Unsupervised Tokenization for Machine Translation

Tagyoung Chung and **Daniel Gildea**
Computer Science Department
University of Rochester
Rochester, NY 14627

Abstract

Training a statistical machine translation starts with tokenizing a parallel corpus. Some languages such as Chinese do not incorporate spacing in their writing system, which creates a challenge for tokenization. Moreover, morphologically rich languages such as Korean present an even bigger challenge, since optimal token boundaries for machine translation in these languages are often unclear. Both rule-based solutions and statistical solutions are currently used. In this paper, we present unsupervised methods to solve tokenization problem. Our methods incorporate information available from parallel corpus to determine a good tokenization for machine translation.

1 Introduction

Tokenizing a parallel corpus is usually the first step of training a statistical machine translation system. With languages such as Chinese, which has no spaces in its writing system, the main challenge is to segment sentences into appropriate tokens. With languages such as Korean and Hungarian, although the writing systems of both languages incorporate spaces between "words", the granularity is too coarse compared with languages such as English. A single word in these languages is composed of several morphemes, which often correspond to separate words in English. These languages also form compound nouns more freely. Ideally, we want to find segmentations for source and target languages that create a one-to-one mapping of words. However, this is not always straightforward for two major reasons. First, what the optimal tokenization for machine translation should be is not always clear. Zhang et al. (2008b) and Chang et al. (2008) show that getting the tokenization of one of the languages in the corpus close to a gold standard does not necessarily help with building better machine translation systems. Second, even statistical methods require hand-annotated training data, which means that in resource-poor languages, good tokenization is hard to achieve.

In this paper, we explore unsupervised methods for tokenization, with the goal of automatically finding an appropriate tokenization for machine translation. We compare methods that have access to parallel corpora to methods that are trained solely using data from the source language. Unsupervised monolingual segmentation has been studied as a model of language acquisition (Goldwater et al., 2006), and as model of learning morphology in European languages (Goldsmith, 2001). Unsupervised segmentation using bilingual data has been attempted for finding new translation pairs (Kikui and Yamamoto, 2002), and for finding good segmentation for Chinese in machine translation using Gibbs sampling (Xu et al., 2008). In this paper, further investigate the use of bilingual information to find tokenizations tailored for machine translation. We find a benefit not only for segmentation of languages with no space in the writing system (such as Chinese), but also for the smaller-scale tokenization problem of normalizing between languages that include more or less information in a "word" as defined by the writing system, using Korean-English for our experiments. Here too, we find a benefit from using bilingual information, with unsupervised segmentation rivaling and in some cases surpassing supervised segmentation. On the modeling side, we use dynamic programming-based variational Bayes, making Gibbs sampling unnecessary. We also develop and compare various factors in the model to control the length of the tokens learned, and find a benefit from adjusting these parameters directly to optimize the end-to-end translation quality.

2 Tokenization

Tokenization is breaking down text into lexemes — a unit of morphological analysis. For relatively isolating languages such as English and Chinese, a word generally equals a single token, which is usually a clearly identifiable unit. English, especially, incorporates spaces between words in its writing system, which makes tokenization in English usually trivial. The Chinese writing system does not have spaces between words, but there is less ambiguity where word boundaries lie in a given sentence compared to more agglutinative languages. In languages such as Hungarian, Japanese, and Korean, what constitutes an optimal token boundary is more ambiguous. While two tokens are usually considered two separate words in English, this may be not be the case in agglutinative languages. Although what is considered a single morphological unit is different from language to language, if someone were given a task to align words between two languages, it is desirable to have one-to-one token mapping between two languages in order to have the optimal problem space. For machine translation, one token should not necessarily correspond to one morphological unit, but rather should reflect the morphological units and writing system of the other language involved in translation.

For example, consider a Korean "word" *meok-eoss-da*, which means *ate*. It is written as a single word in Korean but consists of three morphemes *eat-past-indicative*. If one uses morphological analysis as the basis for Korean tokenization, *meok-eoss-da* would be split into three tokens, which is not desirable if we are translating Korean to English, since English does not have these morphological counterparts. However, a Hungarian word *szekrényemben*, which means *in my closet*, consists of three morphemes *closet-my-inessive* that are distinct words in English. In this case, we do want our tokenizer to split this "word" into three morphemes *szekrény em ben*.

In this paper, we use segmentation and tokenization interchangeably as blanket terms to cover the two different problems we have presented here. The problem of segmenting Chinese sentences into words and the problem of segmenting Korean or Hungarian "words" into tokens of right granularity are different in their nature. However, our models presented in section 3 handle the both problems.

3 Models

We present two different methods for unsupervised tokenization. Both are essentially unigram tokenization models. In the first method, we try learning tokenization from word alignments with a model that bears resemblance to Hidden Markov models. We use IBM Model 1 (Brown et al., 1993) for the word alignment model. The second model is a relatively simpler monolingual tokenization model based on counts of substrings which serves as a baseline of unsupervised tokenization.

3.1 Learning tokenization from alignment

We use expectation maximization as our primary tools in learning tokenization form parallel text. Here, the observed data provided to the algorithm are the tokenized English string $\mathbf{e}_1^n$ and the untokenized string of foreign characters $\mathbf{c}_1^m$. The unobserved variables are both the word-level alignments between the two strings, and the tokenization of the foreign string. We represent the tokenization with a string $\mathbf{s}_1^m$ of binary variables, with $s_i = 1$ indicating that the ith character is the final character in a word. The string of foreign words $\mathbf{f}_1^\ell$ can be thought of as the result of applying the tokenization $\mathbf{s}$ to the character string $\mathbf{c}$:

$$\mathbf{f} = \mathbf{s} \circ \mathbf{c} \quad \text{where} \quad \ell = \sum_{i=1}^{m} s_i$$

We use IBM Model 1 as our word-level alignment model, following its assumptions that each foreign word is generated independently from one English word:

$$\begin{aligned} P(\mathbf{f}|\mathbf{e}) &= \sum_{\mathbf{a}} P(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) \\ &= \sum_{\mathbf{a}} \prod_{i} P(f_i \mid e_{a_i}) P(\mathbf{a}) \\ &= \prod_{i} \sum_{j} P(f_i \mid e_j) P(a_i = j) \end{aligned}$$

and that all word-level alignments $\mathbf{a}$ are equally likely: $P(a) = \frac{1}{n}$ for all positions. While Model 1 has a simple EM update rule to compute posteriors for the alignment variables $\mathbf{a}$ and from them learn the lexical translation parameters $P(f \mid e)$, we cannot apply it directly here because $\mathbf{f}$ itself is unknown, and ranges over an exponential number of possibilities depending on the hidden segmentation $\mathbf{s}$. This can be addressed by applying dynamic programing over the sequence $\mathbf{s}$. We compute the posterior probability of a word beginning at position i, ending at position j, and being generated by English word k:

$$\begin{aligned} &P(\mathbf{s}_{i \ldots j} = (1, 0, \ldots, 0, 1), a = k \mid \mathbf{e}) \\ &= \frac{\alpha(i) P(f \mid e_k) P(a = k) \beta(j)}{P(\mathbf{c} \mid \mathbf{e})} \end{aligned}$$

where $f = c_i \ldots c_j$ is the word formed by concatenating characters i through j, and a is a variable indicating which English position generated f. Here α and β are defined as:

$$\begin{aligned} \alpha(i) &= P(\mathbf{c}_1^i, s_i = 1 \mid \mathbf{e}) \\ \beta(j) &= P(\mathbf{c}_{j+1}^m, s_j = 1 \mid \mathbf{e}) \end{aligned}$$

These quantities resemble forward and backward probabilities of hidden Markov models, and can be computed with similar dynamic programming recursions:

$$\begin{aligned} \alpha(i) &= \sum_{\ell=1}^{L} \alpha(i - \ell) \sum_a P(a) P(c_{i-\ell}^i \mid e_a) \\ \beta(j) &= \sum_{\ell=1}^{L} \sum_a P(a) P(c_j^{j+\ell} \mid e_a) \beta(j + \ell) \end{aligned}$$

where L is the maximum character length for a word.

Then, we can calculate the expected counts of individual word pairs being aligned (c_i^j, e_k) by accumulating these posteriors over the data:

$$ec(c_i^j, e_k) \mathrel{+}= \frac{\alpha(i) P(a) P(c_i^j \mid e_k) \beta(j)}{\alpha(m)}$$

The M step simply normalizes the counts:

$$\tilde{P}(f \mid e) = \frac{ec(f, e)}{\sum_e ec(f, e)}$$

Our model can be compared to a hidden Markov model in the following way: a target word generates a source token which spans a zeroth order Markov chain of characters in source sentence, where a "transition" represents a segmentation and a "emission" represents an alignment. The model uses HMM-like dynamic programming to do inference. For the convenience, we refer to this model as the bilingual model in the rest of the paper. Figure 1 illustrates our first model with an small example. Under this model we are not learning segmentation directly, but rather we are learning alignments between two sentences. The segmentation is by-product of learning the alignment. We can find the optimal segmentation of a new source language sentence using the Viterbi algorithm. Given two sentences $\mathbf{e}$ and $\mathbf{f}$,

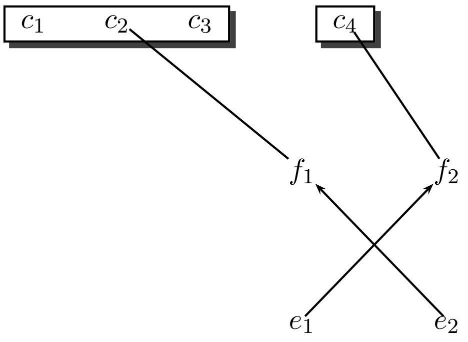

Figure 1: The figure shows a source sentence $\mathbf{f} = f_1, f_2 = \mathbf{s} \circ c_1 \ldots c_4$ where $\mathbf{s} = (0, 0, 1, 1)$ and a target sentence $\mathbf{e} = e_1, e_2$. There is a segmentation between c_3 and c_4; thus c_1, c_2, c_3 form f_1 and c_3 forms f_2. f_1 is generated by e_2 and f_2 is generated by e_1.

$$\mathbf{a}^* = \operatorname*{argmax}_{\mathbf{a}} P(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$$

and segmentation $\mathbf{s}^*$ implied by alignment $\mathbf{a}^*$ is the optimal segmentation of $\mathbf{f}$ found by this model.

3.2 Learning tokenization from substring counts

The second tokenization model we propose is much simpler. More sophisticated unsupervised monolingual tokenization models using hierarchical Bayesian models (Goldwater et al., 2006) and using the minimum description length principle (Goldsmith, 2001; de Marcken, 1996) have been studied. Our model is meant to serve as a computationally efficient baseline for unsupervised monolingual tokenization. Given a corpus of only source language of unknown tokenization, we want to find the optimal $\mathbf{s}$ given $\mathbf{c}$ — $\mathbf{s}$ that gives us the highest $P(\mathbf{s} \mid \mathbf{c})$. According to Bayes' rule,

$$P(\mathbf{s} \mid \mathbf{c}) \propto P(\mathbf{c} \mid \mathbf{s}) P(\mathbf{s})$$

Again, we assume that all $P(\mathbf{s})$ are equally likely. Let $\mathbf{f} = \mathbf{s} \circ \mathbf{c} = f_1 \ldots f_\ell$, where f_i is a word under some possible segmentation $\mathbf{s}$. We want to find the $\mathbf{s}$ that maximizes $P(\mathbf{f})$. We assume that

$$P(\mathbf{f}) = P(f_1) \times \ldots \times P(f_\ell)$$

To calculate $P(f_i)$, we count every possible substring — every possible segmentation of characters — from the sentences. We assume that

$$P(f_i) = \frac{count(f_i)}{\sum_k count(f_k)}$$

We can compute these counts by making a single pass through the corpus. As in the bilingual model, we limit the maximum size of f for practical reasons and to prevent our model from learning unnecessarily long f. With $P(f)$, given a sequence of characters $\mathbf{c}$, we can calculate the most likely segmentation using the Viterbi algorithm.

$$\mathbf{s}^* = \operatorname*{argmax}_{\mathbf{s}} P(\mathbf{f})$$

Our rationale for this model is that if a span of characters $f = c_i \dots c_j$ is an independent token, it will occur often enough in different contexts that such a span of characters will have higher probability than other spans of characters that are not meaningful. For the rest of the paper, this model will be referred to as the monolingual model.

3.3 Tokenizing new data

Since the monolingual tokenization only uses information from a monolingual corpus, tokenizing new data is not a problem. However, with the bilingual model, we are learning $P(f \mid e)$. We are relying on information available from **e** to get the best tokenization for **f**. However, the parallel sentences will not be available for new data we want to translate. Therefore, for the new data, we have to rely only on $P(f)$ to tokenize any new data, which can be obtained by calculating

$$P(f) = \sum_e P(f \mid e) P(e)$$

With $P(f)$ from the bilingual model, we can run the Viterbi algorithm in the same manner as monolingual tokenization model for monolingual data. We hypothesize that we can learn valuable information on which token boundaries are preferable in language f when creating a statistical machine translation system that translates from language f to language e.

4 Preventing overfitting

We introduce two more refinements to our word-alignment induced tokenization model and monolingual tokenization model. Since we are considering every possible token f that can be guessed from our corpus, the data is very sparse. For the bilingual model, we are also using the EM algorithm to learn $P(f \mid e)$, which means there is a danger of the EM algorithm memorizing the training data and thereby overfitting. We put a Dirichlet prior on our multinomial parameter for $P(f \mid e)$ to control this situation. For both models, we also want a way to control the distribution of token length after tokenization. We address this problem by adding a length factor to our models.

4.1 Variational Bayes

Beal (2003) and Johnson (2007) describe variational Bayes for hidden Markov model in detail, which can be directly applied to our bilingual model. With this Bayesian extension, the emission probability of our first model can be summarized as follows:

$$\theta_{\mathbf{e}} \mid \alpha \sim \mathrm{Dir}(\alpha),$$
$$f_i \mid e_i = e \sim \mathrm{Multi}(\theta_{\mathbf{e}}).$$

Johnson (2007) and Zhang et al. (2008a) show having small α helps to control overfitting. Following this, we set our Dirichlet prior to be as sparse as possible. It is set at $\alpha = 10^{-6}$, the number we used as floor of our probability.

For the model incorporating the length factor, which is described in the next section, we do not place a prior on our transition probability, since there are only two possible states, i.e. $P(s = 1)$ and $P(s = 0)$. This distribution is not as sparse as the emission probability.

Comparing variational Bayes to the traditional EM algorithm, the E step stays the same but the M step for calculating the emission probability changes as follows:

$$\tilde{P}(f \mid e) = \frac{\exp(\psi(ec(f,e) + \alpha))}{\exp(\psi(\sum_e ec(f,e) + s\alpha))}$$

where ψ is the digamma function, and s is the size of the vocabulary from which f is drawn. Since we do not accurately know s, we set s to be the number of all possible tokens. As can be seen from the equation, by setting α to a small value, we are discounting the expected count with help of the digamma function. Thus, having lower α leads to a sparser solution.

4.2 Token length

We now add a parameter that can adjust the tokenizer's preference for longer or shorter tokens.

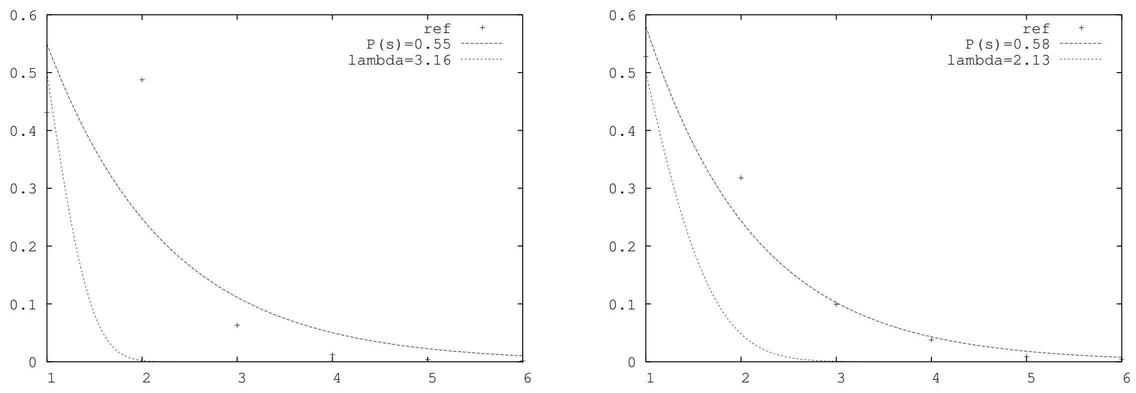

Figure 2: Distribution of token length for (from left to right) Chinese, and Korean. “ref” is the empirical distribution from supervised tokenization. Two length factors — ϕ_1 and ϕ_2 are also shown. For ϕ_1, the parameter to geometric distribution $P(s)$ is set to the value learned from our bilingual model. For ϕ_2, λ is set using the criterion described in the experiment section.

This parameter is beneficial because we want our distribution of token length after tokenization to resemble the real distribution of token length. This parameter is also useful because we also want to incorporate information on the number of tokens in the other language in the parallel corpus. This is based on the assumption that, if tokenization creates a one-to-one mapping, the number of tokens in both languages should be roughly the same. We can force the two languages to have about the same number of tokens by adjusting this parameter. The third reason is to further control overfitting. Our observation is that certain morphemes are very common, such that they will be always observed attached to other morphemes. For example, in Korean, a noun attached with nominative case marker is very common. Our model is likely to learn a noun attached with the morpheme — nominative case marker — rather than noun itself. This is not desirable when the noun occurs with less common morphemes; in these cases the morpheme will be split off creating inconsistencies.

We have experimented with two different length factors, each with one adjustable parameter:

$$\phi_1(\ell) = P(s)(1 - P(s))^{\ell - 1}$$
$$\phi_2(\ell) = 2^{-\ell^{\lambda}}$$

The first, ϕ_1, is the geometric distribution, where l is length of a token and $P(s)$ is probability of segmentation between two characters. The second length factor ϕ_2 was acquired through several experiments and was found to work well. As can been seen from Figure 2, the second factor discounts longer tokens more heavily than the geometric distribution. We can adjust the value of λ and $P(s)$ to increase or decrease number of tokens after segmentation.

For our monolingual model, incorporating these factors is straightforward. We assume that

$$P(\mathbf{f}) \propto P(f_1)\phi(\ell_1) \times \ldots \times P(f_n)\phi(\ell_n)$$

where ℓ_i is the length of f_i. Then, we use the same Viterbi algorithm to select the $f_1 \ldots f_n$ that maximizes $P(\mathbf{f})$, thereby selecting the optimal $\mathbf{s}$ according to our monolingual model with a length factor. We pick the value of λ and $P(s)$ that produces about the same number of tokens in the source side as in the target side, thereby incorporating some information about the target language.

For our bilingual model, we modify our model slightly to incorporate ϕ_1, creating a hybrid model. Now, our forward probability of forward-backward algorithm is:

$$\alpha(i) = \sum_{\ell=1}^{L} \alpha(i-l)\phi_1(\ell) \sum_a P(a)P(c_{i-\ell}^{i} \mid e_a)$$

and the expected count of (c_i^j, e_k) is

$$ec(c_i^j, e_k) \mathrel{+}= \frac{\alpha(i)P(a)P(c_i^j \mid e_k)\beta(j)\phi_1(j-i)}{\alpha(m)}$$

For ϕ_1, we can learn $P(s)$ for the geometric distribution from the model itself:[1]

$$P(s) = \frac{1}{m}\sum_{i}^{m} \frac{\alpha(i)\beta(i)}{\alpha(m)}$$

[1]The equation is for one sentence, but in practice, we sum over all sentences in the training data to calculate $P(s)$.

We can also fix $P(s)$ instead of learning it through EM. We incorporate ϕ_2 into the bilingual model as follows: after learning $P(f)$ from the bilingual model, we pick the λ in the same manner as the monolingual model and run the Viterbi algorithm.

After applying the length factor, what we have is a log-linear model for tokenization, with two feature functions with equal weights: the length factor and $P(f)$ learned from model.

5 Experiments

5.1 Data

We tested our tokenization methods on two different language pairs: Chinese-English, and Korean-English. For Chinese-English, we used FBIS newswire data. The Korean-English parallel data was collected from news websites and sentence-aligned using two different tools described by Moore (2002) and Melamed (1999). We used subsets of each parallel corpus consisting of about 2M words and 60K sentences on the English side. For our development set and test set, Chinese-English had about 1000 sentences each with 10 reference translations taken from the NIST 2002 MT evaluation. For Korean-English, 2200 sentence pairs were randomly sampled from the parallel corpus, and held out from the training data. These were divided in half and used for test set and development set respectively. For all language pairs, very minimal tokenization — splitting off punctuation — was done on the English side.

5.2 Experimental setup

We used Moses (Koehn et al., 2007) to train machine translation systems. Default parameters were used for all experiments except for the number of iterations for GIZA++ (Och and Ney, 2003). GIZA++ was run until the perplexity on development set stopped decreasing. For practical reasons, the maximum size of a token was set at three for Chinese, and four for Korean.[2] Minimum error rate training (Och, 2003) was run on each system afterwards and BLEU score (Papineni et al., 2002) was calculated on the test sets.

For the monolingual model, we tested two versions with the length factor ϕ_1, and ϕ_2. We picked λ and $P(s)$ so that the number of tokens on source side (Chinese, and Korean) will be about the same as the number of tokens in the target side (English).

For the bilingual model, as explained in the model section, we are learning $P(f \mid e)$, but only $P(f)$ is available for tokenizing any new data. We compared two conditions: using only the source data to tokenize the source language training data according to $P(f)$ (which is consistent with the conditions at test time), and using both the source and English data to tokenize the source language training data (which might produce better tokenization by using more information). For the first length factor ϕ_1, we ran an experiment where the model learns $P(s)$ as described in the model section, and we also had experiments where $P(s)$ was pre-set at 0.9, 0.7, 0.5, and 0.3 for comparison. We also ran an experiment with the second length factor ϕ_2 where λ was picked as the same manner as the monolingual model.

We varied tokenization of development set and test set to match the training data for each experiment. However, as we have implied in the previous paragraph, in the one experiment where $P(f \mid e)$ was used to segment training data, directly incorporating information from target corpus, tokenization for test and development set is not exactly consistent with tokenization of training corpus. Since we assume only source corpus is available at the test time, the test and the development set was tokenized only using information from $P(f)$.

We also trained MT systems using supervised tokenizations and tokenization requiring a minimal effort for the each language pair. For Chinese-English, the minimal effort tokenization is maximal tokenization where every Chinese character is segmented. Since a number of Chinese tokenizers are available, we have tried four different tokenizations for the supervised tokenizations. The first one is the LDC Chinese tokenizer available at the LDC website[3], which is compiled by Zhibiao Wu. The second tokenizer is a maxent-based tokenizer described by Xue (2003). The third and fourth tokenizations come from the CRF-based Stanford Chinese segmenter described by Chang et al. (2008). The difference between third and fourth tokenization comes from the different gold standard, the third one is based on Beijing University's segmentation (pku) and the fourth one is based on Chinese Treebank (ctb). For Korean-

[2]In the Korean writing system, one character is actually one syllable block. We do not decompose syllable blocks into individual consonants and vowels.

[3]http://projects.ldc.upenn.edu/Chinese/LDC_ch.htm

	Chinese		Korean
	BLEU	F-score	BLEU
Supervised			
Rule-based morphological analyzer			7.27
LDC segmenter	20.03	0.94	
Xue's segmenter	23.02	0.96	
Stanford segmenter (pku)	21.69	0.96	
Stanford segmenter (ctb)	22.45	1.00	
Unsupervised			
Splitting punctuation only			6.04
Maximal (Character-based MT)	20.32	0.75	
Bilingual $P(f \mid e)$ with ϕ_1 $P(s) = learned$	19.25		6.93
Bilingual $P(f)$ with ϕ_1 $P(s) = learned$	20.04	0.80	7.06
Bilingual $P(f)$ with ϕ_1 $P(s) = 0.9$	20.75	0.87	**7.46**
Bilingual $P(f)$ with ϕ_1 $P(s) = 0.7$	20.59	0.81	7.31
Bilingual $P(f)$ with ϕ_1 $P(s) = 0.5$	19.68	0.80	7.18
Bilingual $P(f)$ with ϕ_1 $P(s) = 0.3$	20.02	0.79	7.38
Bilingual $P(f)$ with ϕ_2	**22.31**	0.88	7.35
Monolingual $P(f)$ with ϕ_1	20.93	0.83	6.76
Monolingual $P(f)$ with ϕ_2	20.72	0.85	7.02

Table 1: BLEU score results for Chinese-English and Korean-English experiments and F-score of segmentation compared against Chinese Treebank standard. The highest unsupervised score is highlighted.

English, the minimal effort tokenization splitting off punctuation and otherwise respecting the spacing in the Korean writing system. A Korean morphological analysis tool[4] was used to create the supervised tokenization.

For Chinese-English, since a gold standard for Chinese segmentation is available, we ran an additional evaluation of tokenization from each methods we have tested. We tokenized the raw text of Chinese Treebank (Xia et al., 2000) using all of the methods (supervised/unsupervised) we have described in this section except for the bilingual tokenization using $P(f \mid e)$ because the English translation of the Chinese Treebank data was not available. We compared the result against the gold standard segmentation and calculated the F-score.

6 Results

Results from Chinese-English and Korean-English experiments are presented in Table 1. Note that nature of data and number of references are different for the two language pairs, and therefore the BLEU scores are not comparable. For both language pairs, our models perform equally well as supervised baselines, or even better. We can observe three things from the result. First, tokenization of training data using $P(f \mid e)$ tested on a test set tokenized with $P(f)$ performed worse than any other experiments. This affirms our belief that consistency in tokenization is important for machine translation, which was also mentioned by Chang et al. (2008). Secondly, we are learning valuable information by looking at the target language. Compare the result of the bilingual model with ϕ_2 as the length factor to the result of the monolingual model with the same length factor. The bilingual version consistently performed better than the monolingual model in all language pairs. This tells us we can learn better token boundaries by using information from the target language. Thirdly, our hypothesis on the need for heavy discount for longer tokens is confirmed. The value for $P(s)$ learned by the model was 0.55, and 0.58 for Chinese, and Korean respectively. For both language pairs, this accurately reflects the empirical distribution of token length, as can be seen in Figure 2. However, experiments where $P(s)$ was directly optimized performed better, indicating that this parameter should be optimized within the context of a complete system. The second length factor ϕ_2, which discounts longer tokens even more heavily, generally performed bet-

[4]http://nlp.kookmin.ac.kr/HAM/eng/main-e.html

English	the two presidents will hold a joint press conference at the end of their summit talks .
Untokenized Korean	양국 정상은 회담이 끝난 뒤 공동 기자회견을 갖고 회담 결과를 공식 발표한다 .
Supervised	양국 정상␣은 회담␣이 끝나␣ㄴ 뒤 공동 기자회견␣을 갖␣고 회담 결과␣를 공식 발표␣하␣ㄴ다 .
Bilingual $P(f \mid e)$ with ϕ_1	양국 정상은 회담␣이 끝난 뒤 공동 기자회견␣을 갖고 회담 결과␣를 공식 발표한␣다 .
Bilingual $P(f)$ with ϕ_2	양국 정상␣은 회담␣이 끝난 뒤 공동 기자회견␣을 갖고 회담 결과␣를 공식 발표␣한다 .
Monolingual $P(f)$ with ϕ_1	양국 정␣상␣은 회담␣이 끝난 뒤 공동 기자회견␣을 갖고 회담 결과를 공식 발표한␣다 .
Monolingual $P(f)$ with ϕ_2	양국 정상␣은 회담␣이 끝난 뒤 공동 기자회견␣을 갖고 회담 결과를 공식 발표␣한다 .

Figure 3: Sample tokenization results for Korean-English data. The underscores are added to clearly visualize where the breaks are.

ter than the first length factor when used in conjunction with the bilingual model. Lastly, F-scores of Chinese segmentations compared against the gold standard shows higher segmentation accuracy does not necessarily lead to higher BLEU score. F-scores presented in Table 1 are not directly comparable for all different experiments because the test data (Chinese Treebank) is used in training for some of the supervised segmenters, but these numbers do show how close unsupervised segmentations are to the gold standard. It is interesting to note that our highest unsupervised segmentation result does make use of bilingual information.

Sample tokenization results for Korean-English experiments are presented in Figure 3. We observe that different configurations produce different tokenizations, and the bilingual model produced generally better tokenizations for translation compared to the monolingual models or the supervised tokenizer. In this example, the tokenization obtained from the supervised tokenizer, although morphologically correct, is too fine-grained for the purpose of translation to English. For example, it correctly tokenized the attributive suffix ㄴ *-n* however, this is not desirable since English has no such counterpart. Both variations of the monolingual tokenization have errors such as incorrectly not segmenting 결과를 *gyeol-gwa-reul*, which is a compound of a noun and a case marker, into 결과␣를 *gyeol-gwa␣reul* as the bilingual model was able to do.

6.1 Conclusion and future work

We have shown that unsupervised tokenization for machine translation is feasible and can outperform rule-based methods that rely on lexical analysis, or supervised statistical segmentations. The approach can be applied both to morphological analysis of Korean and the segmentation of sentences into words for Chinese, which may at first glace appear to be quite different problems. We have only shown how our methods can be applied to one language of the pair, where one language is generally isolating and the other is generally synthetic. However, our methods could be extended to tokenization for both languages by iterating between languages. We also used the most simple word-alignment model, but more complex word alignment models could be incorporated into our bilingual model.

Acknowledgments This work was supported by NSF grants IIS-0546554 and ITR-0428020.

References

Matthew J. Beal. 2003. *Variational Algorithms for Approximate Bayesian Inference*. Ph.D. thesis, University College London.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Pi-Chuan Chang, Michel Galley, and Christopher Manning. 2008. Optimizing Chinese word segmentation for machine translation performance. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 224–232.

Carl de Marcken. 1996. Linguistic structure as composition and perturbation. In *Meeting of the Association for Computational Linguistics*, pages 335–341. Morgan Kaufmann Publishers.

John Goldsmith. 2001. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27(2):153–198.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2006. Contextual dependencies in unsupervised word segmentation. In *Proceedings of the International Conference on Computational Linguistics/Association for Computational Linguistics (COLING/ACL-06)*, pages 673–680.

Mark Johnson. 2007. Why doesn't EM find good HMM POS-taggers? In *2007 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 296–305, Prague, Czech Republic, June. Association for Computational Linguistics.

Genichiro Kikui and Hirofumi Yamamoto. 2002. Finding translation pairs from english-japanese untokenized aligned corpora. In *Proceedings of the 40th Annual Conference of the Association for Computational Linguistics (ACL-02) workshop on Speech-to-speech translation: algorithms and systems*, pages 23–30. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Conference of the Association for Computational Linguistics (ACL-07), Demonstration Session*, pages 177–180.

I. Dan Melamed. 1999. Bitext maps and alignment via pattern recognition. *Computational Linguistics*, 25:107–130.

Robert C. Moore. 2002. Fast and accurate sentence alignment of bilingual corpora. In *AMTA '02: Proceedings of the 5th Conference of the Association for Machine Translation in the Americas on Machine Translation: From Research to Real Users*, pages 135–144, London, UK. Springer-Verlag.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Franz Josef Och. 2003. Minimum error rate training for statistical machine translation. In *Proceedings of the 41th Annual Conference of the Association for Computational Linguistics (ACL-03).*

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Conference of the Association for Computational Linguistics (ACL-02).*

Fei Xia, Martha Palmer, Nianwen Xue, Mary Ellen Okurowski, John Kovarik, Shizhe Huang, Tony Kroch, and Mitch Marcus. 2000. Developing Guidelines and Ensuring Consistency for Chinese Text Annotation. In *Proc. of the 2nd International Conference on Language Resources and Evaluation (LREC-2000)*, Athens, Greece.

Jia Xu, Jianfeng Gao, Kristina Toutanova, and Hermann Ney. 2008. Bayesian semi-supervised chinese word segmentation for statistical machine translation. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 1017–1024, Manchester, UK, August. Coling 2008 Organizing Committee.

Nianwen Xue. 2003. Chinese word segmentation as character tagging. In *International Journal of Computational Linguistics and Chinese Language Processing*, volume 8, pages 29–48.

Hao Zhang, Chris Quirk, Robert C. Moore, and Daniel Gildea. 2008a. Bayesian learning of non-compositional phrases with synchronous parsing. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL-08)*, pages 97–105, Columbus, Ohio.

Ruiqiang Zhang, Keiji Yasuda, and Eiichiro Sumita. 2008b. Improved statistical machine translation by multiple Chinese word segmentation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 216–223.