Hierarchical Phrase-Based Translation with Jane 2

Matthias Huck, Jan-Thorsten Peter, Markus Freitag, Stephan Peitz, Hermann Ney

Machine Translation Marathon
Edinburgh, Scotland, UK – September 6, 2012

Human Language Technology and Pattern Recognition
Lehrstuhl für Informatik 6
Computer Science Department
RWTH Aachen University, Germany

Introduction

```
            ('-.         .-') _   ('-.
           ( OO ).-.    ( OO ) )_(  OO)
     ,--.  / . --. /,--./ ,--,'(,------.
 .-')| ,|  | \-.  \ |   \ |  |\ |  .---'
( OO |(_|.-'-'  |  ||    \|  | )|  |
| `-'|  | \| |_.'  ||  .     |/(|  '--.
,--. |  |  |  .-.  ||  |\    |  |  .--'
|  '-'  /  |  | |  ||  | \   |  |  `---.
 `_____'   `__' `__'`__'  `__'  `______'
```

- RWTH's open source statistical machine translation toolkit (free for non-commercial purposes)
- Implemented in C++
- See [Vilar et al., WMT 2010]
- Version 2.1 now available at `http://www.hltpr.rwth-aachen.de/jane`
- Note that Jane 2 also supports standard phrase-based translation

Outline

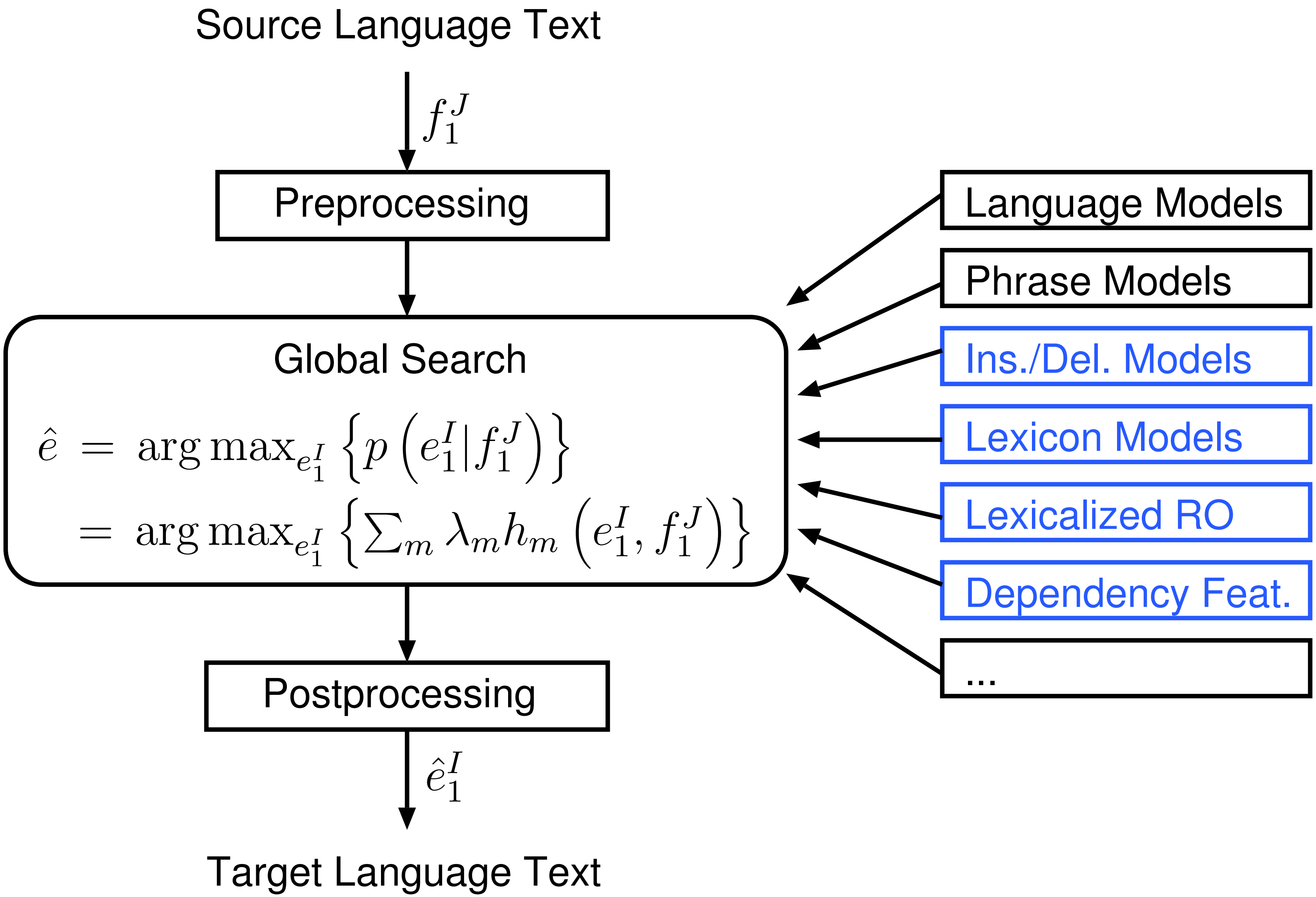
Extensions presented here:

- Insertion and deletion models **[Huck & Ney, NAACL 2012]**
- Lexical scoring variants **[Huck et al., IWSLT 2011]**
- Reordering extensions **[Huck et al., EAMT 2012]**
  - Non-lexicalized reordering rules
  - discriminative lexicalized reordering model
- Soft string-to-dependency hierarchical MT **[Peter et al., IWSLT 2011]**

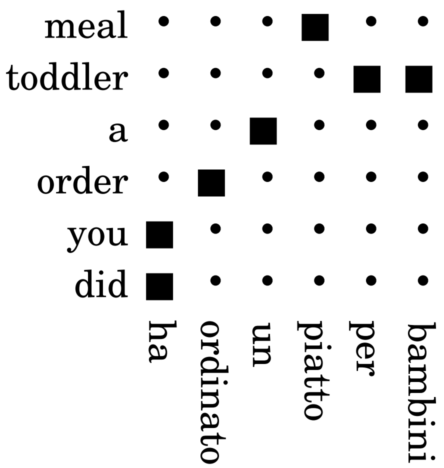
The Jane manual describes how to use the toolkit:

- `http://www.hltpr.rwth-aachen.de/jane/manual.pdf`

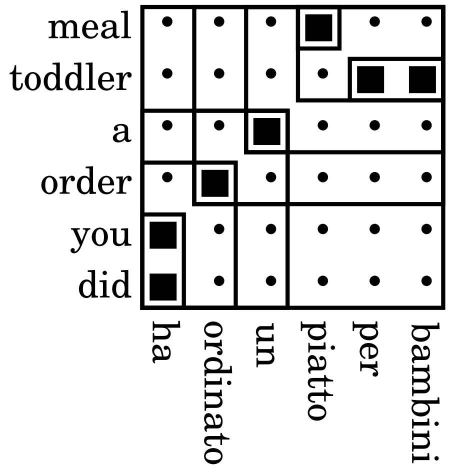
Statistical Machine Translation Architecture



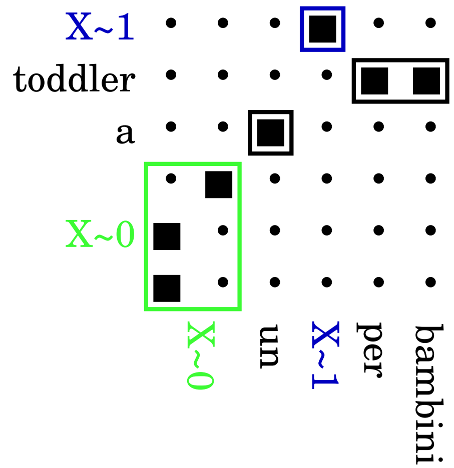
Hierarchical Phrase-Based Translation



Word alignment



Standard phrases



Hierarchical rule

- **Allow for gaps in the phrases**
- **Formalization as a synchronous context-free grammar**
- **Rules of the form $X \rightarrow \langle \alpha, \beta \rangle$, where:**
  - X **is a non-terminal**
  - α **and** β **are strings of terminals and non-terminals**
  - $\sim$ **is a one-to-one correspondence between the non-terminals of** α **and** β
- **Parsing-based decoding (extension of CYK algorithm)**
- **Cube pruning to handle translation alternatives**

Insertion and Deletion Models

- **Goal**: Penalize the omission of words in the hypothesis, or unjustified inclusion of words in it
- **Idea**: Introduce phrase-level feature functions which count the number of inserted or deleted words

Modeling of insertions and deletions relies on standard word lexicon models

- An English word is considered inserted or deleted based on lexical probabilities with the words on the foreign language side of the phrase
- **Thresholding** of lexical probabilities
  - from ***different types of word lexicon models***
  - with ***different types of thresholding methods***

Insertion model scoring will be presented on the next two slides.
See our paper to learn about deletion models and thresholding methods.

Insertion Modeling (Source-to-Target)

- An occurrence of a target word e is considered an insertion *iff* no source word f exists within the phrase with $p(e|f)$ greater than or equal to τ_f
- The feature function counts the number of inserted words on the target side β with respect to the source side α

Insertion model feature function in source-to-target direction:

$$t_{\text{s2tIns}}(\alpha, \beta) = \sum_{i=1}^{I_\beta} \prod_{j=1}^{J_\alpha} \left[p(\beta_i | \alpha_j) < \tau_{\alpha_j} \right] \tag{1}$$

- J_α is defined as the number of terminal symbols in α
- I_β is defined as the number of terminal symbols in β
- α_j, $1 \leq j \leq J_\alpha$, denotes the j-th terminal symbol on the source side
- β_i, $1 \leq i \leq I_\beta$, denotes the i-th terminal symbol on the target side
- $[\cdot]$ denotes a true or false statement (1 if the condition is true, 0 otherwise)

Insertion Modeling (Target-to-Source)

- An occurrence of a source word f is considered an insertion *iff* no target word e exists within the phrase with $p(f|e)$ greater than or equal to τ_e
- The feature function counts the number of inserted words on the source side α with respect to the target side β

Insertion model feature function in target-to-source direction:

$$t_{\mathrm{t2sIns}}(\alpha, \beta) = \sum_{j=1}^{J_\alpha} \prod_{i=1}^{I_\beta} \left[p(\alpha_j|\beta_i) < \tau_{\beta_i} \right] \tag{2}$$

- J_α is defined as the number of terminal symbols in α
- I_β is defined as the number of terminal symbols in β
- α_j, $1 \leq j \leq J_\alpha$, denotes the j-th terminal symbol on the source side
- β_i, $1 \leq i \leq I_\beta$, denotes the i-th terminal symbol on the target side
- $[\cdot]$ denotes a true or false statement (1 if the condition is true, 0 otherwise)

Lexical Scoring Variants

- Problem: Overestimation of phrase translation probabilities of phrase pairs for which little evidence in the training data exists
- Typical solution in state-of-the-art systems:
  Interpolate phrase translation scores with lexical scores

Jane 2 implements four different methods to score phrase pairs,
for use either with

- a lexicon model which is extracted from word-aligned training data (RF word lexicon), or
- the IBM Model 1 lexicon

See our paper for the lexical scoring formulas,
and the Jane manual for usage instructions.

Reordering Extensions: Motivation

**In hierarchical phrase-based machine translation,
reordering is modeled implicitely as part of the phrase translation model**

- **Typically no additional mechanism to perform reorderings that do not result from the application of hierarchical rules**
- **No integration of lexicalized reordering models (e.g. in the manner of phrase orientation models of standard phrase-based systems)**

Example for Reordering Rules: Swap Rule

Standard initial rule and glue rule:

$$\boldsymbol{S \rightarrow \left\langle X^{\sim 0}, X^{\sim 0} \right\rangle}$$
$$\boldsymbol{S \rightarrow \left\langle S^{\sim 0} X^{\sim 1}, S^{\sim 0} X^{\sim 1} \right\rangle}$$

Bring in more reordering capabilities by adding a single swap rule:

$$\boldsymbol{X \rightarrow \left\langle X^{\sim 0} X^{\sim 1}, X^{\sim 1} X^{\sim 0} \right\rangle}$$

- **The swap rule allows adjacent phrases to be transposed**

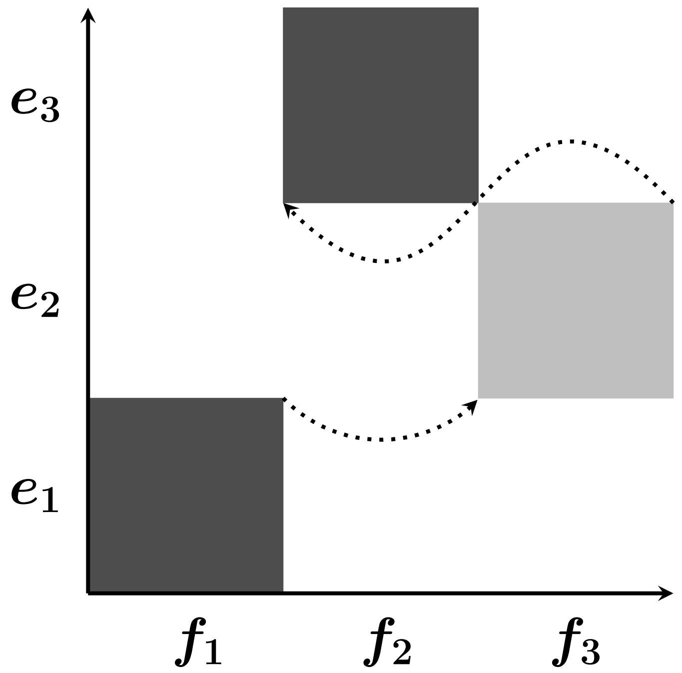
Discriminative Reordering Model (1)

Integrate a discriminatively trained lexicalized reordering model (discrim. RO) that predicts the orientation of neighboring blocks

- **Two orientation classes *left* and *right***
- **Orientation probability is modeled in a *maximum entropy framework***
- **Different feature sets possible**
- **Training with Generalized Iterative Scaling (GIS)**

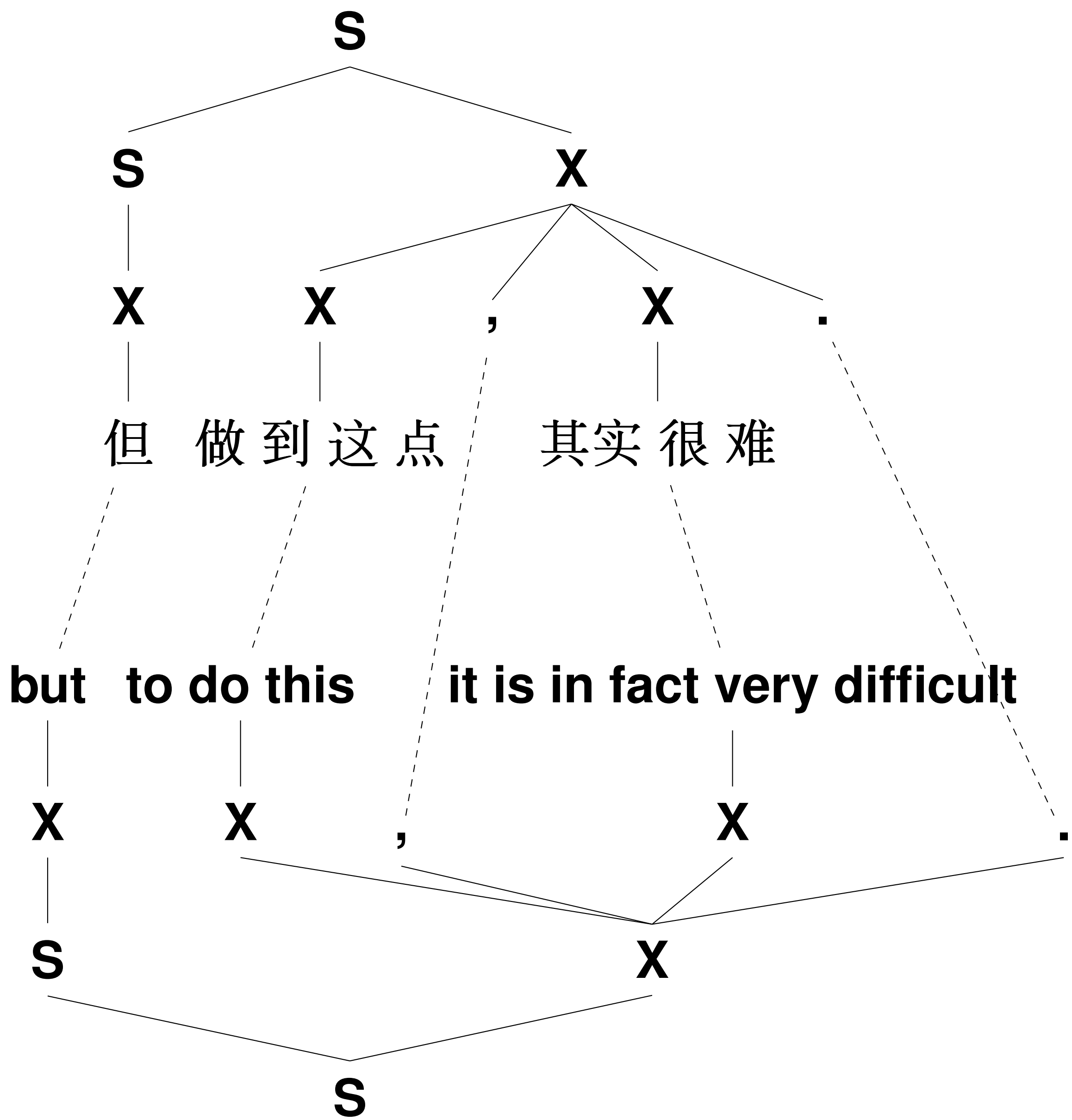
Discriminative Reordering Model (2)

- **The discriminative reordering model is applied at *block boundaries*, where words which are adjacent to gaps within hierarchical phrases are defined as boundary words as well**
- **Example: an embedding of a lexical phrase (light) in a hierarchical phrase (dark), with orientations scored with the neighboring blocks:**

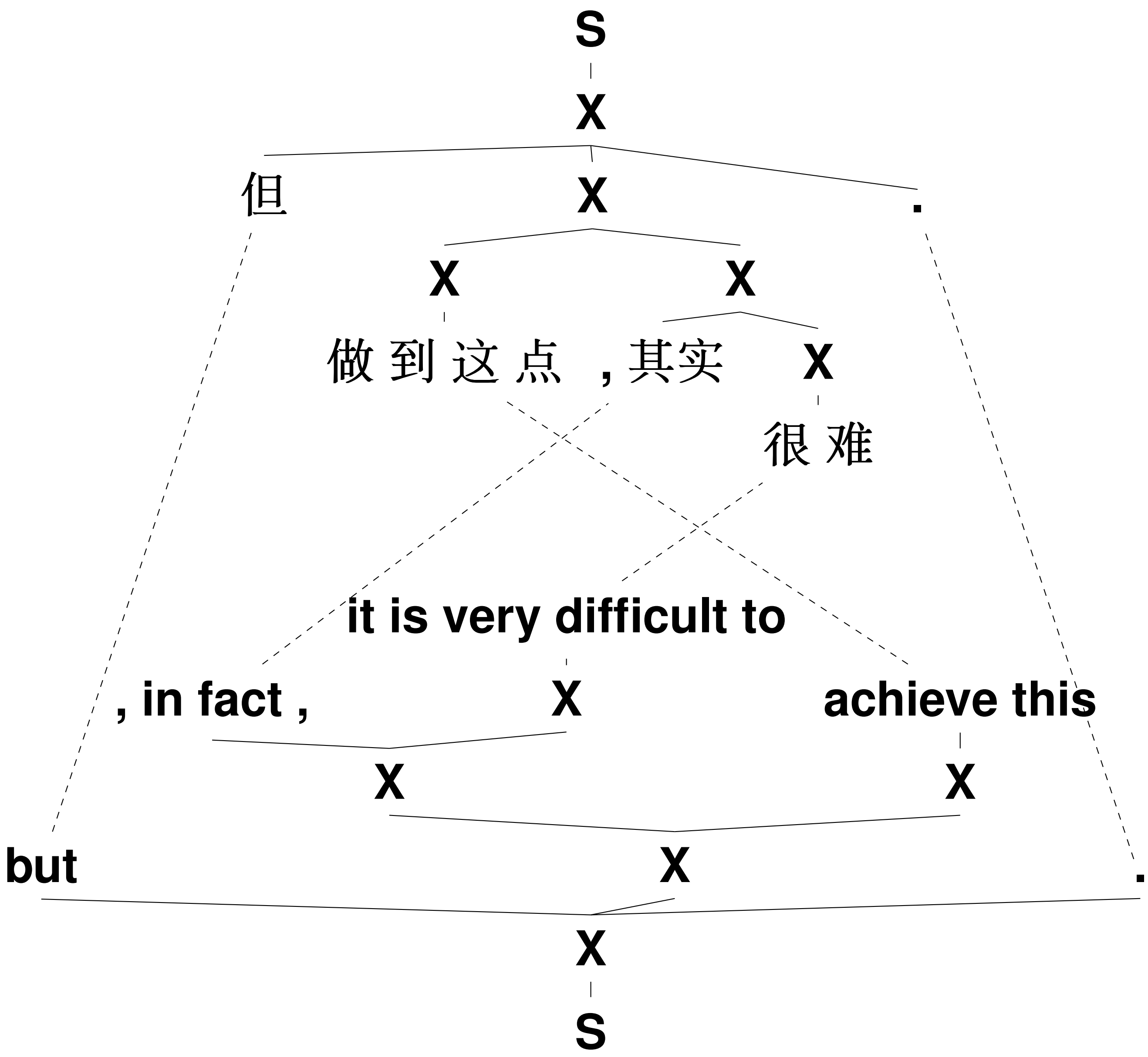


The gap in the hierarchical phrase $\langle f_1 f_2 X^{\sim 0}, e_1 X^{\sim 0} e_3 \rangle$ is filled with the lexical phrase $\langle f_3, e_2 \rangle$.

Translation Example: Baseline



Translation Example: Swap Rule and Discrim. RO



Soft String-to-Dependency Hierarchical MT

Training

- Use a parser to create dependencies on the target side of the training corpora
- Phrase extraction: Store dependency information for each phrase
- Train dependency language model

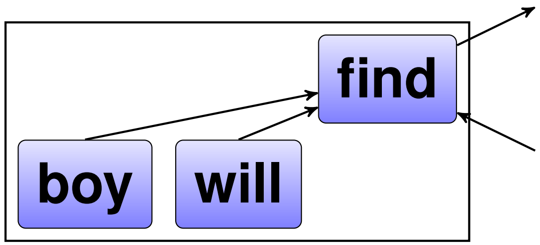
Decoding

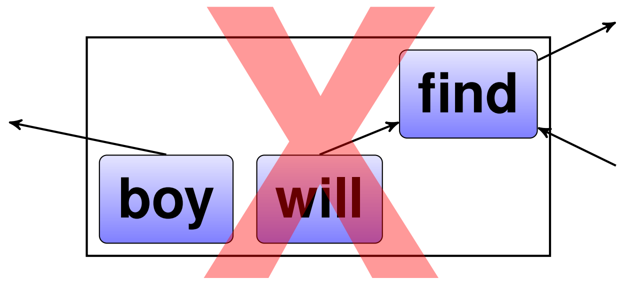
- Assemble dependencies
- Penalize phrases with invalid dependency structures
- Penalize tree merging errors
- Apply dependency language model

Phrases with Invalid Dependency Structures

Valid fixed on head structure (left) and a counterexample (right)

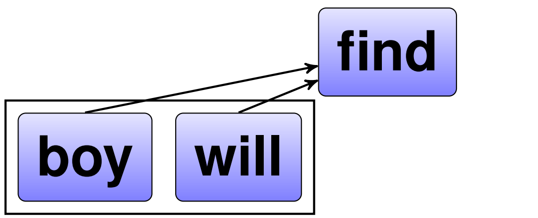
- Only head node allowed to have dependency outside the structure

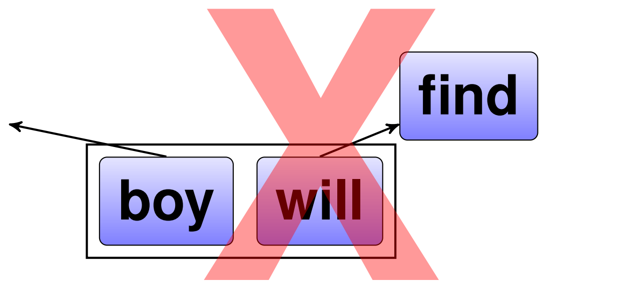




Valid floating with children structure (left) and a counterexample (right)

- All dependencies point to one head outside the structure



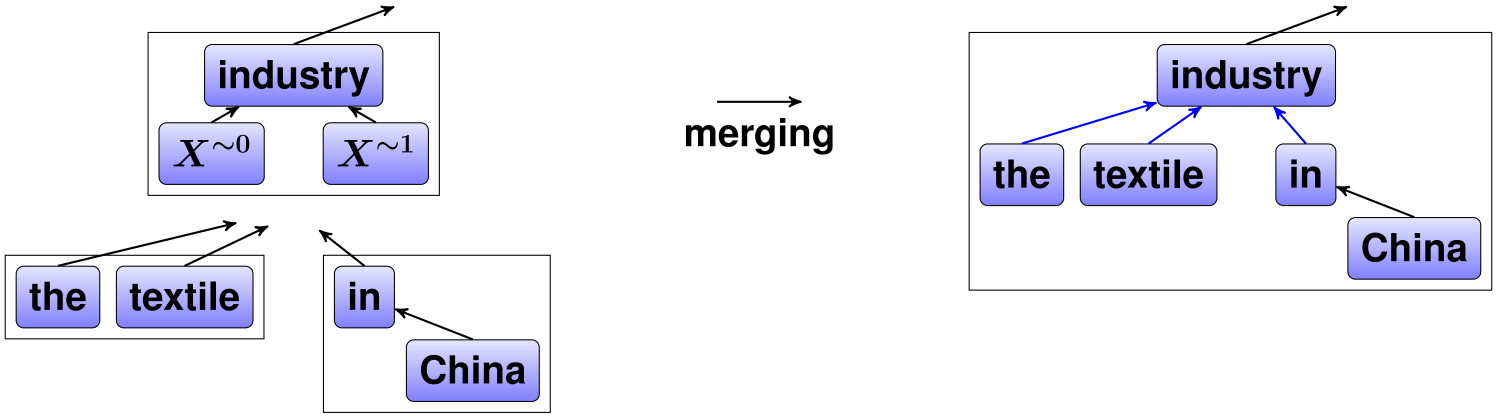


Here: No restrictions on phrase inventory, but penalty during decoding

Merging Dependency Tree Fragments

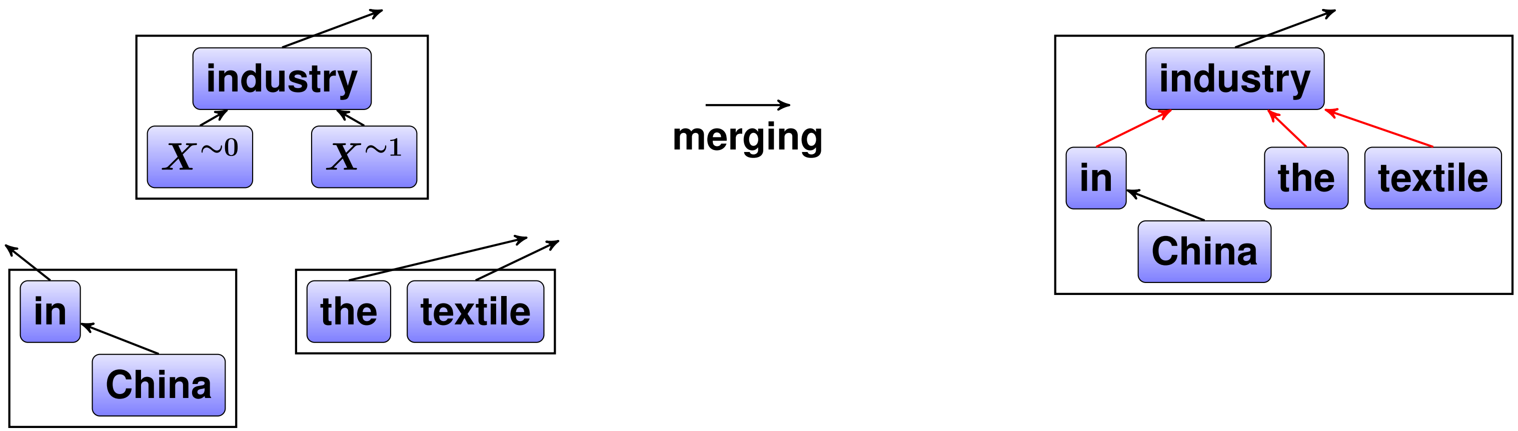
Merging tree fragments without merging errors

- All dependency pointers point into the same directions as the parent dependencies

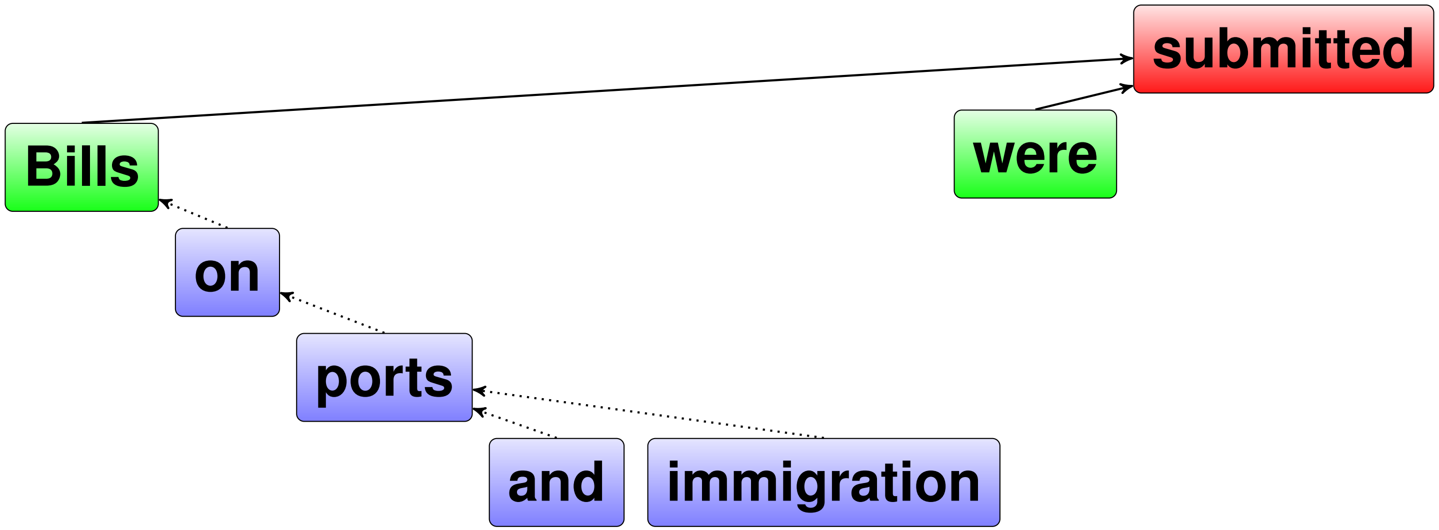


Merging tree fragments with one left and two right merging errors

- The dependency pointers point into other directions as the parent dependencies
- Merging errors are penalized



Dependency Language Model



$$p(e_1^I|T) = p_h(\textbf{submitted}) \times$$
$$p_l(\textbf{were|submitted-as-head}) \times p_l(\textbf{bills|were, submitted-as-head}) \times$$
$$p_r(\textbf{on|bills-as-head}) \times \dots$$

Three n-gram models:

- p_h: **1-gram model for word being head of structure**
- p_l: **n-gram model for dependency level left of head**
- p_r: **n-gram model for dependency level right of head**

Some Experimental Results

NIST Chinese→English translation task

- 3.0M sentences of parallel training data (77.5M Chinese / 81.0M English running words)
- 4-gram LM

	MT06 (Dev)		MT08 (Test)	
	BLEU [%]	TER [%]	BLEU [%]	TER [%]
Baseline (with RF word lexicons)	32.6	61.2	25.2	66.6
+ insertion model	32.9	61.4	25.7	66.2
+ deletion model	32.9	61.4	26.0	66.1
+ IBM Model 1	33.8	60.5	26.9	65.4
+ discrim. RO	33.0	61.3	25.8	66.0
+ swap rule + binary swap feature	33.2	61.3	26.2	66.1
Soft string-to-dependency	33.5	60.8	26.0	65.7
— only valid phrases	32.8	62.0	25.4	67.1
— no merging errors	32.5	61.5	25.5	66.4
+ insertion model + discrim. RO + DWL + triplets	35.0	59.5	27.8	64.4

Summary

```
            ('-.            .-') _    ('-.
           ( OO ).-.       ( OO ) )_(  OO)
     ,--.  / . --. /,--./ ,--,'(,------.
 .-')| ,|  | \-.  \ |   \ |  |\ |  .---'
( OO |(_|.-'-'  |  ||    \|  | )|  |
| `-'|  | \| |_.'  ||  .     |/(|  '--.
,--. |  |  |  .-.  ||  |\    |  |  .--'
|  '-'  /  |  | |  ||  | \   |  |  `---.
 `-----'   `--' `--'`--'  `--'  `------'
```

- **Efficient toolkit for phrase-based and hierarchical phrase-based translation**
- **Open source, free for non-commercial use**
- `http://www.hltpr.rwth-aachen.de/jane`
- **Extensions to HPBT in Jane 2:**
  - **Insertion and deletion models**
  - **Lexical scoring variants**
  - **Lexicalized reordering**
  - **Soft string-to-dependency hierarchical MT**

Thank you for your attention!

Previous papers about Jane:

D. Vilar, D. Stein, M. Huck, and H. Ney. Jane: Open Source Hierarchical Translation, Extended with Reordering and Lexicon Models. In ACL 2010 Joint Fifth Workshop on Statistical Machine Translation and Metrics MATR, Uppsala, Sweden, July 2010.

D. Stein, D. Vilar, S. Peitz, M. Freitag, M. Huck, and H. Ney. A Guide to Jane, an Open Source Hierarchical Translation Toolkit. In The Prague Bulletin of Mathematical Linguistics, Prague, Czech Republic, April 2011.

D. Vilar, D. Stein, M. Huck, and H. Ney. Jane: an advanced freely available hierarchical machine translation toolkit. In Machine Translation (Online First), January 2012. http://dx.doi.org/10.1007/s10590-011-9120-y.

Thank you for your attention

Matthias Huck

huck@cs.rwth-aachen.de

http://www.hltpr.rwth-aachen.de/