IWSLT – Trento, Italy – October 2007

Improved Chunk-level Reordering for Statistical Machine Translation

Yuqi Zhang, Richard Zens and Hermann Ney

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

Overview

- Introduction
- Phrase-based SMT
- Chunk-level reordering system
- Improvements
  - Reordering training data
  - Reordering-lattice weighting
- Results
- Conclusion and Outlook

Introduction

goal:

improve MT utilizing syntactic knowledge

idea:

reordering at the chunk level

approach:

1. **chunk source sentence**
2. **reorder chunks**
3. **represent alternative reorderings in a lattice**
4. **translate lattice**

Phrase-based SMT

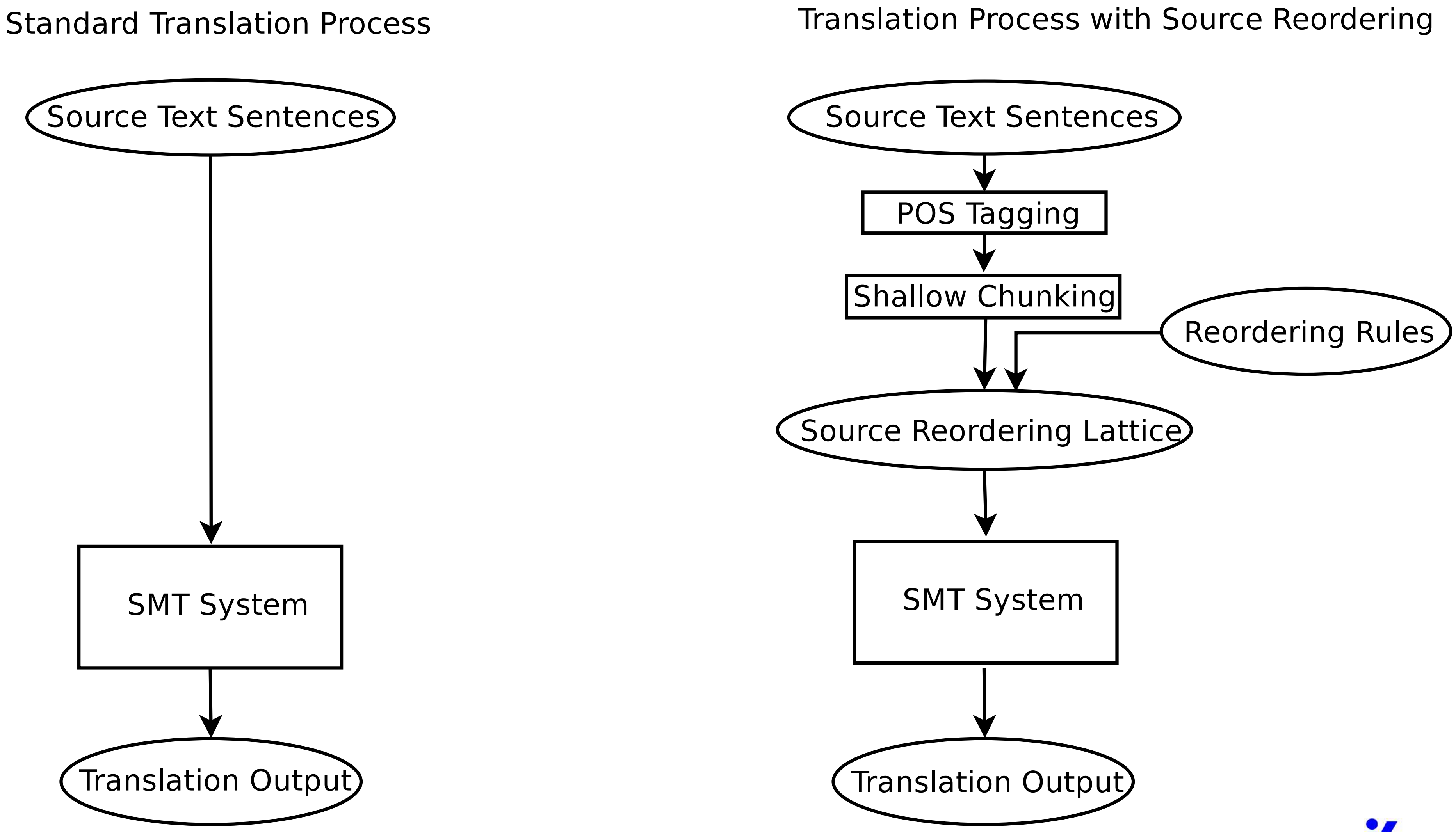
log-linear model:

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

models:

- phrase translation model
- phrase count features
- word-based translation model
- word and phrase penalty
- target language model (6-gram)
- distortion model

System Structure

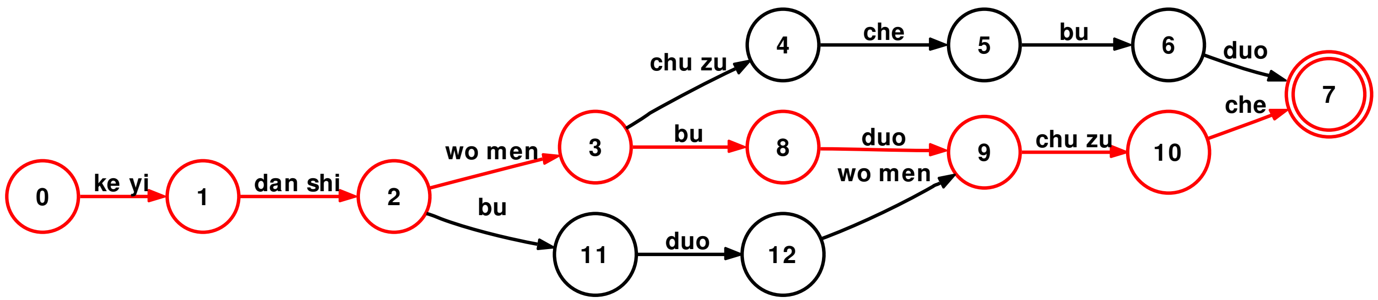


An Example

source	可以	但是	我们	出租	车	不	多
pin yin	ke yi	dan shi	wo men	chu zu	che	bu	duo
POS	v	c	r	v	n	d	m
chunks	v	c	r	NP		VP	
English gloss	yes	but	we	taxi		not many	

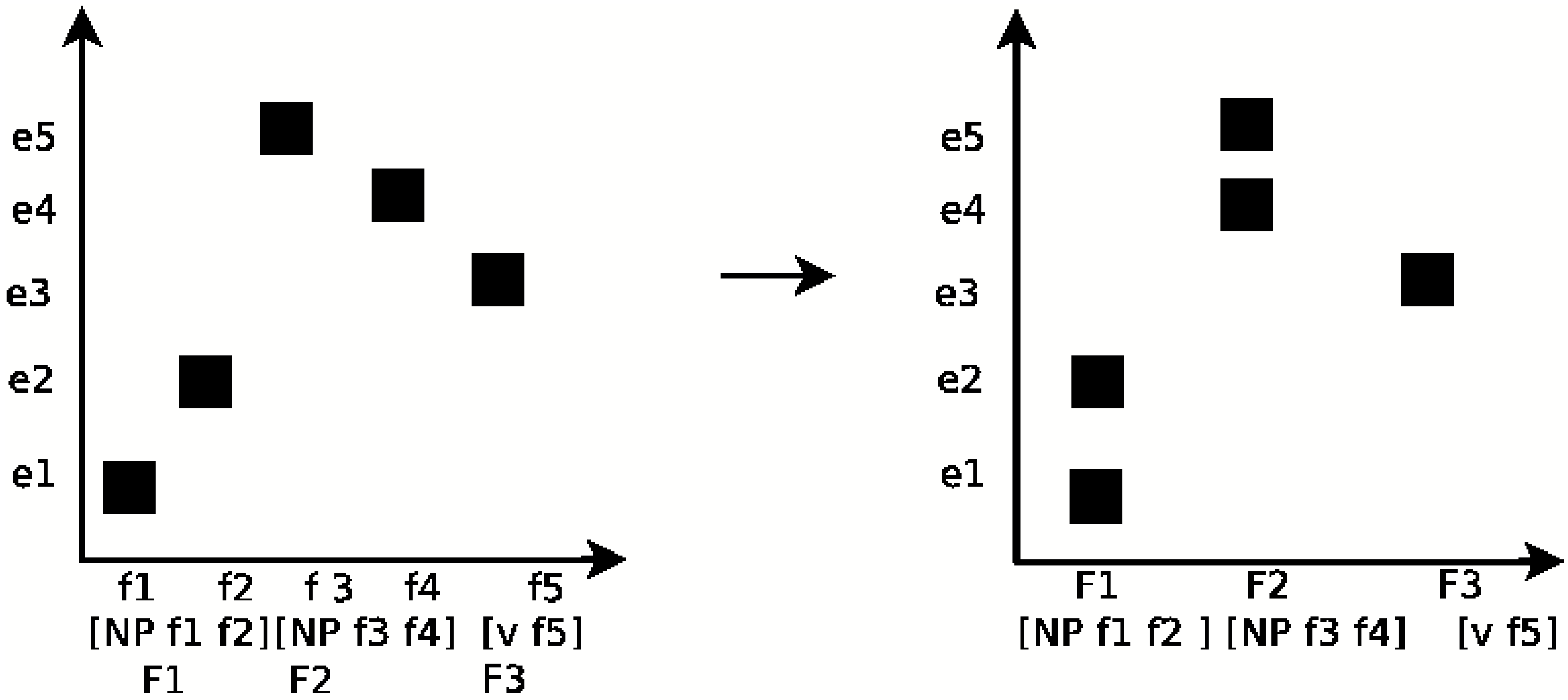
used reordering rules
NP VP $\longrightarrow$ VP NP
r NP VP $\longrightarrow$ r VP NP
r NP VP $\longrightarrow$ VP r NP

Reordering Lattice:



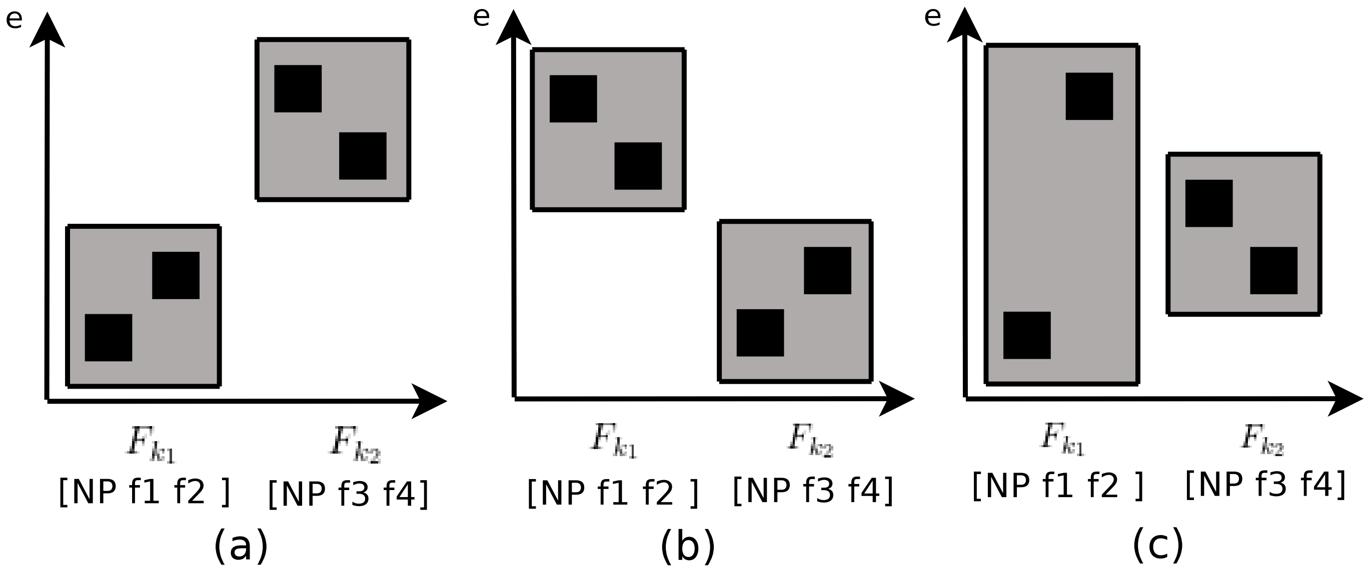
Reordering Rules Extraction

- convert word-to-word alignment to chunk-to-word alignment



- run standard phrase extraction on chunk-to-word alignment

Reordering Rules Extraction (cont'd)



(a) monotone phrase, (b) reordering phrase, (c) cross phrase

- **extract rules from monotone phrases and reordering phrases**
  - **e.g.** $NP_0 NP_1 \# \ NP_0 \ NP_1 \quad NP_0 NP_1 \# \ NP_1 \ NP_0$

Reordering Lattice Generation I

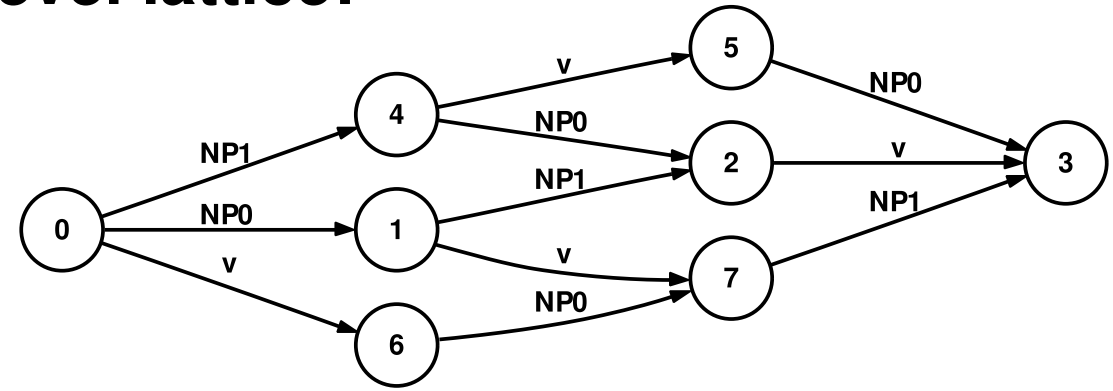
- apply reordering rules to chunked source sentence
- represent alternative reorderings as a lattice
- example:

Sentence Permutations

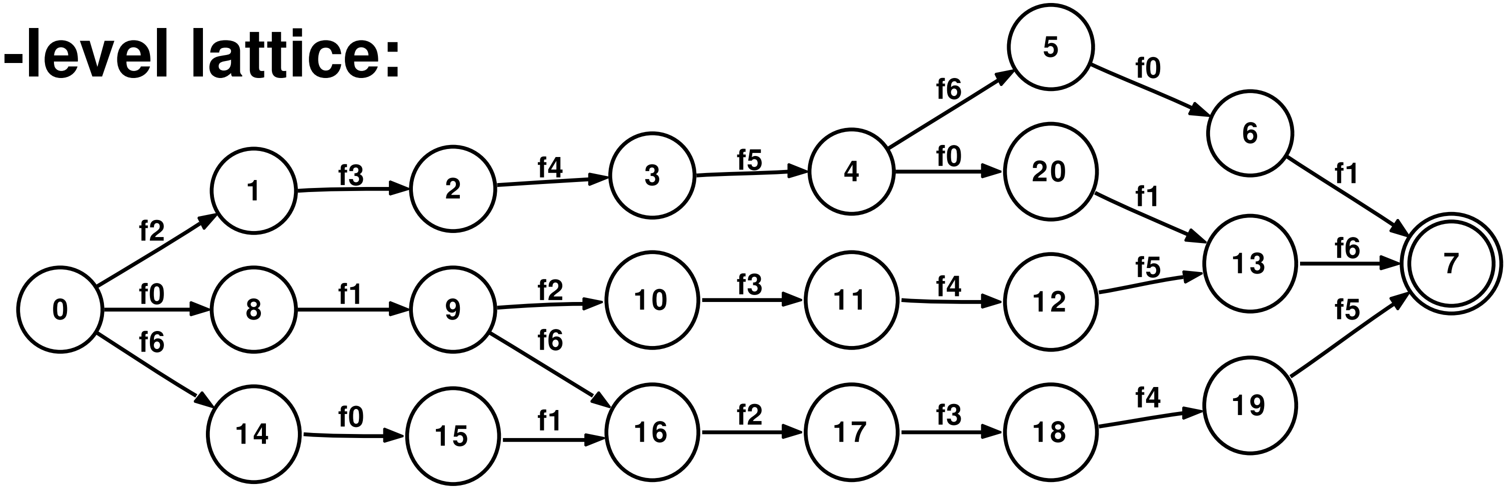
0	1	2	3	4	5	6
2	3	4	5	0	1	6
0	1	2	3	4	5	6
0	1	6	2	3	4	5
0	1	2	3	4	5	6
2	3	4	5	6	0	1
6	0	1	2	3	4	5

Reordering Lattice Generation II

- **chunk-level lattice:**



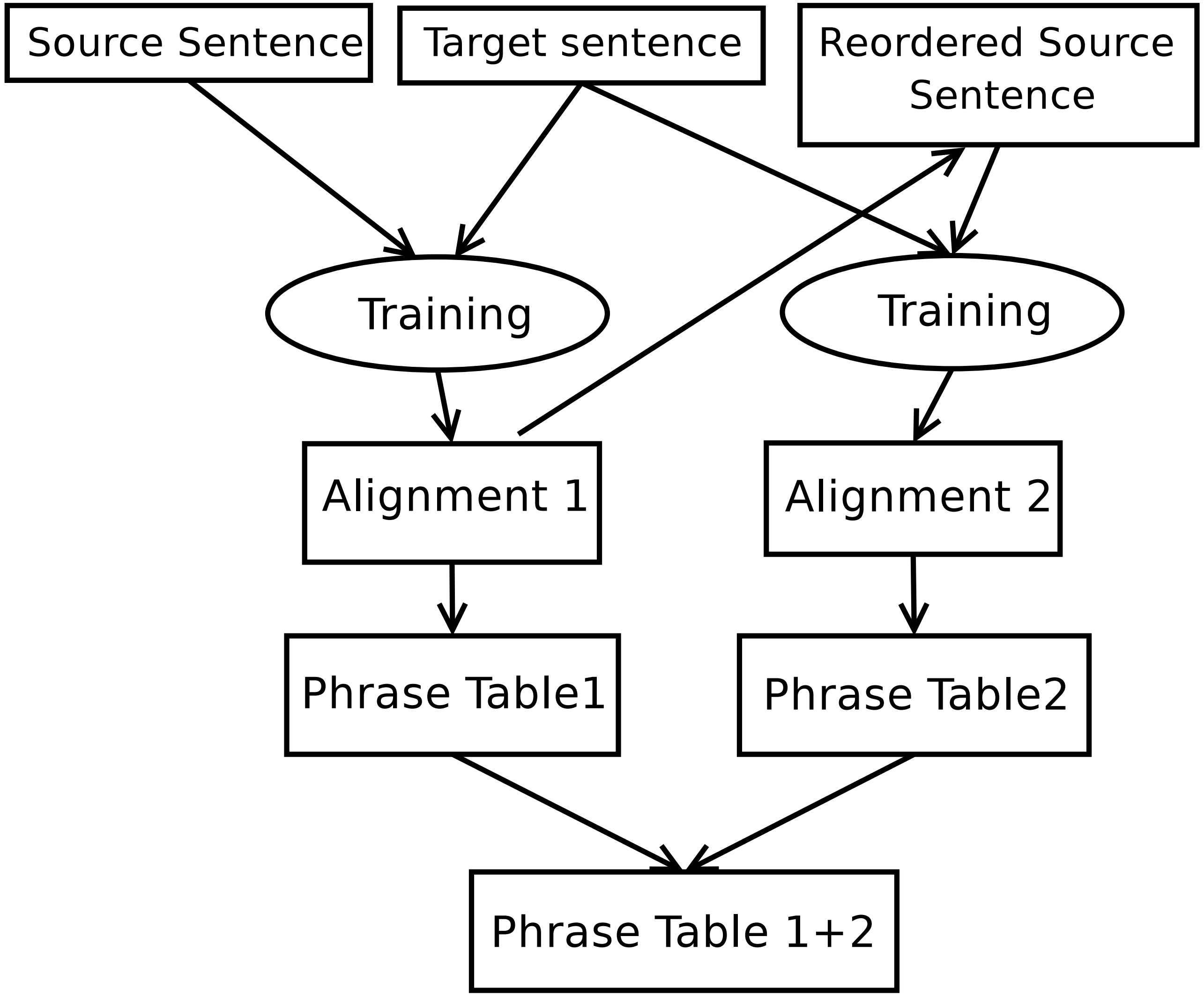
- **word-level lattice:**



Training Data Reordering I

- chunk source training data
- generate chunk-to-word alignment
- reorder source chunks to monotonize alignments.
- train LM on reordered source training data
- extract phrases on reordered training data

Training Data Reordering II



Lattice Weighting

- **For each path in the lattice, the weight is computed by the two models**
  - **reordered source language model h_{slm}**
  - **reordering rules probability model h_{reorder}**

Lattice Weighting: h_{slm}

- **Each path of the lattice is a permutation $f_{\pi_1}^{\pi_J} = f_{\pi_1}, ..., f_{\pi_J}$ for a given source sentence f_1^J**

$$h_{\text{slm}}(f_{\pi_1}^{\pi_J}, f_1^J) = \log p(f_{\pi_1}^{\pi_J} | f_1^J)$$

π_j is the permutation position of word f_j

- **Word trigram language model**

$$\log p(f_{\pi_1}^{\pi_J} | f_1^J) = \sum_{j=1}^{J} \log p(f_{\pi_j} | f_{\pi_{j-1}}, f_{\pi_{j-2}})$$

Lattice Weighting: h_{reorder}

$$h_{\text{reorder}}(\pi_1^N, c_1^N) = \log(p(\pi_1^N | c_1^N))$$

c_1^N: **sequence of chunks,** $f_1^J = c_1^N$

π_n: **permutation position of chunk** c_n.

$$p(\pi_1^N | c_1^N) = \sum_B \alpha(c_1^N) \cdot p(\pi_1^N | c_1^N, B)$$

$$p(\pi_1^N | c_1^N, B) = p(\tilde{\pi}_1^K | \tilde{c}_1^K) = \prod_{k=1}^K p(\tilde{\pi}_k | \tilde{c}_k) = \prod_{k=1}^K \frac{N(\tilde{\pi}_k, \tilde{c}_k)}{N(\tilde{c}_k)}$$

B: **segmentation**

$\tilde{c}_k$: **left-hand side of** r_k

$\tilde{\pi}_k$: **right-hand side** r_k

Corpus Statistics

		Chinese	English
Train	Sentences	43 k	
	Words	380 k	420 k
	Vocabulary	11 760	9 933
Dev dev2	Sentences	500	
	Words	3 578	3 908
	OOVs	73	–
Test dev3	Sentences	506	
	Words	3 837	3 970
	OOVs	70	–

- optimize on BLEU score.

Translation Result I

Translation performance for the Chinese-English IWSLT05 task

test(dev3)	WER[%]	PER[%]	TER[%]	BLEU[%]
baseline: chunk reorder	33.5	27.2	32.0	59.0
+ ruleProb	33.1	27.0	32.0	59.7
+ reordered train data	32.7	27.8	31.5	60.3

- baseline: reordering lattice is weighted by source language model.

Comarison with the RWTH best system

	BLEU[%]
monotone	56.0
RWTH best system	62.4
source reorder improved	60.3

Translation Examples

source	有很多鱼的地方在哪?
chunks	有_v [NP 很多_m 鱼_n] 的_u 地方_n 在_p [NP 哪_r] ?_w
reference	**What place has a lot of fish?**
chunk reorder	**Where can i find a lot of fish?**
RWTH-best-system	**there are many fish Where?**
source	我想要一个面向海滩的房间.
chunks	我_r 想_v 要_v 一个_m [VP 面向_v 海滩_n] 的_u 房间_n ._w
reference	**I'd like a room facing the beach.**
chunk reorder	**I would like a room facing the beach.**
RWTH-best-system	**I would like a beach facing the room.**

Summary

- idea:
  1. chunk input sentence
  2. reorder chunks
  3. represent alternative reorderings as lattice
  4. translate lattice
- improve system
  1. reorder training data
  2. rule probability model to lattice

Outlook

- large data task (e.g. NIST)
- other language pairs
- improve chunk parsing
- analyze what kind of rules work well

THANK YOU FOR YOUR ATTENTION!

Chunk Parsing

- POS tagging + word segmentation with ICTCLAS tool
  (from Institute of Computing Technology,Chinese Academy of Sciences)

- Learn chunks from Chinese Treebank (LDC2005T01) with the constraints:
  - a subtree has more than one child,
  - the children of a subtree are all leaves.

- Tag each source word to mark what chunk it belongs to and its position within a chunk with Maximum Entropy Tagging (YASMET tool)
  - input features: word + POS tag
  - output: chunk types + chunk boundary

Chunking Result

Statistics of training and test corpus for chunk parsing (from Chinese Treebank LDC2005T01)

	train	test
sentences	17 785	1 000
words	486 468	21 851
chunks	105 773	4 680
words out of chunks	244 416	10 282

Result of tagging: found: 4414 chunks; correct:2879

accuracy(%)	precision(%)	recall(%)	F-measure
74.51	65.2	61.5	63.3

- The number of chunk types: 24
- a chunk is correct when both chunk type and boundary are correct
- precision and recall are at chunk level
- accuracy: correct tags at word level

Rules Statistics

	rules	singletons(%)	reorder rules	used rules
pos rules	327k	287k (88%)	118k (36%)	49k
chunk rules	184k	162k (88%)	63k (34%)	25k

Detail Statistics of Chunk Rules