Bayesian Learning of Tokenization for Machine Translation

Tagyoung Chung Daniel Gildea

The University of Rochester
Computer Science Department
Rochester, NY 14627

Technical Report 957

June 2010

Abstract

Training a statistical machine translation system 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. Morphologically rich languages such as Korean and Hungarian 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.

The University of Rochester Computer Science Department supported this work.
1 Introduction

Tokenizing a parallel corpus is usually the first step in 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 several 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 tokens. 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 spaces 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 and Hungarian-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. Preliminary experiments on Chinese-English and Korean-English appeared in our previous publication (Chung and Gildea, 2009), some of which are repeated here for the sake of completeness.

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-declarative. 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 so as to create one-to-one mappings between the tokens.

In this paper, we use segmentation and tokenization interchangeably as blanket terms to cover the two different problems we have presented here. The first is segmenting a sentence into words where there are no spaces between the words, which is the case with Chinese. The second is the problem of segmenting “words” in agglutinative languages into tokens of right granularity, which is the case with Hungarian and Korean. These problems are different in their nature. However, our models presented in Section 3 handle 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 for unsupervised tokenization.

3.1 Learning tokenization from alignment

We use expectation maximization as our primary tool in learning tokenization from parallel text. Here, the observed data provided to the algorithm are the tokenized English string \( e^n_1 \) and the untokenized string of foreign characters \( c^m \). 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 \( s^n_1 \) of binary variables, with \( s_i = 1 \) indicating that the \( i \)th character is the final character in a word. The string of foreign words \( f^\ell_1 \) can be thought of as the result of applying the tokenization \( s \) to the character string \( c \): \[
f = s \circ c \quad \text{where} \quad \ell = \sum_{i=1}^{m} s_i \]
Figure 1: The figure shows a source sentence \( f = f_1, f_2 = s \circ c_1 \ldots c_4 \) where \( s = (0, 0, 1, 1) \) and a target sentence \( e = e_1, e_2 \). 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 \( e_1 \).

Figure 1 illustrates the setup with a small example.

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:

\[
P(f | e) = \sum_a P(f, a | e)
= \sum_a \prod_i P(f_i | e_{a_i})P(a)
= \prod_i \sum_j P(f_i | e_j)P(a_i = j)
\]

and that all word-level alignments \( a \) 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 \( a \), from which it learns the lexical translation parameters \( P(f | e) \), we cannot apply it directly here because \( f \) itself is unknown, and ranges over an exponential number of possibilities depending on the hidden segmentation \( s \). This can be addressed by applying dynamic programming over the sequence \( s \). We compute the posterior probability of a word beginning at position \( i + 1 \), ending at position \( j \), and being generated by English word \( k \):

\[
P(s_{i...j} = (1, 0, \ldots, 0, 1), a = k | e) = \frac{\alpha(i)P(f | e_k)P(a = k)\beta(j)}{P(c | e)}
\]

where \( f = c_{i+1} \ldots c_j \) is the word formed by concatenating characters \( i + 1 \) through \( j \), and \( a \) is a variable indicating which English position generated \( f \). Here \( \alpha \) and \( \beta \) are defined as:

\[
\alpha(i) = P(c_i^i, s_i = 1 | e)
\beta(j) = P(c_j^m, s_j = 1 | e)
\]

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

\[
\alpha(i) = \sum_{\ell=1}^{L} \alpha(i-\ell) \sum_a P(a)P(c_{i-\ell}^i | e_a)
\]

\[
\beta(j) = \sum_{\ell=1}^{L} \sum_a P(a)P(c_{j+\ell}^j | e_a)\beta(j+\ell)
\]

where \(L\) is the maximum character length for a word.

Then, we can calculate the expected counts of individual word pairs being aligned \((c_{i+1}^j, e_k)\) by accumulating these posteriors over the data:

\[
ec(c_{i+1}^j, e_k) + = \frac{\alpha(i)P(a)P(c_{i+1}^j | e_k)\beta(j)}{\alpha(m)}
\]

The M step simply normalizes the counts:

\[
\tilde{P}(f | e) = \frac{ec(f, e)}{\sum_e ec(f, e)}
\]

Figure 2 shows a graphical representation of this inference procedure.

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 convenience, we refer to this model as the bilingual model in the rest of the paper. Under this model, we are not learning segmentation directly, but rather we are learning alignments between sentence pairs. 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 \(e\) and \(f\),

\[
a^* = \arg\max_a P(f, a | e)
\]

and segmentation \(s^*\) implied by alignment \(a^*\) is the optimal segmentation of \(f\) 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; Mochihashi et al., 2009) 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 text of unknown tokenization, we want to find the optimal \( s \) given \( c \) — \( s \) that gives us the highest \( P(s \mid c) \).

According to Bayes’ rule,

\[
P(s \mid c) \propto P(c \mid s)P(s)
\]

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

\[
P(f) = \prod_{i=1}^{\ell} P(f_i)
\]

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

\[
P(f_i) = \frac{\text{count}(f_i)}{\sum_k \text{count}(f_k)}
\]

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 \( c \), we can calculate the most likely segmentation using the Viterbi algorithm.

\[
s^* = \arg\max_s P(f)
\]

Our rationale for this model is that if a span of characters \( f = c_1 \ldots c_j \) 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 Learning tokenization from a monolingual corpus with EM

The parameter learning method we presented in Section 3.2 can be viewed as EM with very early stopping. Early stopping is a common way to deal with overfitting. We collect initial counts of possible tokens and normalize then stop. Regular EM would not work as inference since it will overfit — the maximum likelihood solution would treat an entire sentence as a single unique token. However, in Section 4, we address the issue of overfitting, and with the modification, EM may successfully learn proper token boundaries. Without regard to the overfitting issue, the E step would resemble the inference for bilingual model. The E step would be:

\[
e c(c_{i+1}) = \alpha(i)P(c_{i+1})\beta(j) \frac{\alpha(m)}{\alpha(m)}
\]
where $\alpha$ and $\beta$ are:

$$\alpha(i) = \sum_{\ell=1}^{L} \alpha(i - \ell) P(c_{i-\ell}^i)$$

$$\beta(j) = \sum_{\ell=1}^{L} P(c_{j+\ell}^j) \beta(j + \ell)$$

The M step would be a simple normalization:

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

After the modification stated in Section 4, we compare the two methods of parameter learning for the monolingual model.

### 3.4 Tokenizing new data

Since the monolingual tokenization models only use information from a monolingual corpus, tokenizing new data does not present a new 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 $e$.

### 3.5 Utilizing pre-existing word boundaries

As mentioned in Section 1 and Section 2, Hungarian has pre-existing word boundaries whose granularity is too coarse in comparison with English. Using this boundaries, we can add a bit more information to our segmentation models. We can annotate whether a certain span of characters have come from beginning, middle or end of a Hungarian word. For example, usually in suffix form, Hungarian particle -ben or -ban (depending on vowel harmony with preceding noun) indicates inessive case, which translates to English as in or at. If we explicitly mark these particles as originating from the end of a word rather than the middle or beginning, the model would be able to discriminate between these particles and other words that happen to contain these character spans in the middle or in the beginning. With this modification, a Hungarian word bank, which translates to English as bank has less chance of getting segmented into ban k, which could potentially happen since ban is a common character sequence. We have added this modification for Hungarian and tested it with a small data set by building a machine translation system. With this modification, we have achieved a
small gain in BLEU score. ¹ All the Hungarian-English results shown in later sections include this modification.

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 alleviate this situation. We do the same for the monolingual model when we use EM to learn parameters. 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 models. With this Bayesian extension, the emission probability of our first model can be summarized as follows:

\[
\theta_e \mid \alpha \sim \text{Dir}(\alpha), \\
f_i \mid e_i = e \sim \text{Multi}(\theta_e).
\]

Johnson (2007) and Zhang et al. (2008a) show having small \( \alpha \) helps to control overfitting. Following this, we set our Dirichlet prior to be as sparse as possible. It is set to the value 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:

\[
\hat{P}(f \mid e) = \frac{\exp(\psi(\sum_e ec(f,e) + \alpha))}{\exp(\psi(\sum_e ec(f,e) + \sigma \alpha))}
\]

where \( \psi \) is the digamma function, and \( \sigma \) is the size of the vocabulary from which \( f \) is drawn. Since we do not accurately know \( \sigma \), we set \( \sigma \) to be the number of all possible tokens, which changes with each iteration since possible tokens in the probability table gets pruned away when they fall below the floor of the probability. As can be seen from the equation, by setting \( \alpha \) to a small value, we are discounting the expected count with help of the digamma function. Thus, having lower \( \alpha \) leads to a sparser solution. The procedures are the same when using EM to learn parameters for the monolingual model.

¹The BLEU score went from 17.79 (without modification) to 17.97 (with modification) in one instance where bilingual model of tokenization was used in conjunction with the length factor \( \phi_2 \), which is explained in later sections. This score is not comparable to Hungarian-English translation results presented later sections because it uses different set of training and test data.
4.2 Token length

We now add a parameter that can adjust the tokenizer’s preference for longer or shorter tokens. 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 initially 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, \(\phi_1\), is the geometric distribution, where \(\ell\) is length of a token and \(P(s)\) is probability of segmentation between two characters. The geometric distribution is a simple logical choice if one assumes all segmentations to be independent events. The second length factor \(\phi_2\) was acquired through several experiments and was found to work well. As can been seen from Figure 3, the second factor discounts longer tokens more heavily than the geometric distribution. We can adjust the value of \(\lambda\) and \(P(s)\) to increase or decrease number of tokens after segmentation. We can see from the figure that although \(\phi_1\) and \(\phi_2\) generally follow the real token length distribution for Chinese and Korean, but they are quite divergent when it comes to Hungarian. Since Hungarian is alphabetic, the graph tapers off more slowly and peaks later than the other languages. Therefore, we have tested the third length factor, which is the Poisson distribution:

\[
\phi_3(\ell) = \frac{\lambda^\ell e^{-\lambda}}{\ell!}
\]

The Poisson distribution could have other advantages compare to the first two length factors. The distribution still penalizes very long tokens, but it also penalizes overly short tokens. Since we count every possible substrings and use early stopping for the EM, especially in case of Hungarian, every single character will be over-counted. The Poisson distribution may help the model to correct this situation.

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

\[
P(f) \propto \prod_{i=1}^{n} P(f_i)\phi(\ell_i)
\]

where \(\ell_i\) is the length of \(f_i\). Then, we use the same Viterbi algorithm to select the \(f_1 \ldots f_n\) that maximizes \(P(f)\), thereby selecting the optimal \(s\) according to our monolingual model with a length factor. For \(\phi_1\) and \(\phi_2\), we pick the value of \(\lambda\) and \(P(s)\) that produces about the same number of
“ref” is the empirical distribution from supervised tokenization. Three length factors — $\phi_1$, $\phi_2$, and $\phi_3$ are also shown. For $\phi_1$, the parameter to geometric distribution $P(s)$ is set to the value learned from our bilingual model. For $\phi_2$ and $\phi_3$, $\lambda$ is set using the criterion described in the experiment section. The Poisson distribution and the geometric distribution are discrete, but continuous versions are shown for comparison.
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 \( \phi_1 \), creating a hybrid model. Now, our forward probability of the forward-backward algorithm is:

\[
\alpha(i) = \sum_{\ell=1}^{L} \alpha(i-\ell) \phi_1(\ell) \sum_a P(a) P(c_{i-\ell}^i | e_a)
\]

and the expected count of \((c_{i+1}^j, e_k)\) is

\[
ce(c_{i+1}^j, e_k) = \frac{\alpha(i) P(a) P(c_{i+1}^j | e_k) \beta(j) \phi_1(j-i)}{\alpha(m)}
\]

For \( \phi_1 \), we can learn \( P(s) \) for the geometric distribution from the model itself:

\[
P(s) = \frac{1}{m} \sum_i \frac{\alpha(i) \beta(i)}{\alpha(m)}
\]

We can also fix \( P(s) \) throughout EM iterations instead of learning it through EM. We incorporate \( \phi_2 \) into the bilingual model as follows: after learning \( P(f) \) from the bilingual model, we pick the \( \lambda \) in the same manner as the monolingual model and run the Viterbi algorithm. The parameter value for \( \phi_3 \) was picked in the following manner: Since \( E[|f_i|] = \frac{1}{P(s)} \), and the Poisson distribution’s mean is \( \lambda \), the parameter was set to \( \lambda = \lceil \frac{1}{P(s)} \rceil \) where \( P(s) \) is the parameter learned through EM. The same parameter was applied to the monolingual model.

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.

There have been other works that incorporated the length factor in monolingual tokenization in implicit and explicit manner. Goldwater et al. (2006) used a geometric distribution as the base distribution for a Dirichlet process in their Bayesian segmentation model to model word acquisition in infants. Liang and Klein (2009) used the doubly exponential length factor in their word segmentation model to test their online EM algorithm. Mochihashi et al. (2009) used Poisson distribution as the base distribution for Pitman-Yor process in their Bayesian segmentation model. In the context of machine translation, Chang et al. (2008) did not directly use a length factor but used a feature in their CRF Chinese segmenter to adjust the average size of tokens to improve MT performance.

5 Experiments

5.1 Data

We tested our tokenization methods on three different language pairs: Chinese-English, Korean-English, and Hungarian-English. For Chinese-English, we used FBIS newswire data. The Korean-English parallel data was collected from news websites and sentence-aligned using two tools described by Moore (2002) and Melamed (1999). The Hunglish corpus (Varga et al., 2005) is used in

\(^2\)The equation is for one sentence, but in practice, we sum over all sentences in the training data to calculate \( P(s) \).
our Hungarian-English experiments, which mostly consists of various data from European Parliament, movie subtitles, and translated literature. We used subsets of each parallel corpus consisting of about 2M words 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 and Hungarian-English, 2200 sentence pairs were randomly selected 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). For each language pair, GIZA++ was run until the perplexity on development set stopped decreasing using our supervised baseline data and the same number of iterations were used for all experiments. For practical reasons, the maximum size of a token was set at three for Chinese, four for Korean, and ten for Hungarian. As one can easily guess, even with these limitations, the inference procedure consumes a large amount of memory. However, EM is easily parallelizable and most entries in the probability table (more than 99% of them) quickly get pruned away after the initial iteration. Minimum error rate training (Och, 2003) was run on each system afterwards and the BLEU score (Papineni et al., 2002) was calculated on the test sets.

For the monolingual model, we tested three versions with the length factor $\phi_1$, $\phi_2$, and $\phi_3$. For $\phi_1$ and $\phi_2$, we picked $\lambda$ and $P(s)$ so that the number of tokens on source side (Chinese, Korean, and Hungarian) will be about the same as the number of tokens in the target side (English). How we picked the parameter value for $\phi_3$ is explained in the Section 4.2.

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 $\phi_1$, 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 experiments with the second and third length factor $\phi_2$ and $\phi_3$ where $\lambda$s were picked in 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 the target corpus, tokenization for test and development set is not exactly consistent with tokenization of training corpus. Since we assume that only the source corpus is available at the test time, the test and the development set was tokenized only using information from $P(f)$.

---

3In the Korean writing system, one character is actually one syllable block. We do not decompose syllable blocks into individual consonants and vowels. There are anecdotal evidence that decomposition does not necessary help with the translation since most Korean particles do correspond to a syllable.

4Since one Korean character is a one syllable block, the restriction is comparable to the Korean one.
5.3 Baseline

We also trained MT systems using supervised tokenizations and tokenization requiring a minimal effort for each language pair as comparison. 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\(^5\), which is compiled by Zhibiao Wu. The second tokenizer is a maxent-based tokenizer (Xue, 2003). The third and fourth tokenizations come from the CRF-based Stanford Chinese segmenter (Chang et al., 2008). The difference between the third and fourth tokenizations 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-English, the minimal effort tokenization is splitting off punctuation and otherwise respecting the spacing in the Korean writing system. A Korean morphological analysis tool\(^6\) was used to create the supervised tokenization. For Hungarian-English, the minimal effort tokenization is the same with the Korean. A Hungarian lemmatization tool (Trón et al., 2005) 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 the 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 \(P(f | e)\) 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 and analysis

6.1 Analysis of BLEU score results

Results from Chinese-English and Korean-English experiments are presented in Table 1. Note that the nature of the data and the number of references are different for the three 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 four things from the result. First, tokenization of training data using \(P(f | e)\) 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 results of the bilingual model with \(\phi_2\) as the length factor to the results of the monolingual model (for both inference methods) with the same length factor.\(^7\) The bilingual version consistently performed better than the monolingual model in all language pairs regardless of the inference method. This tells us we can learn better token boundaries by using information from the target language. Thirdly, our hypothesis

\(^5\)http://projects.ldc.upenn.edu/Chinese/LDC_ch.htm
\(^6\)http://nlp.kookmin.ac.kr/HAM/eng/main-e.html
\(^7\)\(\phi_1\) may not be directly comparable since how we set the parameter for the length factor is not exactly the same as in the monolingual and the bilingual models whereas the parameter for \(\phi_2\) is set in the exactly the same way in the both models.
<table>
<thead>
<tr>
<th>Method</th>
<th>Chinese BLEU</th>
<th>Chinese F-score</th>
<th>Korean BLEU</th>
<th>Hungarian BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rule-based morphological analyzer</td>
<td>20.03</td>
<td>0.94</td>
<td>8.70</td>
<td>19.38</td>
</tr>
<tr>
<td>LDC segmenter</td>
<td>23.02</td>
<td>0.96</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Xue’s segmenter</td>
<td>21.69</td>
<td>0.96</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stanford segmenter (pku)</td>
<td>22.45</td>
<td>1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stanford segmenter (ctb)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unsupervised</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Splitting punctuation only</td>
<td>19.25</td>
<td>0.88</td>
<td>7.52</td>
<td>19.05</td>
</tr>
<tr>
<td>Maximal (Character-based MT)</td>
<td>20.04</td>
<td>0.80</td>
<td>9.54</td>
<td>14.42</td>
</tr>
<tr>
<td>Bilingual $P(f \mid e)$ with $\phi_1 P(s) = learned$</td>
<td>20.75</td>
<td>0.87</td>
<td>9.40</td>
<td>18.53</td>
</tr>
<tr>
<td>Bilingual $P(f)$ with $\phi_1 P(s) = 0.9$</td>
<td>20.59</td>
<td>0.81</td>
<td>9.75</td>
<td>19.08</td>
</tr>
<tr>
<td>Bilingual $P(f)$ with $\phi_1 P(s) = 0.7$</td>
<td>19.68</td>
<td>0.80</td>
<td>9.38</td>
<td>19.34</td>
</tr>
<tr>
<td>Bilingual $P(f)$ with $\phi_1 P(s) = 0.5$</td>
<td>20.02</td>
<td>0.79</td>
<td>9.50</td>
<td>19.05</td>
</tr>
<tr>
<td>Bilingual $P(f)$ with $\phi_2$</td>
<td>22.31</td>
<td>0.88</td>
<td>8.57</td>
<td>19.17</td>
</tr>
<tr>
<td>Bilingual $P(f)$ with $\phi_3$</td>
<td>20.25</td>
<td>0.84</td>
<td>9.44</td>
<td>19.41</td>
</tr>
<tr>
<td>Monolingual $P(f)$ with $\phi_1$</td>
<td>20.93</td>
<td>0.83</td>
<td>8.20</td>
<td>18.36</td>
</tr>
<tr>
<td>Monolingual $P(f)$ with $\phi_2$</td>
<td>20.72</td>
<td>0.85</td>
<td>8.68</td>
<td>18.38</td>
</tr>
<tr>
<td>Monolingual $P(f)$ with $\phi_3$</td>
<td>17.72</td>
<td>0.80</td>
<td>9.17</td>
<td>19.01</td>
</tr>
<tr>
<td>Monolingual EM $P(f)$ with $\phi_1$</td>
<td>15.70</td>
<td>0.78</td>
<td>9.10</td>
<td>19.04</td>
</tr>
<tr>
<td>Monolingual EM $P(f)$ with $\phi_2$</td>
<td>21.30</td>
<td>0.86</td>
<td>8.81</td>
<td>18.88</td>
</tr>
<tr>
<td>Monolingual EM $P(f)$ with $\phi_3$</td>
<td>20.40</td>
<td>0.84</td>
<td>9.28</td>
<td>18.85</td>
</tr>
</tbody>
</table>

Table 1: BLEU score results for Chinese-English, Korean-English, and Hungarian-English experiments and F-score of segmentation compared against Chinese Treebank standard. The highest unsupervised score is highlighted.
Figure 4: X-axis represents ratio between number of tokens in Chinese text and English text after tokenization. Y-axis represents BLEU score on test set with MT system trained with the tokenized parallel corpus. Weak correlation between two can be observed. On the need for heavy discounting for longer tokens is confirmed. The value for $P(s)$ learned by the model was 0.55, 0.58, and 0.38 for Chinese, Korean, and Hungarian respectively. For both language pairs, this generally reflects the empirical distribution of token length, as can be seen in Figure 3. 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, $\phi_2$, which discounts longer tokens even more heavily, generally performed better than the first length factor when used in conjunction with the bilingual model. However, results involving the Hungarian-English language pair slightly deviate from this observation; here $\phi_3$ performed the best. As can be seen from Figure 3, $\phi_3$ still heavily penalizes long tokens after a certain length, but peaks later than other length factors, which may reflect the differences in languages involved in the translations. For example, Chinese words are more likely to be monosyllabic than in the other languages. This indicates that selection of length factor should reflect general token length distribution of the language that is being tokenized. Lastly, the F-scores of Chinese segmentations compared against the gold standard show that 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.
6.2 Setting parameters for length factors

One purpose of the experiments was to find some intermediate measure other than BLEU that could be used to find a good tokenization. In the context of machine translation, better tokenization means that it would eventually produce superior translations. It would be convenient to have such a measure since training entire machine translation system to see which tokenization is better is an expensive exercise. We have examined various criteria and tried to see if optimizing the tokenization on one of the criteria would give us a better BLEU score in the end. The criteria included: perplexity of development set during GIZA++ training, size of vocabulary of both languages, similarity with supervised tokenization as measured in F-score and ratio of total number of tokens in the parallel corpus after tokenization. Out of these, the ratio between number of tokens after tokenization was the only criterion that showed any correlation with BLEU, albeit a weak one. Hence, we have used the criterion to pick the parameter of the length factors as explained in other sections, if it was not automatically learned from the model. Figure 4 shows the correlation between the ratio of number of tokens in Chinese-English corpus after tokenization and the BLEU score. All the results represent experiments with bilingual tokenization with different parameters for length factors.

6.3 Sample result

Sample tokenization results for Korean-English experiments are presented in Figure 5. 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 네-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 결과를 gyeol-gwa-reul as the bilingual model was able to do.
7 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 Hungarian. It is also applicable to segmentation of sentences into words for Chinese, which may at first glance appear to be quite different problems. We have also shown Variational Bayes and length factors could be effective means of controlling overfitting in tokenization. The results have confirmed that we can learn better tokenization from utilizing information from a parallel corpus rather than from a monolingual corpus. The results also showed that parameters involved in the tokenization should be optimized to improve end-to-end translation data rather than some monolingual gold standard.

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.

References


