscience Letters*, 391:77–81.

Thomas Schack. 2004. Knowledge and performance in action. *Journal of Knowledge Management*, 8(4):38–53.

Vincent Vandeghinste, Ineke Schuurman, Michael Carl, Stella Markantonatou, and Toni Badia. 2006. METIS-II: Machine Translation for Low Resource Languages. In *Proceedings of the 5th international conference on Language Resources and Evaluation (LREC)*, pages 24–26, Genoa, Italy.

Vincent Vandeghinste, Peter Dirix, and Ineke Schuurman. 2007. The effect of a few rules on a data-driven MT system. In *Proceedings of METIS workshop*, Leuven, Belgium.

Hideo Watanabe, Sadao Kurohashi, and Eiji Aramaki. 2003. Finding Translation Patterns from Dependency Structures. In Michael Carl and Andy Way, editors, *Recent Advances in Example-Based Machine Translation*, pages 397–412, Dordrecht, The Netherlands. Kluwer Academic Publishers.

Dekai Wu. 2006. MT Model Space: Statistical vs. Compositional vs. Example-Based Machine Translation. *Machine Translation*, 19.