Morpho-syntactic Arabic Preprocessing for Arabic-to-English Statistical Machine Translation

Anas El Isbihani **Shahram Khadivi** **Oliver Bender** **Hermann Ney**
Lehrstuhl für Informatik 6 - Computer Science Department
RWTH Aachen University
D-52056 Aachen, Germany
{isbihani,khadivi,bender,ney}@informatik.rwth-aachen.de

Abstract

The Arabic language has far richer systems of inflection and derivation than English which has very little morphology. This morphology difference causes a large gap between the vocabulary sizes in any given parallel training corpus. Segmentation of inflected Arabic words is a way to smooth its highly morphological nature. In this paper, we describe some statistically and linguistically motivated methods for Arabic word segmentation. Then, we show the efficiency of proposed methods on the Arabic-English BTEC and NIST tasks.

1 Introduction

Arabic is a highly inflected language compared to English which has very little morphology. This morphological richness makes statistical machine translation from Arabic to English a challenging task. A usual phenomenon in Arabic is the attachment of a group of words which are semantically dependent on each other. For instance, prepositions like “and” and “then” are usually attached to the next word. This applies also to the definite article “the”. In addition, personal pronouns are attached to the end of verbs, whereas possessive pronouns are attached to the end of the previous word, which constitutes the possessed object. Hence, an Arabic word can be decomposed into “prefixes, stem and suffixes”. We restrict the set of prefixes and suffixes to those showed in Table 1 and 2, where each of the prefixes and suffixes has at least one meaning which can be represented by a single word in the target language. Some prefixes can be combined. For example the word *wbAlqlm* (وبالقلم which means “and with the pen”) has a prefix which is a combination of three prefixes, namely *w*, *b* and *Al*. The suffixes we handle in this paper can not be combined with each other. Thus, the compound word pattern handled here is “prefixes-stem-suffix“.

All possible prefix combinations that do not contain *Al* allow the stem to have a suffix. Note that there are other suffixes that are not handled here, such as *At* (ات), *An* (ان) and *wn* (ون) which make the plural form of a word. The reason why we omit them is that they do not have their own meaning. The impact of Arabic morphology is that the vocabulary size and the number of singletons can be dramatically high, i.e. the Arabic words are not seen often enough to be learned by statistical machine translation models. This can lead to an inefficient alignment.

In order to deal with this problem and to improve the performance of statistical machine translation, each word must be decomposed into its parts. In (Larkey et al., 2002) it was already shown that word segmentation for Arabic improves information retrieval. In (Lee et al., 2003) a statistical approach for Arabic word segmentation was presented. It decomposes each word into a sequence of morphemes (prefixes-stem-suffixes), where all possible prefixes and suffixes (not only those we described in Table 1 and 2) are split from the original word. A comparable work was done by (Diab et al., 2004), where a POS tagging method for Arabic is also discussed. As we have access to this tool, we test its impact on the performance of our translation system. In

Table 1: Prefixes handled in this work and their meanings.

Prefix	و	ف	ك	ل	ب	ال
Transliteration	*w*	*f*	*k*	*l*	*b*	*Al*
Meaning	and	and then	as, like	in order to	with, in	the

(Habash and Rambow, 2005) a morphology analyzer was used for the segmentation and POS tagging. In contrast to the methods mentioned above, our segmentation method is unsupervised and rule based.

In this paper we first explain our statistical machine translation (SMT) system used for testing the impact of the different segmentation methods, then we introduce some preprocessing and normalization tools for Arabic and explain the linguistic motivation beyond them. Afterwards, we present three word segmentation methods, a supervised learning approach, a finite state automaton-based segmentation, and a frequency-based method. In Section 5, the experimental results are presented. Finally, the paper is summarized in Section 6 .

2 Baseline SMT System

In statistical machine translation, we are given a source language sentence $f_1^J = f_1 \ldots f_j \ldots f_J$, which is to be translated into a target language sentence $e_1^I = e_1 \ldots e_i \ldots e_I$. Among all possible target language sentences, we will choose the sentence with the highest probability:

$$\hat{e}_1^{\hat{I}} = \operatorname*{argmax}_{I,e_1^I} \left\{ Pr(e_1^I | f_1^J) \right\} \quad (1)$$

The posterior probability $Pr(e_1^I|f_1^J)$ is modeled directly using a log-linear combination of several models (Och and Ney, 2002):

$$Pr(e_1^I|f_1^J) = \frac{\exp\left(\sum_{m=1}^{M} \lambda_m h_m(e_1^I, f_1^J)\right)}{\sum\limits_{e'^{I'}_1} \exp\left(\sum_{m=1}^{M} \lambda_m h_m(e'^{I'}_1, f_1^J)\right)} \quad (2)$$

The denominator represents a normalization factor that depends only on the source sentence f_1^J. Therefore, we can omit it during the search process. As a decision rule, we obtain:

$$\hat{e}_1^{\hat{I}} = \operatorname*{argmax}_{I,e_1^I} \left\{ \sum_{m=1}^{M} \lambda_m h_m(e_1^I, f_1^J) \right\} \quad (3)$$

This approach is a generalization of the source-channel approach (Brown et al., 1990). It has the advantage that additional models $h(\cdot)$ can be easily integrated into the overall system. The model scaling factors λ_1^M are trained with respect to the final translation quality measured by an error criterion (Och, 2003).

We use a state-of-the-art phrase-based translation system including the following models: an n-gram language model, a phrase translation model and a word-based lexicon model. The latter two models are used for both directions: $p(f|e)$ and $p(e|f)$. Additionally, we use a word penalty and a phrase penalty. More details about the baseline system can be found in (Zens and Ney, 2004; Zens et al., 2005).

3 Preprocessing and Normalization Tools

3.1 Tokenizer

As for other languages, the corpora must be first tokenized. Here words and punctuation marks (except of abbreviations) must be separated. Another criterion is that Arabic has some characters that appear only at the end of a word. We use this criterion to separate words that are wrongly attached to each other.

3.2 Normalization and Simplification

The Arabic written language does not contain vowels, instead diacritics are used to define the pronunciation of a word, where a diacritic is written under or above each character in the word. Usually these diacritics are omitted, which increases the ambiguity of a word. In this case, resolving the ambiguity of a word is only dependent on the context. Sometimes, the authors write a diacritic on a word to help the reader and give him a hint which word is really meant. As a result, a single word with the same meaning can be written in different ways. For example *$Eb* (شعب) can be read[1] as *sha'ab* (Eng. nation) or *sho'ab* (Eng. options). If the author wants to give

[1] There are other possible pronunciations for the word *$Eb* than the two mentioned.

Table 2: Suffixes handled in this work and their meanings.

Suffix	ي	ني	ك	كما، كم، كن
Transliteration	*y*	*ny*	*k*	*kmA, km, kn*
Meaning	my	me	you, your (sing.)	you, your (pl.)
Suffix	نا	ه	ها	هما، هم، هن
Transliteration	*nA*	*h*	*hA*	*hmA, hm, hn*
Meaning	us, our	his, him	her	them, their

the reader a hint that the second word is meant, he can write *$uEb* (شُعب) or *$uEab* (شُعَب). To avoid this problem we normalize the text by removing all diacritics.

After segmenting the text, the size of the sentences increases rapidly, where the number of the stripped article *Al* is very high. Not every article in an Arabic sentence matches to an article in the target language. One of the reasons is that the adjective in Arabic gets an article if the word it describes is definite. So, if a word has the prefix *Al*, then its adjective will also have *Al* as a prefix. In order to reduce the sentence size we decide to remove all these articles that are supposed to be attached to an adjective. Another way for determiner deletion is described in (Lee, 2004).

4 Word Segmentation

One way to simplify inflected Arabic text for a SMT system is to split the words in prefixes, stem and suffixes. In (Lee et al., 2003), (Diab et al., 2004) and (Habash and Rambow, 2005) three supervised segmentation methods are introduced. However, in these works the impact of the segmentation on the translation quality is not studied. In the next subsections we will shortly describe the method of (Diab et al., 2004). Then we present our unsupervised methods.

4.1 Supervised Learning Approach (SL)

(Diab et al., 2004) propose solutions to word segmentation and POS Tagging of Arabic text. For the purpose of training the Arabic TreeBank is used, which is an Arabic corpus containing news articles of the newswire agency AFP. In the first step the text must be transliterated to the Buckwalter transliteration, which is a one-to-one mapping to ASCII characters. In the second step it will be segmented and tokenized. In the third step a partial lemmatization is done. Finally a POS tagging is performed. We will test the impact of the step 3 (segmentation + lemmatization) on the translation quality using our phrase based system described in Section 2.

4.2 Frequency-Based Approach (FB)

We provide a set of all prefixes and suffixes and their possible combinations. Based on this set, we may have different splitting points for a given compound word. We decide whether and where to split the composite word based on the frequency of different resulting stems and on the frequency of the compound word, e.g. if the compound word has a higher frequency than all possible stems, it will not be split. This simple heuristic harmonizes the corpus by reducing the size of vocabulary, singletons and also unseen words from the test corpus. This method is very similar to the method used for splitting German compound words (Koehn and Knight, 2003).

4.3 Finite State Automaton-Based Approach (FSA)

To segment Arabic words into prefixes, stem and one suffix, we implemented two finite state automata. One for stripping the prefixes and the other for the suffixes. Then, we append the suffix automaton to the other one for stripping prefixes. Figure 1 shows the finite state automaton for stripping all possible prefix combinations. We add the prefix *s* (س), which changes the verb tense to the future, to the set of prefixes which must be stripped (see table 1). This prefix can only be combined with *w* and *f*. Our motivation is that the future tense in English is built by adding the separate word "will".
The automaton showed in Figure 1 consists of the following states:

- S: the starting point of the automaton.

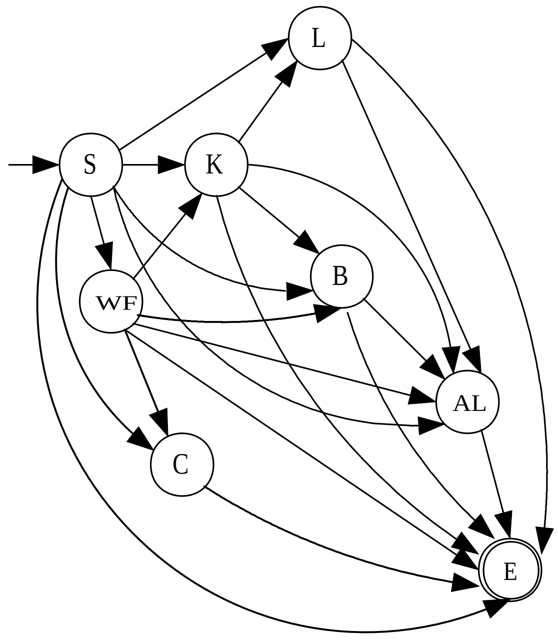

Figure 1: Finite state automaton for stripping prefixes off Arabic words.

- E: the end state, which can only be achieved if the resulting stem exists already in the text.
- WF: is achieved if the word begins with *w* or *f*.
- And the states C, K, L, B and AL are achieved if the word begins with *s*, *k*, *l*, *b* and *Al*, respectively.

To minimize the number of wrong segmentations, we restricted the transition from one state to the other to the condition that the produced stem occurs at least one time in the corpus. To ensure that most compound words are recognized and segmented, we run the segmenter iteratively, where after each iteration the newly generated words are added to the vocabulary. This will enable recognizing new compound words in the next iteration. Experiments showed that running the segmenter twice is sufficient and in higher iterations most of the added segmentations are wrong.

4.4 Improved Finite State Automaton-Based Approach (IFSA)

Although we restricted the finite state segmenter in such a way that words will be segmented only if the yielded stem already exists in the corpus, we still get some wrongly segmented words. Thus, some new stems, which do not make sense in Arabic, occur in the segmented text. Another problem is that the finite state segmenter does not care about ambiguities and splits everything it recognizes. For example let us examine the word *frd* (فرد). In one case, the character *f* is an original one and therefore can not be segmented. In this case the word means "person". In the other case, the word can be segmented to "*f rd*" (which means "and then he answers" or "and then an answer"). If the words *Alfrd*, *frd* and *rd*(الفرد ، فرد and رد) occur in the corpus, then the finite state segmenter will transform the *Alfrd* (which means "the person") to *Al f rd* (which can be translated to "the and then he answers"). Thus the meaning of the original word is distorted. To solve all these problems, we improved the last approach in a way that prefixes and suffixes are recognized simultaneously. The segmentation of the ambiguous word will be avoided. In doing that, we intend to postpone resolving such ambiguities to our SMT system.

The question now is how can we avoid the segmentation of ambiguous words. To do this, it is sufficient to find a word that contains the prefix as an original character. In the last example the word *Alfrd* contains the prefix *f* as an original character and therefore only *Al* can be stripped off the word. The next question we can ask is, how can we decide if a character belongs to the word or is a prefix. We can extract this information using the invalid prefix combinations. For example *Al* is always the last prefix that can occur. Therefore all characters that occur in a word after *Al* are original characters. This method can be applied for all invalid combinations to extract new rules to decide whether a character in a word is an original one or not.

On the other side, all suffixes we handle in this work are pronouns. Therefore it is not possible to combine them as a suffix. We use this fact to make a decision whether the end characters in a word are original or can be stripped. For example the word *trkhm* (تركهم) means "he lets them". If we suppose that *hm* is a suffix and therefore must be stripped, then we can conclude that *k* is an original character and not a suffix. In this way we are able to extract from the corpus itself decisions whether and how a word can be segmented.

In order to implement these changes the original automaton was modified. Instead of splitting a word we mark it with some properties which corespond to the states traversed until the end state. On the other side, we use the technique described above to generate negative properties which avoid the corresponding kind of splitting. If a property and its negation belong to the same word then the property is removed and only the negation is considered. At the end each word is split corresponding to the properties it is marked with.

5 Experimental Results

5.1 Corpus Statistics

The experiments were carried out on two tasks: the corpora of the Arabic-English NIST task, which contain news articles and UN reports, and the Arabic-English corpus of the Basic Travel Expression Corpus (BTEC) task, which consists of typical travel domain phrases (Takezawa et al., 2002). The corpus statistics of the NIST and BTEC corpora are shown in Table 3 and 5. The statistics of the news part of NIST corpus, consisting of the Ummah, ATB, ANEWS1 and eTIRR corpora, is shown in Table 4. In the NIST task, we make use of the NIST 2002 evaluation set as a development set and NIST 2004 evaluation set as a test set. Because the test set contains four references for each sentence we decided to use only the first four references of the development set for the optimization and evaluation. In the BTEC task, C-Star'03 and IWSLT'04 copora are considered as development and test sets, respectively.

5.2 Evaluation Metrics

The commonly used criteria to evaluate the translation results in the machine translation community are: WER (word error rate), PER (position-independent word error rate), BLEU (Papineni et al., 2002), and NIST (Doddington, 2002). The four criteria are computed with respect to multiple references. The number of reference translations per source sentence varies from 4 to 16 references. The evaluation is case-insensitive for BTEC and case-sensitive for NIST task. As the BLEU and NIST scores measure accuracy, higher scores are better.

5.3 Translation Results

To study the impact of different segmentation methods on the translation quality, we apply different word segmentation methods to the Arabic part of the BTEC and NIST corpora. Then, we make use of the phrase-based machine translation system to translate the development and test sets for each task.

First, we discuss the experimental results on the BTEC task. In Table 6, the translation results on the BTEC corpus are shown. The first row of the table is the baseline system where none of the segmentation methods is used. All segmentation methods improve the baseline system, except the SL segmentation method on the development corpus. The best performing segmentation method is IFSA which generates the best translation results based on all evaluation criteria, and it is consistent over both development and evaluation sets. As we see, the segmentation of Arabic words has a noticeable impact in improving the translation quality on a small corpus.

To study the impact of word segmentation methods on a large task, we conduct two sets of experiments on the NIST task using two different amounts of the training corpus: only news corpora, and full corpus. In Table 7, the translation results on the NIST task are shown when just the news corpora were used to train the machine translation models. As the results show, except for the FB method, all segmentation methods improve the baseline system. For the NIST task, the SL method outperforms the other segmentation methods, while it did not achieve good results when comparing to the other methods in the BTEC task.

We see that the SL, FSA and IFSA segmentation methods consistently improve the translation results in the BTEC and NIST tasks, but the FB method failed on the NIST task, which has a larger training corpus . The next step is to study the impact of the segmentation methods on a very large task, the NIST full corpus. Unfortunately, the SL method failed on segmenting the large UN corpus, due to the large processing time that it needs. Due to the negative results of the FB method on the NIST news corpora, and very similar results for FSA and IFSA, we were interested to test the impact of IFSA on the NIST full corpus. In Table 8, the translation results of the baseline system and IFSA segmentation method for the NIST full corpus are depicted. As it is shown in table, the IFSA method slightly improves the translation results in the development and test sets.

The IFSA segmentation method generates the best results among our proposed methods. It acheives consistent improvements in all three tasks over the baseline system. It also outperforms the SL

Table 3: BTEC corpus statistics, where the Arabic part is tokenized and segmented with the SL, FB, FSA and the IFSA methods.

		Arabic					English
		TOKENIZED	SL	FB	FSA	IFSA	
Train:	Sentences	20K					
	Running Words	159K	176.2K	185.5K	190.3K	189.1K	189K
	Vocabulary	18,149	14,321	11,235	11,736	12,874	7,162
Dev:	Sentences	506					
	Running Words	3,161	3,421	3,549	3,759	3,715	5,005
	OOVs (Running Words)	163	129	149	98	118	NA
Test:	Sentences	500					
	Running Words	3,240	3,578	3,675	3,813	3,778	4,986
	OOVs (Running Words)	186	120	156	92	115	NA

Table 4: Corpus statistics for the news part of the NIST task, where the Arabic part is tokenized and segmented with SL, FB, FSA and IFSA methods.

		Arabic					English
		TOKENIZED	SL	FB	FSA	IFSA	
Train:	Sentences	284.9K					
	Running Words	8.9M	9.7M	12.2M	10.9M	10.9M	10.2M
	Vocabulary	118.7K	90.5K	43.1K	68.4K	62.2K	56.1K
Dev:	Sentences	1,043					
	Running Words	27.7K	29.1K	37.3K	34.4K	33.5K	33K
	OOVs (Running Words)	714	558	396	515	486	NA
Test:	Sentences	1,353					
	Running Words	37.9K	41.7K	52.6K	48.6K	48.3K	48.3K
	OOVs (Running Words)	1,298	1,027	612	806	660	NA

segmentation on the BTEC task.

Although the SL method outperforms the IFSA method on the NIST tasks, the IFSA segmentation method has a few notable advantages over the SL system. First, it is consistent in improving the baseline system over the three tasks. But, the SL method failed in improving the BTEC development corpus. Second, it is fast and robust, and capable of being applied to the large corpora. Finally, it employs an unsupervised learning method, therefore can easily cope with a new task or corpus.

We observe that the relative improvement over the baseline system is decreased by increasing the size of the training corpus. This is a natural effect of increasing the size of the training corpus. As the larger corpus provides higher probability to have more samples per word, this means higher chance to learn the translation of a word in different contexts. Therefore, larger training corpus makes a better translation system, i.e. a better baseline, then it would be harder to outperform this better system. Using the same reasoning, we can realize why the FB method achieves good results on the BTEC task, but not on the NIST task. By increasing the size of the training corpus, the FB method tends to segment words more than the IFSA method. This over-segmentation can be compensated by using longer phrases during the translation, in order to consider the same context compared to the non-segmented corpus. Then, it would be harder for a phrase-based machine translation system to learn the translation of a word (stem) in different contexts.

6 Conclusion

We presented three methods to segment Arabic words: a supervised learning approach, a frequency-

Table 5: NIST task corpus statistics, where the Arabic part is tokenized and segmented with the IFSA method.

		ARABIC		ENGLISH
		TOKENIZED	IFSA	
Train:	Sentences	8.5M		
	Running Words	260.5M	316.8M	279.2M
	Vocabulary	510.3K	411.2K	301.2K
Dev:	Sentences	1043		
	Running Words	30.2K	33.3K	33K
	OOVs (Running Words)	809	399	NA
Test:	Sentences	1353		
	Running Words	40K	47.9K	48.3K
	OOVs (Running Words)	871	505	NA

Table 6: Case insensitive evaluation results for translating the development and test data of BTEC task after performing divers preprocessing.

	Dev				Test			
	mPER [%]	mWER [%]	BLEU [%]	NIST	mPER [%]	mWER [%]	BLEU [%]	NIST
Non-Segmented Data	21.4	24.6	63.9	10.0	23.5	27.2	58.1	9.6
SL Segmenter	21.2	24.4	62.5	9.7	23.4	27.4	59.2	9.7
FB Segmenter	20.9	24.4	65.3	10.1	22.1	25.8	59.8	9.7
FSA Segmenter	20.1	23.4	64.8	10.2	21.1	25.2	61.3	10.2
IFSA Segmenter	20.0	23.3	65.0	10.4	21.2	25.3	61.3	10.2

based approach and a finite state automaton-based approach. We explained that the best of our proposed methods, the improved finite state automaton, has three advantages over the state-of-the-art Arabic word segmentation method (Diab, 2000), supervised learning. They are: consistency in improving the baselines system over different tasks, its capability to be efficiently applied on the large corpora, and its ability to cope with different tasks.

7 Acknowledgment

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-06-C-0023.

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Defense Advanced Research Projects Agency (DARPA).

References

P. F. Brown, J. Cocke, S. A. Della Pietra, V. J. Della Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer, and P. S. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85, June.

M. Diab, K. Hacioglu, and D. Jurafsky. 2004. Automatic tagging of arabic text: From raw text to base phrase chunks. In D. M. Susan Dumais and S. Roukos, editors, *HLT-NAACL 2004: Short Papers*, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

M. Diab. 2000. An unsupervised method for multilingual word sense tagging using parallel corpora: A preliminary investigation. In *ACL-2000 Workshop on Word Senses and Multilinguality*, pages 1–9, Hong Kong, October.

G. Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proc. ARPA Workshop on Human Language Technology*.

Table 7: Case sensitive evaluation results for translating the development and test data of the news part of the NIST task after performing divers preprocessing.

	Dev				Test			
	mPER [%]	mWER [%]	BLEU [%]	NIST	mPER [%]	mWER [%]	BLEU [%]	NIST
Non-Segmented Data	43.7	56.4	43.6	9.9	46.1	58.0	37.4	9.1
SL Segmenter	42.0	54.7	45.1	10.2	44.3	56.3	39.9	9.6
FB Segmenter	43.4	56.1	43.2	9.8	45.6	57.8	37.2	9.2
FSA Segmenter	42.9	55.7	43.7	9.9	44.8	56.9	38.7	9.4
IFSA Segmenter	42.6	55.0	44.6	9.9	44.5	56.6	38.8	9.4

Table 8: Case-sensitive evaluation results for translating development and test data of NIST task.

	Dev				Test			
	mPER [%]	mWER [%]	BLEU [%]	NIST	mPER [%]	mWER [%]	BLEU [%]	NIST
Non-Segmented Data	41.5	53.5	46.4	10.3	42.5	53.9	42.6	10.0
IFSA Segmenter	41.1	53.2	46.7	10.2	42.1	53.6	43.4	10.1

N. Habash and O. Rambow. 2005. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proc. of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 573–580, Ann Arbor, Michigan, June. Association for Computational Linguistics.

P. Koehn and K. Knight. 2003. Empirical methods for compound splitting. In *Proc. 10th Conf. of the Europ. Chapter of the Assoc. for Computational Linguistics (EACL)*, pages 347–354, Budapest, Hungary, April.

L. S. Larkey, L. Ballesteros, and M. E. Connell. 2002. Improving stemming for arabic information retrieval: light stemming and co-occurrence analysis. In *Proc. of the 25th annual of the international Association for Computing Machinery Special Interest Group on Information Retrieval (ACM SIGIR)*, pages 275–282, New York, NY, USA. ACM Press.

Y. S. Lee, K. Papineni, S. Roukos, O. Emam, and H. Hassan. 2003. Language model based Arabic word segmentation. In E. Hinrichs and D. Roth, editors, *Proc. of the 41st Annual Meeting of the Association for Computational Linguistics*.

Y. S. Lee. 2004. Morphological analysis for statistical machine translation. In D. M. Susan Dumais and S. Roukos, editors, *HLT-NAACL 2004: Short Papers*, pages 57–60, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

F. J. Och and H. Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 295–302, Philadelphia, PA, July.

F. J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of the 41th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 160–167, Sapporo, Japan, July.

K. Papineni, S. Roukos, T. Ward, and W. J. Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318, Philadelphia, PA, July.

T. Takezawa, E. Sumita, F. Sugaya, H. Yamamoto, and S. Yamamoto. 2002. Toward a broad-coverage bilingual corpus for speech translation of travel conversations in the real world. In *Proc. of the Third Int. Conf. on Language Resources and Evaluation (LREC)*, pages 147–152, Las Palmas, Spain, May.

R. Zens and H. Ney. 2004. Improvements in phrase-based statistical machine translation. In *Proc. of the Human Language Technology Conf. (HLT-NAACL)*, pages 257–264, Boston, MA, May.

R. Zens, O. Bender, S. Hasan, S. Khadivi, E. Matusov, J. Xu, Y. Zhang, and H. Ney. 2005. The RWTH phrase-based statistical machine translation system. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*, pages 155–162, Pittsburgh, PA, October.