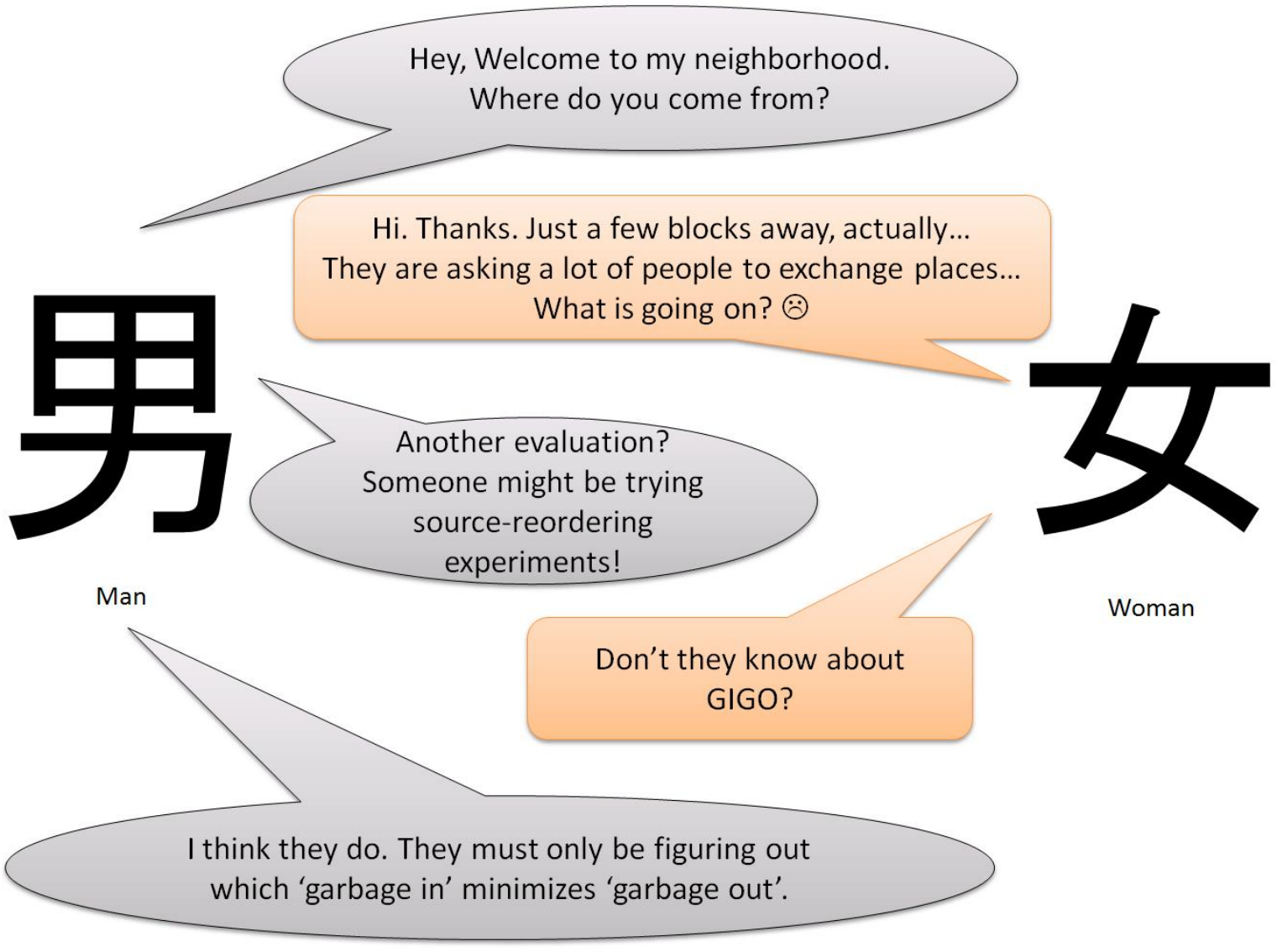
Reordering Problem and Solutions The Word

Alok Parlikar

Language Technologies Institute
Carnegie Mellon University

March 19, 2008 / Advanced MT Seminar

What is going on?



Ok Ok..


Aaah! OK!
I will try to speak normal English now.
I wish I were Shakespeare....

Outline

1. Introduction
   - Reordering in Phrase-based SMT
   - Word Orders between Language Pairs
2. POS-based Reordering
   - Popović, Ney (2006)
   - Crego, Mariño (2006)
   - Rottmann, Vogel (2007)
3. Syntactic Reordering
   - Xia, McCord (2004)
   - Collins, Koehn, Kučerová (2006)
   - Nguyen, Shimazu (2006)
   - Li et al. (2007)
4. Conclusion

Outline

1. Introduction
   - Reordering in Phrase-based SMT
   - Word Orders between Language Pairs
2. POS-based Reordering
   - Popović, Ney (2006)
   - Crego, Mariño (2006)
   - Rottmann, Vogel (2007)
3. Syntactic Reordering
   - Xia, McCord (2004)
   - Collins, Koehn, Kučerová (2006)
   - Nguyen, Shimazu (2006)
   - Li et al. (2007)
4. Conclusion

Reordering in Phrase-based SMT

- Phrase based SMT models has achieved the state-of the art perfromance. These models have several advantages, such as word choice, idomatic expression recognition, and local restructuring.
- There still are potential limitations when it comes to modelling word-order differences between languages.
- Use of reordering allows for important improvement in translation accuracy
- Arbitrary word reorderings could be permitted...
- Typically used reordering: Distance based reordering
- Are somehow 'non-linguistic'

Introducing Linguistic Information

Source Sentence Reordering

Transform the source sentence so that the order of words conforms to that in the target language.

- How to model the word reordering from source to target?
- How to score different reorderings?
- How to apply the model at run-time?

Introducing Linguistic Information

Source Sentence Reordering

Transform the source sentence so that the order of words conforms to that in the target language.

- How to model the word reordering from source to target?
- How to score different reorderings?
- How to apply the model at run-time?

Introducing Linguistic Information

Source Sentence Reordering

Transform the source sentence so that the order of words conforms to that in the target language.

- How to model the word reordering from source to target?
- How to score different reorderings?
- How to apply the model at run-time?

Introducing Linguistic Information

Source Sentence Reordering

Transform the source sentence so that the order of words conforms to that in the target language.

- How to model the word reordering from source to target?
- How to score different reorderings?
- How to apply the model at run-time?

Outline

1. Introduction
   - Reordering in Phrase-based SMT
   - Word Orders between Language Pairs
2. POS-based Reordering
   - Popović, Ney (2006)
   - Crego, Mariño (2006)
   - Rottmann, Vogel (2007)
3. Syntactic Reordering
   - Xia, McCord (2004)
   - Collins, Koehn, Kučerová (2006)
   - Nguyen, Shimazu (2006)
   - Li et al. (2007)
4. Conclusion

Reordering phenomena between languages

- Most papers report results on European languages
- French, Spanish, German, English
- Also: Asian languages (Vietnamese)

French, Spanish $\leftrightarrow$ English

- Local Reorderings
- Most adjectives come after the noun in French, Spanish. In English, Adjectives come before the nouns.
- N ADJ $\overset{?}{\leftrightarrow}$ ADJ N

Example

- French: train *rouge*
- English: *red* train

German ↔ English

- Global Reorderings
- Infinitives and Past Participles are placed at the end of a clause in German. In English, they usually occur towards the beginning of the clause.
- Detached verb prefixes also go to the end of the clause.

Example

- German: Ich *werde* morgen nachmittag . . . *ankommen*
- English: I *will arrive* tomorrow afternoon . . .

Vietnamese ↔ English

- SVO word order, similar to English
- WH-movement is significantly different. (The interrogative word is not moved to the beginning of the sentence).
- Most yes-no questions end in an interrogative word.
- Most phrases are head-final.

Example

- Vietnamese: BOOK 's FRIEND HIS
- English: his friend's book

Outline

1. Introduction
   - Reordering in Phrase-based SMT
   - Word Orders between Language Pairs
2. POS-based Reordering
   - Popović, Ney (2006)
   - Crego, Mariño (2006)
   - Rottmann, Vogel (2007)
3. Syntactic Reordering
   - Xia, McCord (2004)
   - Collins, Koehn, Kučerová (2006)
   - Nguyen, Shimazu (2006)
   - Li et al. (2007)
4. Conclusion

POS-based Word Reordering Models for SMT

Popović, Ney (2006)

- Languages: English, Spanish, German
- Based entirely on POS. Additional syntactic tools (parsers) not requried.
- Limited range of reordering phenomena:
  1. Adjective-Noun reordering in Spanish
     - From Spanish to English/German: Move adjective before noun group
     - From English/German to Spanish: Move adjective after noun group
  2. Verb reordering in German
     - From Spanish/English to German: Move infitive or past participle to end of the clause. Keep auxiliary verb in original position.

Experimental Setup

Popović, Ney (2006)

- Europarl Corpus: 700K sentences
- POS Taggers: FreeLing, ENGCG, GERCG
- Trilingual Corpus: 670K sentences
- Studied effect of data sparsity, and training-corpus reordering.
- RWTH SMT System used for decoding.

		Spanish	English	German	English
Train:	Sentences	730740		751088	
	Running Words+Punctuation	15724914	15222146	15257678	16052330
	Vocabulary	113882	72739	205374	74708
	Singletons [%]	39.2	38.3	49.8	38.3
Dev:	Sentences	2000		2000	
	Running Words+Punctuation	60628	58655	55147	58655
	Distinct Words	8182	6547	9213	6547
	OOVs [%]	0.4	0.2	0.8	0.2
Test:	Sentences	2000		2000	
	Running Words+Punctuation	60332	57951	54260	57951
	Distinct Words	8279	6496	9048	6496
	OOVs [%]	0.4	0.2	0.7	0.2

Experimental Setup

Popović, Ney (2006)

- Europarl Corpus: 700K sentences
- POS Taggers: FreeLing, ENGCG, GERCG
- Trilingual Corpus: 670K sentences
- Studied effect of data sparsity, and training-corpus reordering.
- RWTH SMT System used for decoding.

		Spanish	English	German	English
Train:	Sentences	730740		751088	
	Running Words+Punctuation	15724914	15222146	15257678	16052330
	Vocabulary	113882	72739	205374	74708
	Singletons [%]	39.2	38.3	49.8	38.3
Dev:	Sentences	2000		2000	
	Running Words+Punctuation	60628	58655	55147	58655
	Distinct Words	8182	6547	9213	6547
	OOVs [%]	0.4	0.2	0.8	0.2
Test:	Sentences	2000		2000	
	Running Words+Punctuation	60332	57951	54260	57951
	Distinct Words	8279	6496	9048	6496
	OOVs [%]	0.4	0.2	0.7	0.2

Experimental Setup

Popović, Ney (2006)

- Europarl Corpus: 700K sentences
- POS Taggers: FreeLing, ENGCG, GERCG
- Trilingual Corpus: 670K sentences
- Studied effect of data sparsity, and training-corpus reordering.
- RWTH SMT System used for decoding.

		Spanish	English	German	English
Train:	Sentences	730740		751088	
	Running Words+Punctuation	15724914	15222146	15257678	16052330
	Vocabulary	113882	72739	205374	74708
	Singletons [%]	39.2	38.3	49.8	38.3
Dev:	Sentences	2000		2000	
	Running Words+Punctuation	60628	58655	55147	58655
	Distinct Words	8182	6547	9213	6547
	OOVs [%]	0.4	0.2	0.8	0.2
Test:	Sentences	2000		2000	
	Running Words+Punctuation	60332	57951	54260	57951
	Distinct Words	8279	6496	9048	6496
	OOVs [%]	0.4	0.2	0.7	0.2

Experimental Setup

Popović, Ney (2006)

- Europarl Corpus: 700K sentences
- POS Taggers: FreeLing, ENGCG, GERCG
- Trilingual Corpus: 670K sentences
- Studied effect of data sparsity, and training-corpus reordering.
- RWTH SMT System used for decoding.

		Spanish	English	German	English
Train:	Sentences	730740		751088	
	Running Words+Punctuation	15724914	15222146	15257678	16052330
	Vocabulary	113882	72739	205374	74708
	Singletons [%]	39.2	38.3	49.8	38.3
Dev:	Sentences	2000		2000	
	Running Words+Punctuation	60628	58655	55147	58655
	Distinct Words	8182	6547	9213	6547
	OOVs [%]	0.4	0.2	0.8	0.2
Test:	Sentences	2000		2000	
	Running Words+Punctuation	60332	57951	54260	57951
	Distinct Words	8279	6496	9048	6496
	OOVs [%]	0.4	0.2	0.7	0.2

Experimental Setup

Popović, Ney (2006)

- Europarl Corpus: 700K sentences
- POS Taggers: FreeLing, ENGCG, GERCG
- Trilingual Corpus: 670K sentences
- Studied effect of data sparsity, and training-corpus reordering.
- RWTH SMT System used for decoding.

		Spanish	English	German	English
Train:	Sentences	730740		751088	
	Running Words+Punctuation	15724914	15222146	15257678	16052330
	Vocabulary	113882	72739	205374	74708
	Singletons [%]	39.2	38.3	49.8	38.3
Dev:	Sentences	2000		2000	
	Running Words+Punctuation	60628	58655	55147	58655
	Distinct Words	8182	6547	9213	6547
	OOVs [%]	0.4	0.2	0.8	0.2
Test:	Sentences	2000		2000	
	Running Words+Punctuation	60332	57951	54260	57951
	Distinct Words	8279	6496	9048	6496
	OOVs [%]	0.4	0.2	0.7	0.2

Experimental Results

Popović, Ney (2006)

- Spanish To English
- English To Spanish
- English To German
- Spanish To German

Spanish→English			dev			test		
			WER	PER	BLEU	WER	PER	BLEU
700k	reordered	baseline	56.7	42.8	27.9	57.4	43.5	27.6
		reorder adjective	56.3	42.2	28.7	57.1	43.1	28.3
	rest	baseline	57.1	45.3	26.3	57.0	45.6	26.7
		reorder adjective	56.6	44.9	26.7	57.1	45.5	27.0
7k	reordered	baseline	65.9	49.4	19.6	66.5	49.5	19.7
		reorder adjective	64.2	48.3	22.0	65.0	48.6	22.0
	rest	baseline	64.6	51.6	19.5	64.9	51.6	20.9
		reorder adjective	64.4	51.0	20.3	64.9	51.2	21.6

Experimental Results

Popović, Ney (2006)

- Spanish To English
- English To Spanish
- English To German
- Spanish To German

English→Spanish			dev			test		
			WER	PER	BLEU	WER	PER	BLEU
700k	reordered	baseline	59.0	45.4	32.4	58.8	45.0	32.0
		reorder adjectives	58.6	45.2	32.8	58.3	44.4	32.5
	rest	baseline	57.0	46.4	32.4	57.9	47.9	32.2
		reorder adjectives	56.8	46.4	32.8	57.6	47.5	32.6
7k	reordered	baseline	67.6	52.4	23.6	67.3	51.8	23.5
		reorder adjectives	66.7	51.6	25.2	66.6	51.3	24.8
	rest	baseline	65.3	52.8	25.3	65.9	54.2	25.0
		reorder adjectives	65.2	52.8	25.7	65.9	54.1	25.7

Experimental Results

Popović, Ney (2006)

- Spanish To English
- English To Spanish
- English To German
- Spanish To German

English→German			dev			test		
			WER	PER	BLEU	WER	PER	BLEU
700k	reordered	baseline	70.3	57.8	17.8	70.4	57.4	17.8
		reorder verbs	69.7	56.9	18.8	69.8	56.2	19.0
	rest	baseline	64.9	54.0	23.7	64.0	53.8	24.3
		reorder verbs	64.6	53.8	24.1	64.0	53.5	25.0
7k	reordered	baseline	79.3	62.5	13.5	79.9	62.6	13.6
		reorder verbs	78.9	62.4	14.0	79.1	62.4	14.4
	rest	baseline	73.8	60.6	18.4	72.4	59.6	19.5
		reorder verbs	74.0	60.6	18.6	72.6	59.8	19.6

Experimental Results

Popović, Ney (2006)

- Spanish To English
- English To Spanish
- English To German
- Spanish To German

Spanish→German		dev			test		
		WER	PER	BLEU	WER	PER	BLEU
600k	baseline	68.4	55.0	21.2	68.7	55.2	21.2
	reorder adjectives	68.2	55.0	21.3	68.6	55.2	21.3
	reorder verbs	68.0	55.0	21.3	68.2	55.2	21.3
	reorder adjectives + verbs	67.9	54.7	21.6	68.1	55.1	21.5
6k	baseline	78.9	63.3	14.1	79.1	63.2	14.3
	reorder adjectives	78.3	62.9	14.8	78.6	62.9	14.9
	reorder verbs	78.8	63.3	14.2	78.9	63.0	14.4
	reorder adjectives + verbs	78.5	63.0	14.6	78.8	62.9	14.7

Outline

1. Introduction
   - Reordering in Phrase-based SMT
   - Word Orders between Language Pairs
2. POS-based Reordering
   - Popović, Ney (2006)
   - Crego, Mariño (2006)
   - Rottmann, Vogel (2007)
3. Syntactic Reordering
   - Xia, McCord (2004)
   - Collins, Koehn, Kučerová (2006)
   - Nguyen, Shimazu (2006)
   - Li et al. (2007)
4. Conclusion

Integration of POS-based source reordering into SMT

Crego, Mariño (2006)

- Languages: English to Spanish, and Spanish to English
- Corpus used: Europarl (1.28 M Sentences)
- POS Taggers Used: TNT (English), FreeLing(Spanish)
- Extract reordering patterns from corpus
- Build ‘extended search graph’ by applying reordering patterns to source sentence
- Use MARIE decoder (n-gram based SMT)

Reordering Framework

Crego, Mariño (2006)

- Get bi-directional GIZA alignments for the corpus, and take the UNION.
- Identify all crossing alignments produced.
- For each crossing:
  - Take the sequence of source side tags between the crossings
  - Create a rewrite pattern based on the order in which the source tags appear on the target side.

Reordering Framework

Crego, Mariño (2006)

- Get bi-directional GIZA alignments for the corpus, and take the UNION.
- Identify all crossing alignments produced.
- For each crossing:
  - Take the sequence of source side tags between the crossings
  - Create a rewrite pattern based on the order in which the source tags appear on the target side.

Reordering Framework

Crego, Mariño (2006)

- Get bi-directional GIZA alignments for the corpus, and take the UNION.
- Identify all crossing alignments produced.
- For each crossing:
  - Take the sequence of source side tags between the crossings
  - Create a rewrite pattern based on the order in which the source tags appear on the target side.

Reordering Framework

Crego, Mariño (2006)

- Get bi-directional GIZA alignments for the corpus, and take the UNION.
- Identify all crossing alignments produced.
- For each crossing:
  - Take the sequence of source side tags between the crossings
  - Create a rewrite pattern based on the order in which the source tags appear on the target side.

Reordering Framework

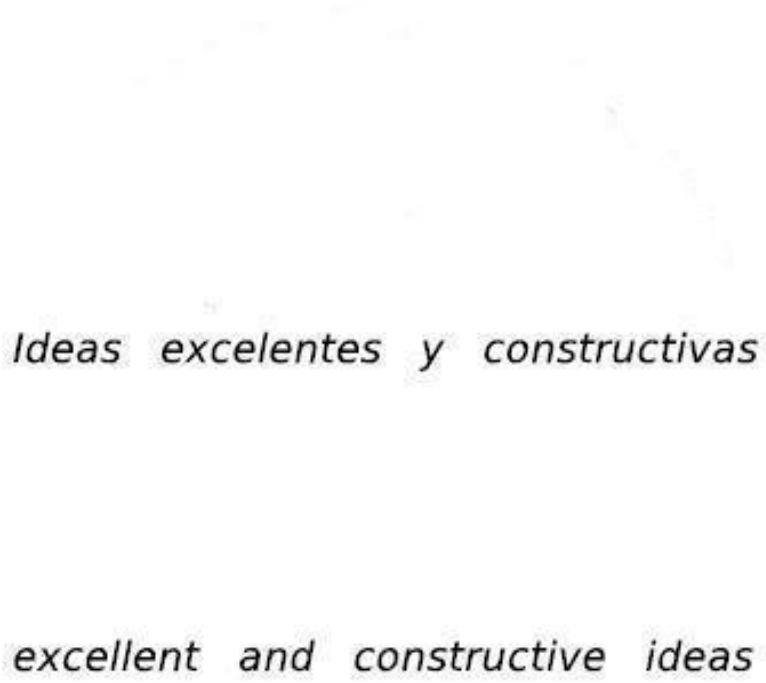
Crego, Mariño (2006)

- Get bi-directional GIZA alignments for the corpus, and take the UNION.
- Identify all crossing alignments produced.
- For each crossing:
  - Take the sequence of source side tags between the crossings
  - Create a rewrite pattern based on the order in which the source tags appear on the target side.

Reordering Framework

Crego, Mariño (2006)

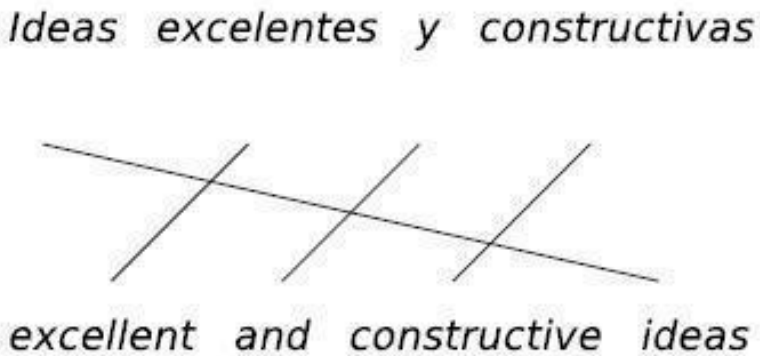
- For each crossing:
  - Take the sequence of source side tags between the crossings
  - Create a rewrite pattern based on the order in which the source tags appear on the target side.



Reordering Framework

Crego, Mariño (2006)

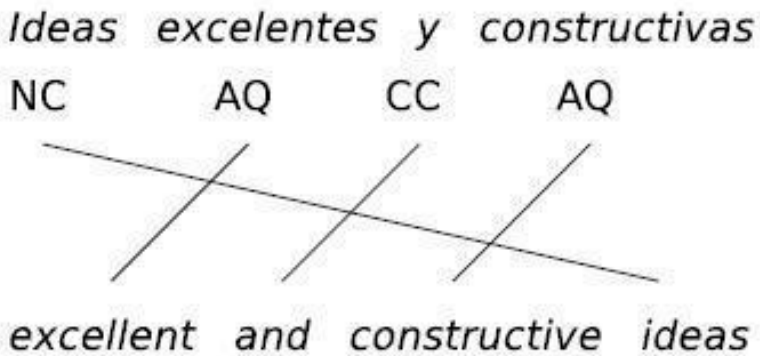
- For each crossing:
  - Take the sequence of source side tags between the crossings
  - Create a rewrite pattern based on the order in which the source tags appear on the target side.



Reordering Framework

Crego, Mariño (2006)

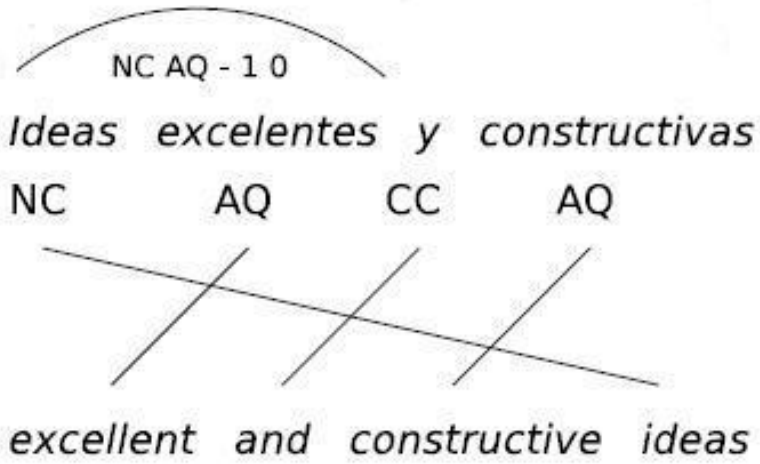
- For each crossing:
  - Take the sequence of source side tags between the crossings
  - Create a rewrite pattern based on the order in which the source tags appear on the target side.



Reordering Framework

Crego, Mariño (2006)

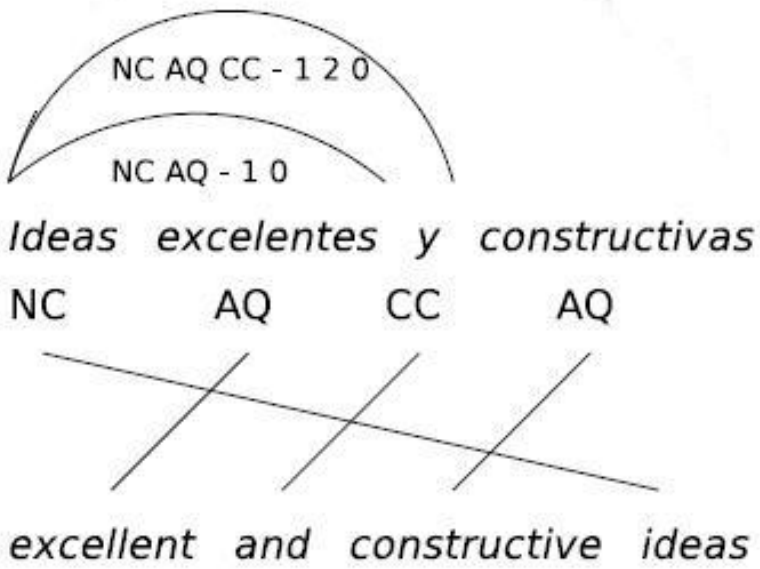
- For each crossing:
  - Take the sequence of source side tags between the crossings
  - Create a rewrite pattern based on the order in which the source tags appear on the target side.



Reordering Framework

Crego, Mariño (2006)

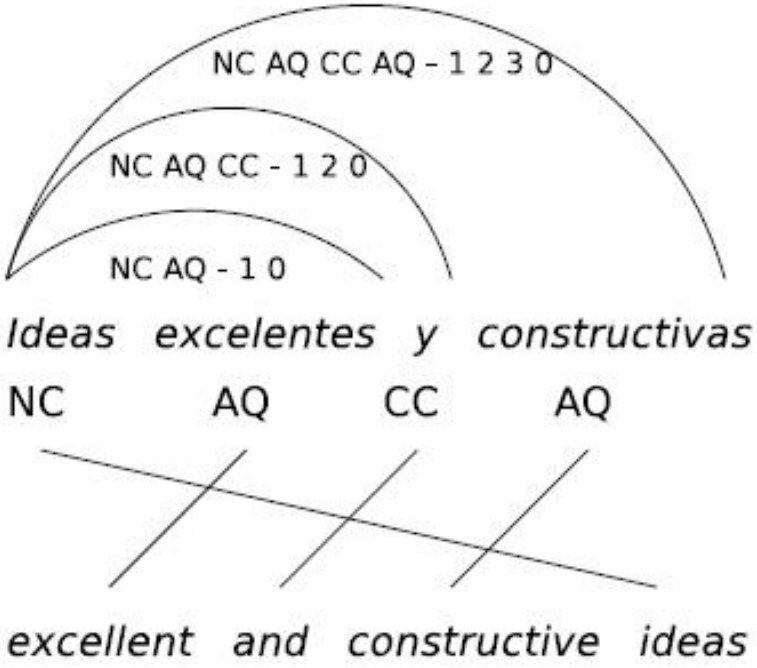
- For each crossing:
  - Take the sequence of source side tags between the crossings
  - Create a rewrite pattern based on the order in which the source tags appear on the target side.



Reordering Framework

Crego, Mariño (2006)

- For each crossing:
  - Take the sequence of source side tags between the crossings
  - Create a rewrite pattern based on the order in which the source tags appear on the target side.



Rule Filtering

Crego, Mariño (2006)

- Large number of rules can be extracted from the corpus
- Most of the rules appear due to wrong word alignments!
- Apply the following filters:
  1. Source and Target Phrases (where crossing occurs) must be atmost 4 words different in length.
  2. Maximum length of a rewrite pattern is 8.
  3. A pattern must occur at least 1000 times.
  4. $\frac{n(pattern)}{n(sourcewords)} > 0.2$
- Rules left after filtering: 29.
- Errors *still* remain!

Rule Filtering

Crego, Mariño (2006)

- Large number of rules can be extracted from the corpus
- Most of the rules appear due to wrong word alignments!
- Apply the following filters:
  1. Source and Target Phrases (where crossing occurs) must be atmost 4 words different in length.
  2. Maximum length of a rewrite pattern is 8.
  3. A pattern must occur at least 1000 times.
  4. $\frac{n(pattern)}{n(sourcewords)} > 0.2$
- Rules left after filtering: 29.
- Errors *still* remain!

Rule Filtering

Crego, Mariño (2006)

- Large number of rules can be extracted from the corpus
- Most of the rules appear due to wrong word alignments!
- Apply the following filters:
  1. Source and Target Phrases (where crossing occurs) must be atmost 4 words different in length.
  2. Maximum length of a rewrite pattern is 8.
  3. A pattern must occur at least 1000 times.
  4. $\frac{n(pattern)}{n(sourcewords)} > 0.2$
- Rules left after filtering: 29.
- Errors *still* remain!

Rule Filtering

Crego, Mariño (2006)

- Large number of rules can be extracted from the corpus
- Most of the rules appear due to wrong word alignments!
- Apply the following filters:
  1. Source and Target Phrases (where crossing occurs) must be atmost 4 words different in length.
  2. Maximum length of a rewrite pattern is 8.
  3. A pattern must occur at least 1000 times.
  4. $\frac{n(pattern)}{n(sourcewords)} > 0.2$
- Rules left after filtering: 29.
- Errors *still* remain!

Rule Filtering

Crego, Mariño (2006)

- Large number of rules can be extracted from the corpus
- Most of the rules appear due to wrong word alignments!
- Apply the following filters:
  1. Source and Target Phrases (where crossing occurs) must be atmost 4 words different in length.
  2. Maximum length of a rewrite pattern is 8.
  3. A pattern must occur at least 1000 times.
  4. $\frac{n(pattern)}{n(sourcewords)} > 0.2$
- Rules left after filtering: 29.
- Errors *still* remain!

Rule Filtering

Crego, Mariño (2006)

- Large number of rules can be extracted from the corpus
- Most of the rules appear due to wrong word alignments!
- Apply the following filters:
  1. Source and Target Phrases (where crossing occurs) must be atmost 4 words different in length.
  2. Maximum length of a rewrite pattern is 8.
  3. A pattern must occur at least 1000 times.
  4. $\frac{n(pattern)}{n(sourcewords)} > 0.2$
- Rules left after filtering: 29.
- Errors *still* remain!

Rule Filtering

Crego, Mariño (2006)

- Large number of rules can be extracted from the corpus
- Most of the rules appear due to wrong word alignments!
- Apply the following filters:
  1. Source and Target Phrases (where crossing occurs) must be atmost 4 words different in length.
  2. Maximum length of a rewrite pattern is 8.
  3. A pattern must occur at least 1000 times.
  4. $\frac{n(pattern)}{n(sourcewords)} > 0.2$
- Rules left after filtering: 29.
- Errors *still* remain!

Rule Filtering

Crego, Mariño (2006)

- Large number of rules can be extracted from the corpus
- Most of the rules appear due to wrong word alignments!
- Apply the following filters:
  1. Source and Target Phrases (where crossing occurs) must be atmost 4 words different in length.
  2. Maximum length of a rewrite pattern is 8.
  3. A pattern must occur at least 1000 times.
  4. $\frac{n(pattern)}{n(sourcewords)} > 0.2$
- Rules left after filtering: 29.
- Errors *still* remain!

Rule Filtering

Crego, Mariño (2006)

- Large number of rules can be extracted from the corpus
- Most of the rules appear due to wrong word alignments!
- Apply the following filters:
  1. Source and Target Phrases (where crossing occurs) must be atmost 4 words different in length.
  2. Maximum length of a rewrite pattern is 8.
  3. A pattern must occur at least 1000 times.
  4. $\frac{n(pattern)}{n(sourcewords)} > 0.2$
- Rules left after filtering: 29.
- Errors *still* remain!

Building an Extended Search Graph

Crego, Mariño (2006)

- Take the POS tagged input sentence
- Consider all applicable reordering rules
- Build a monotone search path for the input
- Apply each rule, and add entry to the path.

programa ambicioso y realista
NC AQ CC AQ

Building an Extended Search Graph

Crego, Mariño (2006)

- Take the POS tagged input sentence
- Consider all applicable reordering rules
- Build a monotone search path for the input
- Apply each rule, and add entry to the path.

programa ambicioso y realista
NC AQ CC AQ

NC AQ - 1 0
NC AQ CC - 1 2 0
NC AQ CC AQ - 1 2 3 0

Building an Extended Search Graph

Crego, Mariño (2006)

- Take the POS tagged input sentence
- Consider all applicable reordering rules
- Build a monotone search path for the input
- Apply each rule, and add entry to the path.

programa ambicioso y realista
NC AQ CC AQ

NC AQ - 1 0
NC AQ CC - 1 2 0
NC AQ CC AQ - 1 2 3 0

Building an Extended Search Graph

Crego, Mariño (2006)

- Take the POS tagged input sentence
- Consider all applicable reordering rules
- Build a monotone search path for the input
- Apply each rule, and add entry to the path.

programa ambicioso y realista
NC AQ CC AQ

NC AQ - 1 0
NC AQ CC - 1 2 0
NC AQ CC AQ - 1 2 3 0

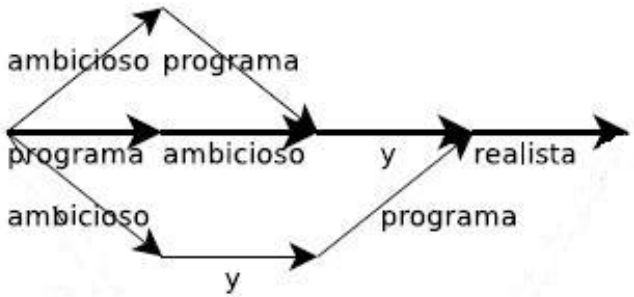
Building an Extended Search Graph

Crego, Mariño (2006)

- Take the POS tagged input sentence
- Consider all applicable reordering rules
- Build a monotone search path for the input
- Apply each rule, and add entry to the path.

programa ambicioso y realista
NC AQ CC AQ

NC AQ - 1 0
NC AQ CC - 1 2 0
NC AQ CC AQ - 1 2 3 0



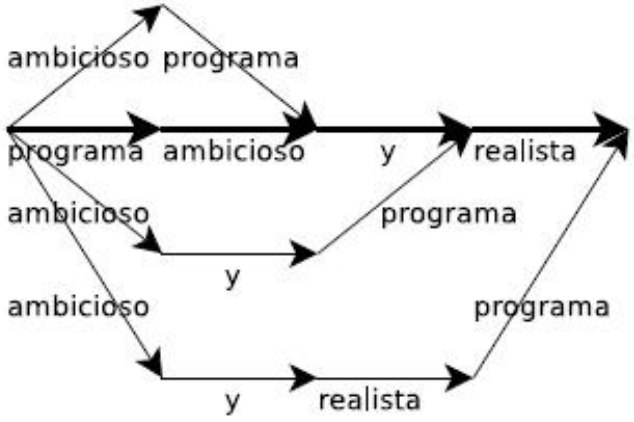
Building an Extended Search Graph

Crego, Mariño (2006)

- Take the POS tagged input sentence
- Consider all applicable reordering rules
- Build a monotone search path for the input
- Apply each rule, and add entry to the path.

programa ambicioso y realista
NC AQ CC AQ

NC AQ - 1 0
NC AQ CC - 1 2 0
NC AQ CC AQ - 1 2 3 0



Experimental Setup

Crego, Mariño (2006)

- Source side training corpus was reordered using the given rewrite patterns, and a 5-gram source-side LM was used.
- If more than one pattern can be applied, priority goes to the longest pattern.
- Three comparable systems:
  1. baseline: Monotone Search
  2. rgraph: Monotone search within reorder graphs
  3. pos: Monotone search within reorder graphs, with source side LM
- Two reference translations per sentence

Results

Crego, Mariño (2006)

Conf	bleu'	bleu	nist	mwer	per
Spanish-to-English					
base	.529	.552	10.69	34.40	25.32
rgraph	.533	.556	10.70	34.23	25.50
pos	.539	.564	10.75	33.75	25.41
English-to-Spanish					
base	.481	.480	9.84	41.18	31.11
rgraph	.490	.485	9.81	41.15	31.87
pos	.491	.489	9.91	40.29	31.27

Results

Crego, Mariño (2006)

Pattern	train	dev	test	swap	error	Example
NC RG AQ CC AQ ⇝ 1 2 3 4 0	1,406	1	1	1	0	ideas muy sencillas y elementales
NC AQ CC AQ ⇝ 1 2 3 0	27,119	13	**23**	**17**	**2**	programa ambicioso y realista
NC AQ RG AQ ⇝ 2 3 1 0	1,971	0	4	1	0	control fronterizo más estricto
NC CC NC AQ ⇝ 3 0 1 2	3,355	6	**12**	**6**	**3**	mezquitas y centros islámicos
NC RG AQ CC ⇝ 1 2 3 0	2,226	3	2	0	0	ideas muy sencillas y
AQ RG AQ ⇝ 1 2 0	2,777	21	*7*	*2*	*1*	europea más sólida
NC AQ AQ ⇝ 2 1 0	35,661	11	**24**	**18**	**3**	decisiones políticas delicadas
NC RG AQ ⇝ 1 2 0	32,887	0	**35**	**26**	**1**	ideas muy sencillas
NC RG RG ⇝ 1 2 0	1,473	0	3	3	2	texto mucho más
NC AQ ⇝ 1 0	877,580	113	**142**	**110**	**16**	preguntas serias
NC RG ⇝ 1 0	54,968	27	*47*	*7*	*7*	actividades aparentemente
AQ AQ ⇝ 1 0	46,509	14	*40*	*4*	*2*	medioambientales europeas
RN VM ⇝ 1 0	45,777	4	2	1	1	no promuevan
RG VA ⇝ 1 0	9,824	0	2	1	0	ahora habíamos
AQ RG ⇝ 1 0	8,701	11	*21*	*4*	*2*	suficiente todavía
RG VS ⇝ 1 0	5,043	1	1	1	0	supuestamente somos
VM PP ⇝ 1 0	4,769	6	**13**	**12**	**2**	estar ustedes
Total (17)	1,162,046	231	379	214	42	

Outline

1. Introduction
   - Reordering in Phrase-based SMT
   - Word Orders between Language Pairs
2. POS-based Reordering
   - Popović, Ney (2006)
   - Crego, Mariño (2006)
   - Rottmann, Vogel (2007)
3. Syntactic Reordering
   - Xia, McCord (2004)
   - Collins, Koehn, Kučerová (2006)
   - Nguyen, Shimazu (2006)
   - Li et al. (2007)
4. Conclusion

Word Reordering in SMT with POS based DM

Rottmann, Vogel(2007)

- Approach similar to Crego-Mariño
- POS based rules for reordering source text
- Use lattice to represent reorderings, and keep decoding monotone
- Use context information to help differentiate reorderings that are purely context based.

Learning Rules
Rottmann, Vogel(2007)

- Get alignments for bilingual corpus. Use POS tagger to get source side tags.
- Find crossings in alignment. Extract a reordering rule for every crossing.
- A rule which is observed as a part of a longer reordering is stored only if it also occurs as the longest reordering sequence in some other sentence pair.
- Filter rules for 5 or more occurrences. Assign rule scores using relative frequency.

- Example:

Learning Rules
Rottmann, Vogel(2007)

- Get alignments for bilingual corpus. Use POS tagger to get source side tags.
- Find crossings in alignment. Extract a reordering rule for every crossing.
- A rule which is observed as a part of a longer reordering is stored only if it also occurs as the longest reordering sequence in some other sentence pair.
- Filter rules for 5 or more occurrences. Assign rule scores using relative frequency.
- Example:

Learning Rules

Rottmann, Vogel(2007)

- Get alignments for bilingual corpus. Use POS tagger to get source side tags.
- Find crossings in alignment. Extract a reordering rule for every crossing.
- A rule which is observed as a part of a longer reordering is stored only if it also occurs as the longest reordering sequence in some other sentence pair.
- Filter rules for 5 or more occurrences. Assign rule scores using relative frequency.
- Example:

Learning Rules

Rottmann, Vogel(2007)

- Get alignments for bilingual corpus. Use POS tagger to get source side tags.
- Find crossings in alignment. Extract a reordering rule for every crossing.
- A rule which is observed as a part of a longer reordering is stored only if it also occurs as the longest reordering sequence in some other sentence pair.
- Filter rules for 5 or more occurrences. Assign rule scores using relative frequency.

- Example:

Learning Rules

Rottmann, Vogel(2007)

- Get alignments for bilingual corpus. Use POS tagger to get source side tags.
- Find crossings in alignment. Extract a reordering rule for every crossing.
- A rule which is observed as a part of a longer reordering is stored only if it also occurs as the longest reordering sequence in some other sentence pair.
- Filter rules for 5 or more occurrences. Assign rule scores using relative frequency.
- Example:

source sequence	rule	freq.
PDAT NN VVINF	3 1 2	0.60
VAFIN :: PDAT NN VVINF	3 1 2	0.63
KOUI :: PDAT NN VVINF	3 2 2	0.88
moechte :: PDAT NN VVINF	3 1 2	0.92

Applying Rules
Rottmann, Vogel(2007)

- Start with the POS tags of the input sentence.
- Match the POS tags to the rules and expand the lattice to reflect new word orders. Use context information if applicable.
- Once the lattice is built, assign rule scores.

Applying Rules
Rottmann, Vogel(2007)

- Start with the POS tags of the input sentence.
- Match the POS tags to the rules and expand the lattice to reflect new word orders. Use context information if applicable.
- Once the lattice is built, assign rule scores.

Applying Rules
Rottmann, Vogel(2007)

- Start with the POS tags of the input sentence.
- Match the POS tags to the rules and expand the lattice to reflect new word orders. Use context information if applicable.
- Once the lattice is built, assign rule scores.

Experimental Setup

Rottmann, Vogel(2007)

- Europarl corpus used. Two references for English and Spanish. One reference for German-English.
- POS Taggers used: Brill (Eng), Stuttgart Tree Tagger (German).
- Rules of upto length 15 extracted.

Experimental Setup

Rottmann, Vogel(2007)

- Europarl corpus used. Two references for English and Spanish. One reference for German-English.
- POS Taggers used: Brill (Eng), Stuttgart Tree Tagger (German).
- Rules of upto length 15 extracted.

Experimental Setup

Rottmann, Vogel(2007)

- Europarl corpus used. Two references for English and Spanish. One reference for German-English.
- POS Taggers used: Brill (Eng), Stuttgart Tree Tagger (German).
- Rules of upto length 15 extracted.

Results

Rottmann, Vogel(2007)

System		# en → es		# en → de		# de → en	
Context	Threshold	Rules Learned	Rule Matches	Rules Learned	Rule Matches	Rules Learned	Rule Matches
no	0.05	21388	12715	7929	60692	13396	72728
	0.1	6848	7740	4061	27809	8528	32233
	0.2	2321	4247	1291	8192	3738	14615
	0.3	1136	3369	469	3879	1601	7076
yes	0.01	72772	21119	32380	89225	38858	88549
	0.05	46014	6888	22836	36765	28485	37608
	0.1	25962	4924	15941	19319	21469	17148
	0.2	15304	3461	8462	8574	14466	9534

System	en → es
Baseline(RO3)	49.98
POS no Context 0.05	50.36
POS no Context 0.1	51.09
POS no Context 0.2	50.66
POS no Context 0.3	50.59
POS + Context 0.01	50.92
POS + Context 0.05	50.90
POS + Context 0.1	50.84
POS + Context 0.2	50.74
unseen Baseline(RO3)	48.51
unseen no Context	49.57
unseen with Context	49.49

Results

Rottmann, Vogel(2007)

System		# en → es		# en → de		# de → en	
Context	Threshold	Rules Learned	Rule Matches	Rules Learned	Rule Matches	Rules Learned	Rule Matches
no	0.05	21388	12715	7929	60692	13396	72728
	0.1	6848	7740	4061	27809	8528	32233
	0.2	2321	4247	1291	8192	3738	14615
	0.3	1136	3369	469	3879	1601	7076
yes	0.01	72772	21119	32380	89225	38858	88549
	0.05	46014	6888	22836	36765	28485	37608
	0.1	25962	4924	15941	19319	21469	17148
	0.2	15304	3461	8462	8574	14466	9534

System	en→ de	de → en
Baseline(RO3)	18.92	25.64
POS no Context 0.05	19.48	26.69
POS no Context 0.1	19.55	26.46
POS no Context 0.2	19.30	26.01
POS no Context 0.3	19.22	25.73
POS + Context 0.01	19.34	25.85
POS + Context 0.05	19.34	25.86
POS + Context 0.1	19.44	25.79
unseen Baseline(RO3)	17.69	23.70
unseen no Context	17.78	24.79
unseen with Context	17.79	23.87

Results

Rottmann, Vogel(2007)

System	en → es
unseen Baseline(RO3)	48.51
unseen no Context	49.52
unseen with Context	49.49
unseen combination	49.58
unseen combination-Lex	49.83

System	en→ de	de → en
unseen Baseline(RO3)	17.69	23.70
unseen no Context	17.78	24.79
unseen with Context	17.79	23.87
unseen combination	18.27	24.85
unseen combination-Lex	18.21	24.88

Results

Rottmann, Vogel(2007)

Corpus	en → de	de → en
Combination	19.61	26.88
Reordered (Giza)	19.44	26.76
Reordered (Lattice)	20.00	27.06
unseen Baseline(RO3)	17.69	23.70
unseen combination	18.27	24.85
unseen reordered corpus	18.42	25.06

Outline

1 Introduction
- Reordering in Phrase-based SMT
- Word Orders between Language Pairs

2 POS-based Reordering
- Popović, Ney (2006)
- Crego, Mariño (2006)
- Rottmann, Vogel (2007)

3 Syntactic Reordering
- Xia, McCord (2004)
- Collins, Koehn, Kučerová (2006)
- Nguyen, Shimazu (2006)
- Li et al. (2007)

4 Conclusion

Improving SMT With Automatic Rewrite Patterns

Xia, McCord(2004)

- Languages: English, French.
- Learn rewrite patterns for transformation of parse trees of source sentences.
- Newly ordered source sentence sent to the decoder.

Rewrite patterns

Xia, McCord(2004)

- A rewrite Pattern is a tuple: (Src Rule, Tgt Rule, Src Head Position, Tgt Head Position, Child-Alignment).
- If the Src or Tgt Rule contains the head word, the pattern is said to be lexicalized. (Useful e.g. french adjectives)

Learning Rewrite patterns

Xia, McCord(2004)

- Parse the sentences. They used Slot Grammar parsers.
- Align Phrases.
  - How?
  - Let S be source phrase, T be target phrase.
  - $Score(S,T) = \frac{links(S,T)}{Span(S)+Span(T)}$
  - Align S to the T that gives the max score.
- Extract Rewrite Patterns.
  - Find aligned nodes
  - Find if children on src and target align among themselves
  - Find if source head node aligns to target head node
  - Source node is lexicalized iff Target node is too.
  - Rule-length must be atmost 5
- Apply Rewrite Patterns
  - Use Greedy Strategy
  - Visit each node and apply the most specific pattern applicable at that node.

Learning Rewrite patterns

Xia, McCord(2004)

- Parse the sentences. They used Slot Grammar parsers.
- Align Phrases.
  - How?
  - Let S be source phrase, T be target phrase.
  - $Score(S, T) = \frac{links(S,T)}{Span(S)+Span(T)}$
  - Align S to the T that gives the max score.
- Extract Rewrite Patterns.
  - Find aligned nodes
  - Find if children on src and target align among themselves
  - Find if source head node aligns to target head node
  - Source node is lexicalized iff Target node is too.
  - Rule-length must be atmost 5
- Apply Rewrite Patterns
  - Use Greedy Strategy
  - Visit each node and apply the most specific pattern applicable at that node.

Learning Rewrite patterns

Xia, McCord(2004)

- Parse the sentences. They used Slot Grammar parsers.
- Align Phrases.
  - How?
    - Let S be source phrase, T be target phrase.
    - $Score(S,T) = \frac{links(S,T)}{Span(S)+Span(T)}$
    - Align S to the T that gives the max score.
- Extract Rewrite Patterns.
  - Find aligned nodes
  - Find if children on src and target align among themselves
  - Find if source head node aligns to target head node
  - Source node is lexicalized iff Target node is too.
  - Rule-length must be atmost 5
- Apply Rewrite Patterns
  - Use Greedy Strategy
  - Visit each node and apply the most specific pattern applicable at that node.

Learning Rewrite patterns

Xia, McCord(2004)

- Parse the sentences. They used Slot Grammar parsers.
- Align Phrases.
  - How?
  - Let S be source phrase, T be target phrase.
  - $Score(S,T) = \frac{links(S,T)}{Span(S)+Span(T)}$
  - Align S to the T that gives the max score.
- Extract Rewrite Patterns.
  - Find aligned nodes
  - Find if children on src and target align among themselves
  - Find if source head node aligns to target head node
  - Source node is lexicalized iff Target node is too.
  - Rule-length must be atmost 5
- Apply Rewrite Patterns
  - Use Greedy Strategy
  - Visit each node and apply the most specific pattern applicable at that node.

Learning Rewrite patterns

Xia, McCord(2004)

- Parse the sentences. They used Slot Grammar parsers.
- Align Phrases.
  - How?
  - Let S be source phrase, T be target phrase.
  - $Score(S, T) = \frac{links(S,T)}{Span(S)+Span(T)}$
  - Align S to the T that gives the max score.
- Extract Rewrite Patterns.
  - Find aligned nodes
  - Find if children on src and target align among themselves
  - Find if source head node aligns to target head node
  - Source node is lexicalized iff Target node is too.
  - Rule-length must be atmost 5
- Apply Rewrite Patterns
  - Use Greedy Strategy
  - Visit each node and apply the most specific pattern applicable at that node.

Learning Rewrite patterns

Xia, McCord(2004)

- Parse the sentences. They used Slot Grammar parsers.
- Align Phrases.
  - How?
  - Let S be source phrase, T be target phrase.
  - $Score(S, T) = \frac{links(S,T)}{Span(S)+Span(T)}$
  - Align S to the T that gives the max score.
- Extract Rewrite Patterns.
  - Find aligned nodes
  - Find if children on src and target align among themselves
  - Find if source head node aligns to target head node
  - Source node is lexicalized iff Target node is too.
  - Rule-length must be atmost 5
- Apply Rewrite Patterns
  - Use Greedy Strategy
  - Visit each node and apply the most specific pattern applicable at that node.

Learning Rewrite patterns

Xia, McCord(2004)

- Parse the sentences. They used Slot Grammar parsers.
- Align Phrases.
  - How?
  - Let S be source phrase, T be target phrase.
  - $Score(S, T) = \frac{links(S,T)}{Span(S)+Span(T)}$
  - Align S to the T that gives the max score.
- Extract Rewrite Patterns.
  - Find aligned nodes
  - Find if children on src and target align among themselves
  - Find if source head node aligns to target head node
  - Source node is lexicalized iff Target node is too.
  - Rule-length must be atmost 5
- Apply Rewrite Patterns
  - Use Greedy Strategy
  - Visit each node and apply the most specific pattern applicable at that node.

Learning Rewrite patterns

Xia, McCord(2004)

- Parse the sentences. They used Slot Grammar parsers.
- Align Phrases.
  - How?
  - Let S be source phrase, T be target phrase.
  - $Score(S, T) = \frac{links(S,T)}{Span(S)+Span(T)}$
  - Align S to the T that gives the max score.
- Extract Rewrite Patterns.
  - Find aligned nodes
  - Find if children on src and target align among themselves
  - Find if source head node aligns to target head node
  - Source node is lexicalized iff Target node is too.
  - Rule-length must be atmost 5
- Apply Rewrite Patterns
  - Use Greedy Strategy
  - Visit each node and apply the most specific pattern applicable at that node.

Learning Rewrite patterns

Xia, McCord(2004)

- Parse the sentences. They used Slot Grammar parsers.
- Align Phrases.
  - How?
  - Let S be source phrase, T be target phrase.
  - $Score(S, T) = \frac{links(S,T)}{Span(S)+Span(T)}$
  - Align S to the T that gives the max score.
- Extract Rewrite Patterns.
  - Find aligned nodes
  - Find if children on src and target align among themselves
  - Find if source head node aligns to target head node
  - Source node is lexicalized iff Target node is too.
  - Rule-length must be atmost 5
- Apply Rewrite Patterns
  - Use Greedy Strategy
  - Visit each node and apply the most specific pattern applicable at that node.

Learning Rewrite patterns

Xia, McCord(2004)

- Parse the sentences. They used Slot Grammar parsers.
- Align Phrases.
  - How?
  - Let S be source phrase, T be target phrase.
  - $Score(S, T) = \frac{links(S,T)}{Span(S)+Span(T)}$
  - Align S to the T that gives the max score.
- Extract Rewrite Patterns.
  - Find aligned nodes
  - Find if children on src and target align among themselves
  - Find if source head node aligns to target head node
  - Source node is lexicalized iff Target node is too.
  - Rule-length must be atmost 5
- Apply Rewrite Patterns
  - Use Greedy Strategy
  - Visit each node and apply the most specific pattern applicable at that node.

Learning Rewrite patterns

Xia, McCord(2004)

- Parse the sentences. They used Slot Grammar parsers.
- Align Phrases.
  - How?
  - Let S be source phrase, T be target phrase.
  - $Score(S, T) = \frac{links(S,T)}{Span(S)+Span(T)}$
  - Align S to the T that gives the max score.
- Extract Rewrite Patterns.
  - Find aligned nodes
  - Find if children on src and target align among themselves
  - Find if source head node aligns to target head node
  - Source node is lexicalized iff Target node is too.
  - Rule-length must be atmost 5
- Apply Rewrite Patterns
  - Use Greedy Strategy
  - Visit each node and apply the most specific pattern applicable at that node.

Learning Rewrite patterns

Xia, McCord(2004)

- Parse the sentences. They used Slot Grammar parsers.
- Align Phrases.
  - How?
  - Let S be source phrase, T be target phrase.
  - $Score(S, T) = \frac{links(S,T)}{Span(S)+Span(T)}$
  - Align S to the T that gives the max score.
- Extract Rewrite Patterns.
  - Find aligned nodes
  - Find if children on src and target align among themselves
  - Find if source head node aligns to target head node
  - Source node is lexicalized iff Target node is too.
  - Rule-length must be atmost 5
- Apply Rewrite Patterns
  - Use Greedy Strategy
  - Visit each node and apply the most specific pattern applicable at that node.

Learning Rewrite patterns

Xia, McCord(2004)

- Parse the sentences. They used Slot Grammar parsers.
- Align Phrases.
  - How?
  - Let S be source phrase, T be target phrase.
  - $Score(S, T) = \frac{links(S,T)}{Span(S)+Span(T)}$
  - Align S to the T that gives the max score.
- Extract Rewrite Patterns.
  - Find aligned nodes
  - Find if children on src and target align among themselves
  - Find if source head node aligns to target head node
  - Source node is lexicalized iff Target node is too.
  - Rule-length must be atmost 5
- Apply Rewrite Patterns
  - Use Greedy Strategy
  - Visit each node and apply the most specific pattern applicable at that node.

Learning Rewrite patterns

Xia, McCord(2004)

- Parse the sentences. They used Slot Grammar parsers.
- Align Phrases.
  - How?
  - Let S be source phrase, T be target phrase.
  - $Score(S, T) = \frac{links(S,T)}{Span(S)+Span(T)}$
  - Align S to the T that gives the max score.
- Extract Rewrite Patterns.
  - Find aligned nodes
  - Find if children on src and target align among themselves
  - Find if source head node aligns to target head node
  - Source node is lexicalized iff Target node is too.
  - Rule-length must be atmost 5
- Apply Rewrite Patterns
  - Use Greedy Strategy
  - Visit each node and apply the most specific pattern applicable at that node.

Learning Rewrite patterns

Xia, McCord(2004)

- Parse the sentences. They used Slot Grammar parsers.
- Align Phrases.
  - How?
  - Let S be source phrase, T be target phrase.
  - $Score(S, T) = \frac{links(S,T)}{Span(S)+Span(T)}$
  - Align S to the T that gives the max score.
- Extract Rewrite Patterns.
  - Find aligned nodes
  - Find if children on src and target align among themselves
  - Find if source head node aligns to target head node
  - Source node is lexicalized iff Target node is too.
  - Rule-length must be atmost 5
- Apply Rewrite Patterns
  - Use Greedy Strategy
  - Visit each node and apply the most specific pattern applicable at that node.

Results
Xia, McCord(2004)

- En-Fr Canadian Hansard corpus used (90M word)
- 2.9M rewrite patterns extracted
- Patterns filtered down to 56K.
- 1042 patterns were lexicalized

	non-monotonic decoding	monotonic decoding
rewrite patterns not used	0.187	0.196
rewrite patterns used	0.185	0.215

Outline

1. Introduction
   - Reordering in Phrase-based SMT
   - Word Orders between Language Pairs
2. POS-based Reordering
   - Popović, Ney (2006)
   - Crego, Mariño (2006)
   - Rottmann, Vogel (2007)
3. Syntactic Reordering
   - Xia, McCord (2004)
   - Collins, Koehn, Kučerová (2006)
   - Nguyen, Shimazu (2006)
   - Li et al. (2007)
4. Conclusion

Clause Restructuring for SMT

Collins, Koehn, Kučerová (2006)

- Works with German. German has more reordering phenomena than French wrt English.
- Rules are manually crafted. Not automatically learned.
- Rule-set consists of 6 transformations, very specific to German.

Results

Collins, Koehn, Kučerová (2006)

- Europarl Corpus (750K sentences) used.
- Baseline SMT System score: 25.2
- New System BLEU Score: 26.8
- Human Evaluation performed on 100 random sentences by 2 judges
  - 33 sentences showed improvement over SMT
  - 13 sentences were worse after reordering

Outline

1. Introduction
   - Reordering in Phrase-based SMT
   - Word Orders between Language Pairs
2. POS-based Reordering
   - Popović, Ney (2006)
   - Crego, Mariño (2006)
   - Rottmann, Vogel (2007)
3. Syntactic Reordering
   - Xia, McCord (2004)
   - Collins, Koehn, Kučerová (2006)
   - Nguyen, Shimazu (2006)
   - Li et al. (2007)
4. Conclusion

Syntactic Transformational Model for SMT

Nguyen, Shimazu (2006)

- Unlike previous syntactic methods, this transformational model is based on statistical decisions.
- Rules are learned from corpora, and scored.
- Application of rules to new sentences is also done statistically.

What is a Transformation?

Nguyen, Shimazu (2006)

- There could be multiple ways to reorder a CFG rule.
- Lexicalization of rules can help decide which reordering should be applied.
- Lexicalization can lead to too many rules: score estimation is a problem.
- Use LPCFG to get the scores

The Training process

Nguyen, Shimazu (2006)

- Parse text. Get GIZA alignments.
- Align source-side phrases. (Similar to Xia, McCord (2004))
- If there are one-to-many alignments:
  - If source span is one word, choose the best link based on intersection of bidirectional alignments and lexical scores.
  - For each word outside source phrase, there should be no link to any word outside the target phrase, and vice versa.
- For each node, based on the target phrase position of children, learn a reordering rule.
- Score all rules:
  $p(LHS \rightarrow RHS|LHS \rightarrow RHS') = \frac{n(LHS \rightarrow RHS|LHS \rightarrow RHS')}{n(LHS \rightarrow RHS')}$.

The Training process

Nguyen, Shimazu (2006)

- Parse text. Get GIZA alignments.
- Align source-side phrases. (Similar to Xia, McCord (2004))
- If there are one-to-many alignments:
  - If source span is one word, choose the best link based on intersection of bidirectional alignments and lexical scores.
  - For each word outside source phrase, there should be no link to any word outside the target phrase, and vice versa.
- For each node, based on the target phrase position of children, learn a reordering rule.
- Score all rules:
  $p(LHS \rightarrow RHS|LHS \rightarrow RHS') = \frac{n(LHS \rightarrow RHS|LHS \rightarrow RHS')}{n(LHS \rightarrow RHS')}$.

The Training process

Nguyen, Shimazu (2006)

- Parse text. Get GIZA alignments.
- Align source-side phrases. (Similar to Xia, McCord (2004))
- If there are one-to-many alignments:
  - If source span is one word, choose the best link based on intersection of bidirectional alignments and lexical scores.
  - For each word outside source phrase, there should be no link to any word outside the target phrase, and vice versa.
- For each node, based on the target phrase position of children, learn a reordering rule.
- Score all rules:
  $p(LHS \rightarrow RHS|LHS \rightarrow RHS') = \frac{n(LHS \rightarrow RHS|LHS \rightarrow RHS')}{n(LHS \rightarrow RHS')}$.

The Training process

Nguyen, Shimazu (2006)

- Parse text. Get GIZA alignments.
- Align source-side phrases. (Similar to Xia, McCord (2004))
- If there are one-to-many alignments:
  - If source span is one word, choose the best link based on intersection of bidirectional alignments and lexical scores.
  - For each word outside source phrase, there should be no link to any word outside the target phrase, and vice versa.
- For each node, based on the target phrase position of children, learn a reordering rule.
- Score all rules:
  $p(LHS \rightarrow RHS|LHS \rightarrow RHS') = \frac{n(LHS \rightarrow RHS|LHS \rightarrow RHS')}{n(LHS \rightarrow RHS')}$.

The Training process

Nguyen, Shimazu (2006)

- Parse text. Get GIZA alignments.
- Align source-side phrases. (Similar to Xia, McCord (2004))
- If there are one-to-many alignments:
  - If source span is one word, choose the best link based on intersection of bidirectional alignments and lexical scores.
  - For each word outside source phrase, there should be no link to any word outside the target phrase, and vice versa.
- For each node, based on the target phrase position of children, learn a reordering rule.
- Score all rules:
  $p(LHS \rightarrow RHS|LHS \rightarrow RHS') = \frac{n(LHS \rightarrow RHS|LHS \rightarrow RHS')}{n(LHS \rightarrow RHS')}$.

The Training process

Nguyen, Shimazu (2006)

- Parse text. Get GIZA alignments.
- Align source-side phrases. (Similar to Xia, McCord (2004))
- If there are one-to-many alignments:
  - If source span is one word, choose the best link based on intersection of bidirectional alignments and lexical scores.
  - For each word outside source phrase, there should be no link to any word outside the target phrase, and vice versa.
- For each node, based on the target phrase position of children, learn a reordering rule.
- Score all rules:
  $p(LHS \rightarrow RHS|LHS \rightarrow RHS') = \frac{n(LHS \rightarrow RHS|LHS \rightarrow RHS')}{n(LHS \rightarrow RHS')}$.

The Training process

Nguyen, Shimazu (2006)

- Parse text. Get GIZA alignments.
- Align source-side phrases. (Similar to Xia, McCord (2004))
- If there are one-to-many alignments:
  - If source span is one word, choose the best link based on intersection of bidirectional alignments and lexical scores.
  - For each word outside source phrase, there should be no link to any word outside the target phrase, and vice versa.
- For each node, based on the target phrase position of children, learn a reordering rule.
- Score all rules:
  $p(LHS \rightarrow RHS|LHS \rightarrow RHS') = \frac{n(LHS \rightarrow RHS|LHS \rightarrow RHS')}{n(LHS \rightarrow RHS')}$.

Applying the rules

Nguyen, Shimazu (2006)

- Parse the input sentence.
- Lexicalize the tree. (Propogate heads bottom up).
- For the tree, apply the best possible transformation sequence.
- $Q^* = \{RS_i^* : RS_i^* = \text{argmax}\left[P(L_i \rightarrow R_i | L_i \rightarrow R_i') * P(L_i \rightarrow R_i')\right]\}$
- Extract the surface string

Applying the rules

Nguyen, Shimazu (2006)

- Parse the input sentence.
- Lexicalize the tree. (Propogate heads bottom up).
- For the tree, apply the best possible transformation sequence.
- $Q^* = \{RS_i^* : RS_i^* = \text{argmax}\left[P(L_i \rightarrow R_i | L_i \rightarrow R_i') * P(L_i \rightarrow R_i')\right]\}$
- Extract the surface string

Applying the rules

Nguyen, Shimazu (2006)

- Parse the input sentence.
- Lexicalize the tree. (Propogate heads bottom up).
- For the tree, apply the best possible transformation sequence.
- $Q^* = \{RS_i^* : RS_i^* = \text{argmax}\left[P(L_i \rightarrow R_i | L_i \rightarrow R_i') * P(L_i \rightarrow R_i')\right]\}$
- Extract the surface string

Applying the rules

Nguyen, Shimazu (2006)

- Parse the input sentence.
- Lexicalize the tree. (Propogate heads bottom up).
- For the tree, apply the best possible transformation sequence.
- $Q^* = \{RS_i^* : RS_i^* = \text{argmax}\left[P(L_i \rightarrow R_i | L_i \rightarrow R_i') * P(L_i \rightarrow R_i')\right]\}$
- Extract the surface string

Applying the rules

Nguyen, Shimazu (2006)

- Parse the input sentence.
- Lexicalize the tree. (Propogate heads bottom up).
- For the tree, apply the best possible transformation sequence.
- $Q^* = \{RS_i^* : RS_i^* = \text{argmax}\left[P(L_i \rightarrow R_i | L_i \rightarrow R_i') * P(L_i \rightarrow R_i')\right]\}$
- Extract the surface string

Experiments
Nguyen, Shimazu (2006)

- Experimented with English, Vietnamese, French
- Restricted training to 40K trees.

Experiments
Nguyen, Shimazu (2006)

- Experimented with English, Vietnamese, French
- Restricted training to 40K trees.

Experiments
Nguyen, Shimazu (2006)

- Experimented with English, Vietnamese, French
- Restricted training to 40K trees.

Corpus	UCFGRs	TRGs	AGs
Computer	4779	3702	951
Conversation	3634	2642	669
Europarl	14462	10738	3706

Results

Nguyen, Shimazu (2006)

Corpus	Baseline	Syntactic transformation
Computer	45.12	47.62
Conversation	33.85	36.26
Europarl	26.41	28.02

Results

Nguyen, Shimazu (2006)

Maximum phrase size	2	3	4	5	6
Pharaoh	21.71	24.84	25.74	26.19	26.41
Syntactic transformation	24.1	27.01	27.74	27.88	28.02

Results

Nguyen, Shimazu (2006)

Training-set size	10K	20K	40K	80K	94K
Pharaoh	21.84	23.35	24.43	25.43	25.74
Syntactic transformation	23.65	25.67	26.86	27.52	27.74

Results

Nguyen, Shimazu (2006)

Training-set size	10K	20K	40K	80K	94K
Pharaoh	1.98	2.52	2.93	3.45	3.67
Syntactic transformation	0.1	0.13	0.16	0.19	0.22

Outline

1 Introduction
- Reordering in Phrase-based SMT
- Word Orders between Language Pairs

2 POS-based Reordering
- Popović, Ney (2006)
- Crego, Mariño (2006)
- Rottmann, Vogel (2007)

3 Syntactic Reordering
- Xia, McCord (2004)
- Collins, Koehn, Kučerová (2006)
- Nguyen, Shimazu (2006)
- Li et al. (2007)

4 Conclusion

Probabilistic approach to Syntax-based RO for SMT

Li et al. (2007)

- Previous syntactic systems propose: $S \rightarrow S' \rightarrow T$.
- They propose: $S \rightarrow n * S' \rightarrow n * T \rightarrow \hat{T}$.
- Give up using rewrite patterns. Instead acquire RO knowledge.

Training
Li et al. (2007)

- Simplified case: binary tree.
- Let $A \rightarrow B\,C$ be a node in the tree.
- Use word alignments to determine:
  1. What is the minimum and maximum position on target side that yield of B aligns to? ($T(B)$)
  2. What is the minimum and maximum position on target side that yield of C aligns to? ($T(C)$)
- If $T(B)$ and $T(C)$ overlap:
  1. Keep remove the worst-scoring link from word-alignments in the phrases until overlap goes away.
  2. If too many links are removed, the node A not used as training item.
- Easily extended to n-ary trees.

Learning to Reorder

Li et al. (2007)

- Strategy 1: Learn Rules
  1. Consider every rule $Z : XY$ in the trees
  2. Use relative frequency to estimate how many times it is reordered
- Strategy 2: Maximum Entropy Model:
  1. Binary classification of whether the children of a node are reordered or not.
  2. Features used:
     - Leftmost word of a phrase and its POS
     - Rightmost word of a phrase and its POS
     - Head word and its POS
     - Context words (phrase $\pm$ 1 word) and their POS.

Applying learned knowledge

Li et al. (2007)

- Use bottom-up approach
- If current node has unary production, assign it a score of 1.
- Determine which rules are applicable at the node, or determine via EM if node should be reordered. Obtain the rule score of the new order.
- Set value of current node: *val* = RuleScore $*$ Product of Values of Children.
- Keep track of N-highest probabilities of nodes. They correspond to the N-Best list.

During Decoding

Li et al. (2007)

- Split input sentence into clauses, using IP nodes in parse trees.
- Reorder each clause, get an n-best list for each clause.
- Translate each of the n-best items of each clause
- Choose best-scoring translation of each clause
- Combine these translations back to one sentence.
- Decoder has additional feature: $P(S \rightarrow S')$. This is the score of the tree for each reordering.

Experiments
Li et al. (2007)

- Pharaoh-like decoder
- GIGAword corpus as training data
- MT05 Chinese-English data for testing.

Experiments
Li et al. (2007)

- Pharaoh-like decoder
- GIGAword corpus as training data
- MT05 Chinese-English data for testing.

Experiments
Li et al. (2007)

- Pharaoh-like decoder
- GIGAword corpus as training data
- MT05 Chinese-English data for testing.

Experiments
Li et al. (2007)

- Pharaoh-like decoder
- GIGAword corpus as training data
- MT05 Chinese-English data for testing.

Branching Factor	2	3	>3
Count	12294	3173	1280
Percentage	73.41	18.95	7.64

Results

Li et al. (2007)

Test	Setting	BLEU
B1	standard phrase-based SMT	29.22
B2	(B1) + clause splitting	29.13

Test	Setting	BLEU 2-ary	BLEU 2,3-ary
1	rule	29.77	30.31
2	ME (phrase label)	29.93	30.49
3	ME (left,right)	30.10	30.53
4	ME ((3)+head)	30.24	30.71
5	ME ((3)+phrase label)	30.12	30.30
6	ME ((4)+context)	30.24	30.76

Results

Li et al. (2007)

Test	Setting	BLEU
a	length constraint	30.52
b	DL=0	30.48
c	n=100	30.78

Bright future with forests

Wildcard rules

- Almost all systems currently use rules, or patterns to transform text
- It would be interesting to use wildcards in rules
- For POS, this may be easier to define: AUX-V-* becomes AUX-*-V
- Wildcards with trees could be really wild!
- Recent experiments with reordering arabic trees: ‘good’ rules don’t just contain wildcards, they contain tgrep style regular expressions.
- Learning such rules is a very challenging problem.

Wildcard rules

- Almost all systems currently use rules, or patterns to transform text
- It would be interesting to use wildcards in rules
- For POS, this may be easier to define: AUX-V-* becomes AUX-*-V
- Wildcards with trees could be really wild!
- Recent experiments with reordering arabic trees: ‘good’ rules don’t just contain wildcards, they contain tgrep style regular expressions.
- Learning such rules is a very challenging problem.

Wildcard rules

- Almost all systems currently use rules, or patterns to transform text
- It would be interesting to use wildcards in rules
- For POS, this may be easier to define: AUX-V-* becomes AUX-*-V
- Wildcards with trees could be really wild!
- Recent experiments with reordering arabic trees: ‘good’ rules don’t just contain wildcards, they contain tgrep style regular expressions.
- Learning such rules is a very challenging problem.

Wildcard rules

- Almost all systems currently use rules, or patterns to transform text
- It would be interesting to use wildcards in rules
- For POS, this may be easier to define: AUX-V-* becomes AUX-*-V
- Wildcards with trees could be really wild!
- Recent experiments with reordering arabic trees: ‘good’ rules don’t just contain wildcards, they contain tgrep style regular expressions.
- Learning such rules is a very challenging problem.

Wildcard rules

- Almost all systems currently use rules, or patterns to transform text
- It would be interesting to use wildcards in rules
- For POS, this may be easier to define: AUX-V-* becomes AUX-*-V
- Wildcards with trees could be really wild!
- Recent experiments with reordering arabic trees: ‘good’ rules don’t just contain wildcards, they contain tgrep style regular expressions.
- Learning such rules is a very challenging problem.

Wildcard rules

- Almost all systems currently use rules, or patterns to transform text
- It would be interesting to use wildcards in rules
- For POS, this may be easier to define: AUX-V-* becomes AUX-*-V
- Wildcards with trees could be really wild!
- Recent experiments with reordering arabic trees: ‘good’ rules don’t just contain wildcards, they contain tgrep style regular expressions.
- Learning such rules is a very challenging problem.

Parser issues

- Syntax-based rules depend on what parser is used.
- Stanford parser creates deep trees. Too many nodes may hamper the process of learning good rules. Shallow parsers may be an option.
- Not all languages have a parser trained on large data. Does a light-weight parser introduce too much noise in the forest? Are current methods robust to parser errors ?

Parser issues

- Syntax-based rules depend on what parser is used.
- Stanford parser creates deep trees. Too many nodes may hamper the process of learning good rules. Shallow parsers may be an option.
- Not all languages have a parser trained on large data. Does a light-weight parser introduce too much noise in the forest? Are current methods robust to parser errors ?

Parser issues

- Syntax-based rules depend on what parser is used.
- Stanford parser creates deep trees. Too many nodes may hamper the process of learning good rules. Shallow parsers may be an option.
- Not all languages have a parser trained on large data. Does a light-weight parser introduce too much noise in the forest? Are current methods robust to parser errors ?

Evaluation issues

- Is BLEU a good metric to study effect of reordering sentences?
- If source-reordering claims to produce more grammatical sentences, could the grammaticality be evaluated?

Evaluation issues

- Is BLEU a good metric to study effect of reordering sentences?
- If source-reordering claims to produce more grammatical sentences, could the grammaticality be evaluated?

Questions

Summary

Works great the reordering of sentences source.
The problem but solved is not.