Character based Chinese-English statistical machine translation

Jia Xu[†], Jianfeng Gao[*], Kristina Toutanova[*], Hermann Ney[†]

Computer Science 6[†]
RWTH Aachen University
D-52056 Aachen, Germany
{xujia,ney}@cs.rwth-aachen.de

Microsoft Corporation[*]
One Microsoft Way
Redmond, WA 98052, USA
{jfgao,kristout}@microsoft.com

Abstract

State-of-the-art machine translation is based on words. However, Chinese sentences are written in the form of a sequence of Chinese characters. Thus, word boundaries are detected using an off-the-shelf segmentation method before translation. Optimal words may be lost as the Chinese word segmentation and the translation process are separated. The segmentation does not only depend on the context but also on the language to be translated into.

Therefore, we build a translation system based on Chinese characters directly. Chinese words are identified implicitly in translation models. In the model training, word segmentations and alignments are learned simultaneously, new words are invented using the Dirichlet process, and word distributions are calculated by Gibbs sampling. In search of the best translation, segmentation alternatives are represented as a lattice so that the final decision is integrated into the processing of decoding.

Our translation results improve state-of-the-art Chinese-English translation systems on GALE and IWSLT tasks. Moreover, the proposed algorithms can be applied to statistical machine translation from Chinese to any other languages.

1 Introduction

In Chinese texts, sentences are written in the form of a sequence of Chinese characters, and words composed of single or multiple characters are not separated by delimiters. This is different from most European languages and poses a challenge in natural language processing tasks, such as machine translation. The conventional way is to segment the Chinese character sequence into Chinese "words". Finding proper word boundaries in a sequence of Chinese characters is the so-called *Chinese word segmentation* (CWS) problem.

The Chinese word segmentation performance is usually evaluated by precision, which is calculated based on how well the segmented text matches the reference text. However, in our experiments we observed that the correlation of word segmentation and translation error rates are not close enough. Therefore, we explore the idea that the best segmentation depends on the task and concentrate on developing a CWS method for machine translation. For machine translation, intuitively, the best Chinese words should be the units that provide the best word alignment and lead to the best translation performance.

The common solution has been to segment the Chinese text explicitly and to perform a standard training and translation once the words are fixed. There are many ways to recognize word boundaries. The simplest method is to use the **maximum matching**. Characters in a sequence are checked whether they match a word in the dictionary from left to right; firstly to the longer words then to the shorter words. This method is rather naive, because words with equal lengths are treated in the same way. Hence, statistical methods are introduced to estimate model parameters on the training data. Under various statistical methods, the **N-gram** based Chinese word segmentation is widely applied, such as in (Chen et al., 2005), (Wang and Liu, 2005), (LDC, 2003), serving as our baseline segmentation method. **HMM** is another statistical model to perform word segmentation. In (Zhang and Malik, 2003) features derived from name entity and lexicon etc. are considered. Bigram class dependency and word conditional probability are taken into account in the HMM based framework to search for optimal segmentation boundaries. **Maximum entropy** (Low et al., 2005) is a similar approach that combines user defined features and segmentation decisions. Single character words, characters in the middle, at the beginning and at the end of a word constitute four basic classes of the model. (Andrew, 2006) employed a conditional random field (**CRF**) model for sequence segmentation to include various information for a segmentation decision.

Nonetheless, significant improvements in translation performance have not yet been shown to result from using these more sophisticated CWS methods, due to the following two reasons: 1. The segmentations may be erroneous, because the

context varies. 2. The best segmentation for a given character sequence also depends on its translation. For a destined character sequence the 'correct' segmentation is not universal, but we need to consider the contexts and the language to be translated into. In standard approaches word segmentation is performed previously and independently on the translation system. Segmentation and alignment of words are two separate processes, though they actually influence each other.

The main characteristic of our Chinese word segmentation method is that our segmentation model is designed for and integrated into the machine translation system. We renovate the trivial approach segmenting words in the preprocessing but put the word segmentation into the word alignment training as well as decoding for the best translation. Translation on Chinese characters is feasible therefore as segmentation process is pushed to the translation.

There are two major problems to be solved for Chinese word segmentation: the first one is how to train the word segmentation model; the second one is how to perform the segmentation on a test corpus using the trained model.

For the model training, we will present two different approaches, both employing the bilingual information: the alignment derived segmentation in Section 4 and the semi-supervised CWS in Section 5. The previous one is easier to implement with an initialization of the alignment between Chinese characters and English words; the latter one is more refined with an initialization of an unigram word segmentation with LDC lexicon. These two methods are parallel. Although we can initialize the latter one with the previous one, we did not apply it for simplicity. In alignment derived segmentation a monolingual lexicon is extracted from single-best alignments of Chinese characters and English words, where bilingual and context information are employed. However, the single-best alignments may contain errors, thus we further refined the model into an semi-supervised method. Word segmentation and alignment are trained jointly considering both monolingual and bilingual information, in order to derive a segmentation suitable for machine translation, see (Xu et al., 2008). New word entries and their distributions are introduced automatically using Dirichlet process. Our experiments on both large (GALE) and small (IWSLT) data tracks show improvements over the state-of-the-art machine translation systems with respect to translation performance.

Once the segmentation model, more precisely the lexicon is obtained, we need to solve the second problem, finding best Chinese word segmentation on a test corpus. We can use either an unigram word segmenter (as standard approach) or the word segmentation lattice in Section 6 to perform Chinese word segmentation on the test corpus. With the lattice segmentation, translation is carried out on character level in decoding, and multiple segmentations are represented as a lattice so that segmentation decisions are integrated into the search for the best translation. Similar approaches were applied in speech translation, e.g. (Ney, 1999), where speech recognition and text translation are combined by using the recognition lattices. We also weight the different segmentations with a language model trained on the Chinese corpus at the word level. Weighting the word segmentation by language model cost was introduced in (Luo and Roukos, 1996). Our experiments on Chinese-to-English translation show that our method improves the performance of a state-of-the-art machine translation system.

In the following context, we will first introduce the definition of Chinese word segmentation in Section 2, along with the notations for later sections and the baseline approach ngram segmentation as in Section 3. As chief contents, the alignment derived segmentation and semi-supervised CWS method is described in Section 4 and 5 respectively, and the translation on segmentation lattices is discussed in Section 6. Finally, translation experiments are shown in Section 7.

2 Definition

In statistical machine translation we are given a Chinese sentence in characters $c_1^K = c_1 c_2 ... c_k ... c_K$ ($k \in 1, 2, .., K$), which is to be translated into an English sentence $e_1^I = e_1 e_2 ... e_i ... e_I$ ($i \in 1, 2, .., I$), where K and I is the length of Chinese sentence in characters and English sentence in words respectively.

In order to obtain a more adequate mapping between Chinese and English translation units, c_1^K is usually segmented into words. The positions of *Chinese word boundaries* on a character sequence c_1^K is indicated with $k_0 \equiv 0$ and $k_1^J = k_1 k_2 ... k_j ... k_J$ ($j \in 1, 2, .., J$), where $k_J \equiv K$, $k_j \in \{1, 2, ...K\}$ and $k_{j-1} < k_j$, and J is the number of Chinese words for c_1^K. They store the information where the Chinese words are delimited in a sentence. k_j indicates the j-th word segmentation boundary taking place after (on the right side of) the Chinese character c_{k_j} and before the character c_{k_j+1}, when $1 \leq j < J$. k_0 is a boundary before (on the left side of) the first Chinese character c_1, which is defined as 0 constantly, and k_J is always after the last Chinese character c_K and therefore equals K.

Given a sequence of Chinese characters c_1^K and its positions of segmentation boundaries k_1^J, a sentence can also be represented in the form of a sequence of Chinese words $f_1^J = f_1 f_2 ... f_j ... f_J$

($j \in 1, 2, .., J$), and each individual Chinese word f_j is defined as

$$f_j = c_{k_{j-1}+1}..c_{k_j} = c_{k_{j-1}+1}^{k_j}.$$

f_1^J is composed of two information sources, the character sequence and its word segmentation.

Table 1: An example for the definition of Chinese words and word segmentations

characters	小	孩	玩	纸	牌
c_1^K	c_1	c_2	c_3	c_4	c_5
segmentation boundaries	小	孩 |	玩	| 纸	牌 |
k_1^J	$k_1 = 2$		$k_2 = 3$	$k_3 = 5$	
characters into words	c_1c_2		c_3	c_4c_5	
words	小孩		玩	纸牌	
f_1^J	f_1		f_2	f_3	

An example is illustrated in Table 1. The sentence 小孩玩纸牌 contains five Chinese characters ($K = 5$), where c_1 denotes 小, c_2 denotes 孩, etc. The first word segmentation boundary takes place after the second Chinese character ($k_1 = 2$), and the second and third boundary is after the third and fifth character respectively ($k_2 = 3, k_3 = 5$), which indicates that 小(c_1) and 孩(c_2) compose together a word 小孩(f_1). 玩(f_2) is a single character word. 纸牌(c_4c_5) is the third word (f_3) of this sentence.

3 Ngram segmentation

Table 2: Manual Chinese word lexicon

Word	小	孩	玩	小孩	...
Frequency	3465	22	588	367	...

The simplest and widely applied automatic segmentation tool is based on an *unigram segmentation*, which requires a manual lexicon containing a list of Chinese words and their frequencies, as shown in Table 2. The lexicon and frequencies can be obtained using manually annotated data, e.g. the LDC (LDC, 2003) lexicon or extracted from the alignment of the training corpora (Xu et al., 2004) to be discussed in Section 4.

We need to maximize the probability of a sentence considering all word segmentation alternatives. Assuming each Chinese word in the sentence is independently distributed, we are interested to know how to put the word delimiters properly so that the product of probabilities of all words is maximized:

$$\hat{k}_1^{\hat{J}} = \underset{k_1^J, J}{\operatorname{argmax}} \prod_{j=1}^{J} Pr(c_{k_{j-1}+1}^{k_j}) \quad (1)$$

$$= \underset{k_1^J, J}{\operatorname{argmax}} \prod_{j=1}^{J} Pr(c_{k_{j-1}}^{k_j} | c_{k_{j-2-n}+1}^{k_{j-1-n}}, .., c_{k_{j-2}+1}^{k_{j-1}})$$

$Pr(c_{k_{j-1}+1}^{k_j})$ is the probability of a word f_j which is a sequence of characters $c_{k_{j-1}+1}^{k_j}$ with a boundary after the k_{j-1}-th character and before the k_j-th character in the sentence. Taking the word dependency into account and using the concept of the language model, we obtain an ngram model for Chinese word segmentation of order n. The dynamic programming algorithm is used to find the word sequence which has the highest multiplier product of word probabilities.

Another instance of this type segmenter is the LDC tool, which is also based on unigram segmentation but with additional text normalizations. The next word is selected from the longest phrase. More details can be found on the LDC web pages (LDC, 2003).

The unigram Chinese word segmentation method is so far the most commonly applied method in machine translation, but it has problems: First, finding the maximum mutual word probabilities does not guarantee the best combination among those words, so that the segmentation may contain errors; second, a more accurate word segmentation does not always lead to a great improvement in translation performance. The 'correct' segmentation for one character sequence is not universal but depends on the Chinese context and the destination language. For example, 纸 (paper) and 牌 (card) can be separated or composed into one word 纸牌 (cards). As 纸牌 does not exist in the manual lexicon, it cannot be generated by this unigram method.

4 Alignment derived segmentation

Table 3: An example of an alignment matrix between Chinese characters and English words. A gray box indicates a single-best (Viterbi) word alignment.

cards				■	■
play			■		
children	■	■			
	小	孩	玩	纸	牌

We will introduce our first word segmentation approach, namely alignment derived segmentation. In statistical machine translation we have a

bilingual corpus to obtain the Chinese word segmentation in the following way (Xu et al., 2004):

First, we train the statistical translation models with the bilingual corpus using GIZA++ (Och, 2000) tool. There is no word segmentation performed on Chinese texts, and each Chinese character is interpreted as a word.

As a result of this alignment training, we obtain for each sentence pair a mapping of Chinese characters to the corresponding English words, i.e. the single-best word alignment between Chinese characters and English words. Such an alignment is represented as a binary matrix with $K \cdot I$ elements. An example is shown in Table 3, where a Chinese training sentence in characters is plotted along the horizontal axis and its English translation sentence in words along the vertical axis. The black boxes show the best alignments for this sentence pair. In this example the first two Chinese characters are aligned to 'children', the next one is aligned to 'play', and the last two tokens are aligned to 'cards'.

Based on this information, we can generate a Chinese word list with each entry composed of one or more Chinese characters, which are aligned into one English word in the word alignment matrix. If we calculate the frequencies for every word, the distribution can be obtained, too. We accumulate the frequency of each entry over all sentence pairs in the training corpus. For instance, if we only have one sentence pair as in Table 3, we obtain words '小孩', '玩' and '纸牌', each of them has an absolute frequency of one and a relative frequency of $\frac{1}{3}$. With this self-learned lexicon we use a segmentation tool, such as an unigram segmenter in Section 3 to obtain a segmented Chinese text. Finally, we retrain our translation system with the segmented corpus.

This lexicon shows the most probable situation of Chinese characters occurrence, if they are combined or used alone, according to the single-best alignment. The extraction method differs from other self-learned methods because it uses the bilingual training corpus instead of the monolingual corpus such as in (Sproat and Shih, 1990). As we are more interested in the relationship between the languages, this method is more suitable for the machine translation application.

The central idea of our lexicon learning method is: A contiguous sequence of Chinese characters constitute a Chinese word, if they are aligned to the same English word. Using this idea and the bilingual corpus, we can automatically generate a Chinese lexicon. As a conclusion, our 'learned translation with learned segmentation' derived from character based word alignment consists of three steps:

1. The input is a sequence of Chinese characters without segmentation. After the training using GIZA++ , we extract a monolingual Chinese dictionary from the alignment.

2. Using this learned dictionary we segment the sequence of Chinese characters into words. In other words, the unigram method is used, but the manual lexicon is replaced by the learned lexicon.

3. Based on this word segmentation, we perform another training using GIZA++ . Then, after training the models IBM1, HMM and IBM4, we extract bilingual word groups, which are referred to as phrase based translation.

5 Semi-supervised Chinese word segmentation

The alignment derived segmentation models the word distributions in the lexicon using Viterbi alignment information. It uses the bilingual information and is trained with the respect to the translation performance. But the erroneous single-best alignment can result in the incorrect word segmentation, which may lead to sub-optimal translation results. Therefore, we further propose another Chinese word segmentation model in training, which is more refined and learns both Chinese word entries and their distributions to generate a dynamic lexicon. New words are introduced with a prior distribution using the Dirichlet process. Chinese word segmentation and word alignment, which have an influence and effect on each other, are trained simultaneously.

This method is semi-supervised, namely, Chinese word segmentation is jointly trained with word alignment given an initialized word segmentation and alignment. Motivated by (Goldwater et al., 2006), we employ Dirichlet process to introduce new words to the lexicon with a prior distribution. In addition, we describe a generative model which consists of a word model and two alignment models, representing the monolingual and bilingual information respectively. In our methods, we first initialize Chinese text using a unigram segmenter and then learn new word types and word distributions, which are suitable for machine translation.

The training and translation processes for semi-supervised Chinese word segmentation are as follows: The inputs to the system are the bilingual training data including Chinese sentences in characters and its English translations in words, a manual Chinese word lexicon, such as LDC lexicon, as well as a test corpus on the character level. First, we segment the Chinese training corpus in characters with an unigram word segmentation using the manual lexicon, then the word alignment and Chinese word segmentation are jointly trained as

output using semi-supervised Chinese word segmentation. By counting the Chinese word frequencies, we easily obtain an automatic generated lexicon. A combined lexicon with the automatic and manual lexicons is then applied to perform an unigram segmentation on the test corpus. The Chinese word segmentation on the test corpus is another output from our segmentation system.

We apply Gibbs sampling for model training. The Gibbs sampling algorithm generates an instance from the distribution of each variable in turn, conditional on the current values of the other variables. This characteristic is particular interesting if the Chinese words are unknown and to be learned. There are approximately 90K Chinese characters and 7000 commonly in use; any of these characters can be elements of a word. Usually, a Chinese word is composed of one to four characters. The number of Chinese words is calculated as $7000 + 7000^2 + 7000^3 + 7000^4$, which is difficult to be fixed before translation. If there is no previously defined lexicon, the number of all possible segmentations for a sequence of characters c_1^K is 2^{K-1}. That means that the complexity is exponential in the order of character sequence length. Therefore, we have to approximate the space of all possible derivations in some way: We can define that a word contains at most four characters, the complexity goes down to polynomial, but it is still a high order polynomial; we can perform a standard beam search to prune low cost paths. As an alternative to draw a space with all segmentation derivations for one sentence, we imply the Gibbs sampling algorithm, which learns each parameter value conditioning on all other parameter values in turn. (Blunsom and Osborne, 2008) showed that the search spaces produced by the sampling approach occupied roughly half the disc space as those produced by the beam search with similar results.

5.1 Unigram Dirichlet Process model for CWS

The simplest version of this model is based on an unigram Dirichlet Process (DP) model as introduced by (Goldwater et al., 2006), using only monolingual information. Different from a standard unigram model for CWS, our model can introduce new Chinese word types and learns word distributions automatically from unlabeled data. On the contrary, the conventional approach applies a manual lexicon containing fixed Chinese words and their frequencies as distributions.

According to this model, a corpus of Chinese words $f_1, \ldots f_j, \ldots$ is generated based on Dirichlet Process. Each random variable f_j is drawn independently and identically from G , where G is a distribution over words drawn from a Dirichlet Process prior with base measure P_0 and concentration parameter α.

We never estimate G explicitly but instead integrate over its possible values and perform Bayesian inference. It is easy to compute the probability of a Chinese word given a set of already generated words, while integrating over G. This is done by casting a Chinese word generation as a Chinese restaurant process (CRP) (Aldous, 1985), i.e. a restaurant with an infinite number of tables (approximately corresponding to Chinese word types), each table with infinite number of seats (approximately corresponding to Chinese word frequencies).

The Dirichlet Process model can be viewed intuitively as a cache model (Goldwater et al., 2006). Each word f_l in the corpus is either retrieved from a cache or generated anew given the previously observed words f_{-l}:

$$P(f_l|f_{-l}) = \frac{N(f_l)_+ \alpha P_0(f_l)}{N + \alpha}, \quad (2)$$

where $N(f_l)$ is the number of Chinese words f_l in the previous context. N is the total number of Chinese words, P_0 is the base probability over words, and α influences the probability of introducing a new word at each step and controls the size of the lexicon. This so-called rich-get-richer process creates a Zipf distribution. The probability of generating a word from the cache increases as more instances of that word are seen. α controls the number of word types, i.e. size of the lexicon. It is the total probability to generate any new words. P_0 defines a probability distribution over new words, i.e. how likely a sequence of Chinese characters forms a word.

For the base distribution P_0, which governs the generation of new words, we use the following distribution (called the **spelling model**):

$$P_0(f) = P(\mathcal{L}) \cdot P(f|\mathcal{L}) \quad (3)$$

$$= \frac{\kappa^{\mathcal{L}}}{\mathcal{L}!} e^{-\kappa} \cdot (\frac{1}{v})^{\mathcal{L}}, \quad (4)$$

where $\mathcal{L}$ is the number of Chinese characters of word f. We decompose the spelling model into a word length model $P(\mathcal{L})$ and a word model depending on its length $P(f|\mathcal{L})$. The length model follows a Poisson distribution, and the word model is a uniform distribution over all words given a length. Therefore, Equation 3 is extended to Equation 4. v is the number of characters in the document, i.e. character vocabulary size. We note that the sum of probabilities $(\frac{1}{v})^{\mathcal{L}}$ over all words with a length $\mathcal{L}$ equals to one, because there are $v^{\mathcal{L}}$ words with length $\mathcal{L}$, and each word is equally distributed

with a probability $\frac{1}{v}^{\mathcal{L}}$, so $(v^{\mathcal{L}})((\frac{1}{v})^{\mathcal{L}}) = 1$. In our experiments we used $\kappa = 2$ and $\alpha = 0.3$.

5.2 Generative model

As shown in Figure 4, the generative model assumes that a corpus of parallel sentences (${c_1}^K$,${e_1}^I$) is generated along with a hidden sequence of Chinese words ${f_1}^J$ and a hidden word alignment ${b_1}^I$ for every sentence. The alignment indicates the aligned Chinese word f_{b_i} for each English word e_i, where f_0 indicates a special *null* word as in the IBM models.

The joint probability of the observations $({c_1}^K, e_1^I)$ can be obtained by summing up all possible values of the hidden variables k_1^J and b_1^I. The model probability $Pr(c_1^K, e_1^I)$ can be seen as the sum of all possible Chinese word segmentations k_1^J of the character sequence c_1^K:

$$\begin{aligned} & Pr(c_1^K, e_1^I) \\ &= \sum_{k_1^J} \sum_{b_1^I} Pr(c_1^K, e_1^I, k_1^J, b_1^I) \quad (5) \\ &= \sum_{k_1^J} \sum_{b_1^I} Pr(f_1^J) Pr(e_1^I, b_1^I | f_1^J) \quad (6) \end{aligned}$$

Without assuming any special form for the probability of a sentence pair along with hidden variables, we can factor it into a monolingual Chinese sentence probability and a bilingual translation probability. As f_1^J represents the information of c_1^K and k_1^J we can rewrite Equation 5 into Equation 6. Therefore, the observations (c_1^K, e_1^I) are assumed to be generated in three steps:

1. Word sequence f_1^J is generated via a word model $Pr(f_1^J)$.
2. Chinese character sequence is generated from f via a spelling model $P(f)$.
3. English words are generated via a translation model $Pr(e_1^I | f_1^J)$.

In the following paragraphs we will describe the modeling assumptions behind the monolingual Chinese sentence model including word, spelling models and the translation model, respectively.

5.2.1 Monolingual Chinese sentence model

We use the Dirichlet Process unigram word model. In this model the parameters of a distribution over words G are first drawn from the Dirichlet prior $DP(\alpha, P_0)$. Then words are independently generated according to G. The probability of a sequence of Chinese words in a sentence is thus:

$$Pr(f_1^J) = \prod_{j=1}^{J} P_G(f_j), \quad (7)$$

where $P_0(f_j)$ is further explained by the spelling model.

5.2.2 Translation model

We employ the Dirichlet process inverse IBM model 1 to generate English words and alignments given the Chinese words. In this model, for every Chinese word f (including the *null* word), a distribution over English words G_f is drawn firstly from a Dirichlet Process prior $DP(\alpha, P_0(e))$, where $P_0(e)$ we used the empirical distribution over English words in the parallel data. Then, given these parameters, the probability of an English sentence and alignment given a Chinese sentence (sequence of words) is given by:

$$P(e_1^I, b_1^I | f_1^J) = \prod_{i=1}^{I} \frac{1}{J+1} P_{G_{f_{b_i}}}(e_i | f_{b_i}),$$

where e_i is distributed according to $G_{f_{b_i}}$. This is the same model form as inverse IBM model 1. We have placed Dirichlet Process priors on the Chinese-word specific distributions over English words. [1]

In practice, we observed that using a word-alignment model in one direction is not sufficient. We then added a factor to our model which includes word alignment in the other direction. Such combinations of models in both directions are widely used for phrase extraction (Och and Ney, 2004).

Therefore, we also used a translation model in the other direction, , i.e. a Dirichlet Process IBM model 1. We ignore the detailed description here, because the calculation is the same as that of the inverse IBM model 1. According to this model, for every English word e (including the *null* word), a distribution over Chinese words G_e is first drawn from a Dirichlet Process prior $DP(\alpha, P_0(f))$. Here, for the base distribution $P_0(f)$ we used the same spelling model as for the monolingual unigram Dirichlet Process prior.

5.3 Final model

We put the monolingual model and the translation models in both directions together into a single model, where each of the component models is weighted by a scaling factor. This is similar to a maximum entropy model. We fit the weights of

[1] f_{b_i} is the Chinese word aligned to e_i and $G_{f_{b_i}}$ is the distribution over English words conditioned on the word f_{b_i}. Similarly, e_{a_j} is the English word aligned to f_j in the other direction and $G_{e_{a_j}}$ is the distribution over Chinese words conditioned on e_{a_j}.

Table 4: Observations and hidden variables of the generative model for Chinese word segmentation.

	Symbol	Abb.	Example
Observations			
Chinese sequence in characters	c_1^K	C	小 孩 玩 纸 牌
English sentence	e_1^I	E	Children play cards
Hidden variables			
Alignment normal	a_1^J	A	e.g. (cards,纸),(cards,牌),..
Alignment inverse	b_1^I	B	e.g. (纸,cards),(牌, cards)
Segmentation (Chinese sequence in words)	f_1^J	F	e.g. 小孩→玩→纸牌

the sub-models on a development set by maximizing the BLEU score of the final translation.

We used three features derived from Equation 7 and equations in Section 5.2.2.

The maximum entropy model can be viewed as a weighted linear combination of the log probabilities of sub-models. The weights that are optimized on development datasets have empirical justifications. Since different sub-models have been trained on different datasets, their dynamic value ranges can be so different that it is inappropriate to combine their log probabilities through simple addition. Moreover, for instance, some models may be poorly estimated due to the lack of large amount of training data. Therefore, empirical results have demonstrated that the use of scaling factors that reflect the relative contribution of different sub-models often improves the performance. The final model used in our experiments is

$$\begin{aligned} & Pr(c_1^K, e_1^I, f_1^J, a_1^J, b_1^I) \qquad (8) \\ \approx\ & \frac{1}{Z} Pr(f_1^J)^{\lambda_1} \cdot Pr(e_1^I, b_1^I | f_1^J)^{\lambda_2} \\ & \cdot Pr(f_1^J, a_1^J | e_1^I)^{\lambda_3}, \end{aligned}$$

where Z is the normalization factor and a is the alignment for Chinese to English translation.

In practice, we do not re-normalize the probabilities and our model is thus deficient because it does not sum to 1 over valid observations. However, we observed that the model worked very well in our experiments. Similar deficient models have been used very successfully before, for example in the IBM models 3–6 (Och, 1999).

5.4 Gibbs sampling training

Using our generative model we would like to choose the most likely word segmentation given the observed pairs of Chinese-English sentences.

It is generally impossible to find the most likely segmentation according to our Bayesian model using exact inference, because the hidden variables do not allow exact computation of the integrals. Nonetheless, it is possible to define algorithms using Markov chain Monte Carlo (MCMC) that produce a stream of samples from the posterior distribution of the hidden variables given the observations. We applied the Gibbs sampler (Geman and Geman, 1984), one of the simplest MCMC methods, in which transitions between states of the Markov chain result from sampling each component of the state conditioned on the current value of all other variables.

In our problem the observations are $D = (d_1, ..d_s, .., d_S)$, where d_s=(c_1^K, e_1^I) indicates a bilingual sentence pair, the hidden variables are the word segmentations f_1^J and the alignments in two directions a_1^J and b_1^I.

Gibbs sampling is an iterative procedure that samples variables given the current values of all other variables. Our Gibbs sampler for Chinese word segmentation works as follows: for each step we take a single possible boundary point by fixing other segmentations and alignments, then compare hypotheses considering this boundary and the related alignments. After sampling by using the posterior probabilities of each candidate, we choose one of the candidates and perform the same operation for the next position.

To perform Gibbs sampling we start with an initial word segmentation and initial word alignments. We re-sample iteratively the word segmentation and alignments according to our model of Equation 8.

For example, we are interested in determining the word boundary after 纸in the sentence 小孩玩纸牌. We only use the monolingual model for convenience. We suppose that 纸 牌are two words from the initialization. N is the number of words in Chinese corpus. First, we decrease the related counts N, N(纸), N(牌), N(纸, Children), .. with one. After that, we calculate the probabilities P(纸牌|..), P(纸牌|..), ... again. Now, we compare the P(纸 牌|..) and P(纸牌|..) using sampling, i.e. after the normalization on the probabilities: P'(纸牌|..) and P'(纸牌|..), we select a random number x in $(0, 1)$, If x in $(0, P'$(纸牌|..)), we choose 纸牌, otherwise, we choose 纸 牌. That means higher probability segmentation is more likely to

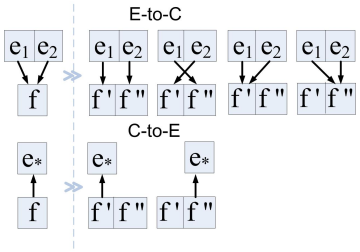

Figure 1: Case I, transition from a no-boundary to a boundary state, f to $f'f''$.

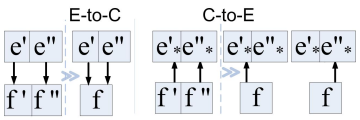

Figure 2: Case II, transition from a boundary to a no-boundary state, $f'f''$ to f.

be chosen. Finally, we increase the corresponding counts N, N(纸牌), N(纸牌, Children), N(纸牌, play), N(纸牌, card),.. with one. This is an iterative process going over all positions in a document until the segmentation result converges.

We only allow limited modifications to the initial word alignments for reasons of efficiency. Thus, we only use models derived from IBM-1 (instead of IBM-4) for comparing different word segmentations and not for large-scale modification of the word alignment. IBM model 4 from GIZA++ is an improved model in comparison to IBM model 1 that we use. On the other hand, re-sampling the segmentation causes re-linking alignment points to parts or groups of the original words.

Hence, we organize our sampling process around possible word boundaries. For each character c_k in each sentence, we consider two alternative segmentations: ${c_k}^+$ indicates the segmentation where a boundary exists after c_k and ${c_k}^-$ indicates the segmentation where no boundary exists after c_k, keeping all other boundaries fixed. Let f denote the single word spanning character c_k if there is no boundary after it, and f',f'' denote the two adjacent words resulting if there is a boundary: f' includes c_k and f'' starts just to the right, with character c_{k+1}. The introduction of f' and f'' leads to P new possible alignments in the E-to-C direction $b^+_{k1}, \ldots, b^+_{kP}$, such as in Figure 1. Together with the boundary vs no-boundary state at each character position, we re-sample a set of alignment links between English words and any of the Chinese words f,f', and f'', keeping all other word alignments in the sentence pair fixed. (See Figures 1 and 2.)

Table 5: General Algorithm of GS for CWS.

Input: D with an initial segmentation and alignments
Output: D with sampled segmentation and alignments
for $s = 1$ to S : each sentence
for $k = 1$ to K that $c_k \in d_s$: each character position
Create P+1 candidates, $cba^+_{k,p}$ and cba^-_k, where
$cba^+_{k,p}$: there is a word boundary after c_k
cba^-_k: there is no word boundary after c_k
Compute probabilities
$P(cba^+_{k,p}\|{dh_{sk}}^-)$
$P(cba^-_k\|{dh_{sk}}^-)$
Sample boundary and relevant alignments
Update counts

Thus, we consider a set of alternatives for the boundary after c_k and relevant alignment links at each step in the Gibbs sampler, keeping all other hidden variables fixed. We need to compute the probability of each of the alternatives at each step, given the fixed values of the other hidden variables.

We introduce some notation to make the presentation easier. For every position k in sentence pair s, we denote by ${dh_{sk}}^-$ the observations and hidden variables for all other sentences than sentence s, and the observations and hidden variables inside sentence s, not involving character position c_k. The fixed variables inside the sentence are the words not neighboring position k and the alignments in both directions to these words.

In the process of sampling we consider a set of alternatives: segmentation ${c_k}^+$ along with the product space of relevant alignments in both directions $b^+_{k1}, \ldots, b^+_{kP}$, and a^+_k, and segmentation c^-_k along with relevant alignments ${b_k}^-$ and a^-_k. For brevity, we denote these alternatives by ${cba_{k,p}}^+$ and ${cba_k}^-$.

We will describe how we derive the set of alternatives in Section 5.5 and how we compute their probabilities in section 5.6.1.

Table 5 shows schematically one iteration of Gibbs sampling through the whole training corpus of parallel sentences, where S is the number of parallel sentences.

5.5 Computing probabilities of alternatives

For the Gibbs sampling algorithm in Table 5, we need to compute the probability of each alternative segmentation/alignments, given the fixed values of the rest of the data ${dh_{sk}}^-$. The probability of the hidden variables in the alternatives is proportional to the joint probability of the hidden variables and observations, and thus it is sufficient to compute

the probability of the latter. We compute these probabilities using the Chinese restaurant process sampling scheme for the Dirichlet Process, thus integrating over all of the possible values of the distributions G, G_f and G_e.

Let cba_k denote an alternative hypothesis including boundary or no boundary at position k, and relevant alignments to English words in both directions of the one or two Chinese words resulting from the segmentation at k. The probability of this configuration given by our model is:

$$P(cba_k|dh_{sk}{}^-) \propto P_m(cba_k|dh_{sk}{}^-)^{\lambda_1} \cdot P_{ef}(cba_k|dh_{sk}{}^-)^{\lambda_2} \cdot P_{fe}(cba_k|dh_{sk}{}^-)^{\lambda_3}, \quad (9)$$

where $P_m(cba_k|dh_{sk}{}^-)$ is the monolingual word probability, and $P_{fe}(cba_k|dh_{sk}{}^-)$ and $P_{ef}(cba_k|dh_{sk}{}^-)$ are the translation probabilities in the two directions.

Now we describe the computations of each of the component probabilities.

5.5.1 Word model probability

The word model probability $P_m(cab_k|dh_{sk}{}^-)$ in Equation 9 is derived from Equations 7 and 2. There are two cases: If the hypothesis specifies that there is a boundary after character c_k, we need multiply probabilities of the two resulting words f', and f'' using Equations 7; otherwise, $P_m(c_k^+|dh_{nk}{}^-)$ is estimated by the probability of the single word f. (See the initial states in Figures 1 and 2, respectively.)

In the first case, due to theoretical correctness, we need to update the counts N and $N(f'')$ if computing the probability of the second word, but this is unlikely to change the behavior of the algorithm and we kept the counts fixed while computing the probability of hypotheses for the word and translation models.

5.5.2 Translation model probability

The translation model probabilities depend on whether or not there is a segmentation boundary at c_k. They also depend on which English words are aligned to the relevant Chinese words.

In the first case, we assume that there is a word boundary in cab_k, and that English words $\{e_1\}$ are aligned to f' and words $\{e_2\}$ are aligned to f'' in the E-to-C direction according to the alignment b_k, and that f' is aligned to e_* or f'' is aligned to e_* in the C-to-E direction according to the alignment a_k (see the initial state in Figure 1). Here, we overloaded notations and use b_k and a_k to indicate the alignments of the relevant Chinese words at position k to any English words. Let I denote the total number of English words in the sentence, and $J+1$ denote the number of Chinese words according to this segmentation. We consider the *null* words. We also denote the total number of English words aligned to either f' or f'' in the E-to-C direction by P.

The translation model probability in the E-to-C direction is thus:

$$P_{ef}(c_k^+, b_k, a_k|dh_{nk}{}^-) \propto \left(\frac{1}{J+2}\right)^P \cdot \prod_{e'} P(e'|f', dh_{nk}{}^-) \prod_{e''} P(e''|f'', dh_{nk}{}^-)$$

Here we compute $P(e|f, dh_{nk}{}^-)$ as:

$$P(e|f, dh_{nk}{}^-) = \frac{N(e,f) + \alpha P_0(e)}{N(f) + \alpha},$$

where the counts are computed over the fixed assignments $dh_{nk}{}^-$.

The translation probability in the other direction is computed similarly.

The parameters θ are estimated on-the-fly, which means updating θ is to update the counts $N(f,e)$, N, $N(e)$ and $N(f)$ according to our model. The probabilities are computed if it is called in the sampling.

5.6 Determining the set of alternative hypotheses

Sampling on word segmentation can change the Chinese word and its alignment. Therefore, some implementation issues need to be addressed to enable the algorithm to work properly in our experiments.

5.6.1 How to maintain one-to-many alignment during sampling?

As mentioned earlier, we consider alternative alignments which deviate minimally from the current alignments and which satisfy the constraints of the IBM model 1 in both directions. In order to describe the set of alternatives, we consider two cases depending on whether there is a boundary at the current character before sampling at position k.

Case 1. There is no boundary at c_k in the previous state (see Figure 1).

If there is no boundary at c_k, there is a single word f spanning that position. We denote by $\{e\}$ the set of English words aligned to f at that state in the E-to-C direction and by e_* the English word aligned to f in the C-to-E direction. Due to the fact that every state we consider satisfies the IBM one-to-many constraints, there is exactly one English word aligned to f in the C-to-E direction and the words $\{e\}$ have no other words aligned to them in the E-to-C direction.

In this case, we consider as hypothesis ${cba_k}^-$ the same segmentation and alignment as in the previous state. (see Table 5 for an overview of the alternative hypotheses.)

We consider M different hypotheses which include a boundary at k in this case, where M depends on the number of words $\{e\}$ aligned to f in the previous state. As we are breaking the word f into two words f' and f'' by placing a boundary at c_k, we need to re-align the words $\{e\}$ to either f' or f''. Additionally, we need to align f' and f'' to English words in the C-to-E direction. The number of different hypotheses is equal to 2^P where $P = |\{e\}|$. These alternatives arise by considering that each of the words in $\{e\}$ needs to align to either f' or f'', and there are 2^P combinations of these alignments. For example, if $\{e\} = \{e_1, e_2\}$, after splitting the word f there are four possible alignments, illustrated in Figure 1: I. (f', e_1) and (f'', e_2), II. (f', e_2) and (f'', e_1), III. (f', e_1) and (f', e_2), IV. (f'', e_1) and (f'', e_2). For the alignment a_k in the C-to-E direction, we consider one option only, in which both resulting words f' and f'' align to e_*. These alternatives form ${cba_{k,m}}^+$ in Table 5.

Case 2. There was a boundary at c_k in the previous state (see Figure 2).

In this case, for the hypothesis c_k^+ we only consider one alternative, which is exactly the same as the assignment of segmentation and alignments in the previous state. Thus we have $P = 1$ in Table 5.

Let f' and f'' denote the two words at position k in the previous state, $\{e'\}$ and $\{e''\}$ denote the sets of English words aligned to them in the E-to-C direction, respectively, and ${e_*}'$ and ${e_*}''$ denote the English words aligned to f' and f'' in the C-to-E direction.

We only consider one hypothesis ${cba_k}^-$ where there is no boundary at c_k. In this hypothesis, there is a single word $f = f'f''$ spanning position k, and all words $\{e'\} \cup \{e''\}$ align to f in the E-to-C direction. For the C-to-E direction we consider the 'better' one of the alignments (f, e'_*) and (f, e''_*) where the better alignment is defined as the one having higher probability according to the C-to-E word translation probabilities.

5.7 Complete segmentation algorithm

So far, we have described how we re-sample word segmentation and alignments according to our model, starting from an initial segmentation and alignments from GIZA++. Putting these pieces together, we get the algorithm that is summarized in Table 5.

We figured out that we can further improve performance by aligning repeatedly the corpus using GIZA++. We do so after deriving a new segmentation using our model. The complete algorithm, which includes this step, is shown in Table 6, where F_t indicates the word segmentation at iteration t and A_t denotes the GIZA++ corpus alignment in both directions. The GS re-segmentation step is done according to the algorithm in Table 5.

Table 6: Complete algorithm of Gibbs sampler for CWS including alignment models. The observations are $D = (d_1, ..d_s, .., d_S)$, where d_s=(c_1^K, e_1^I) indicates a bilingual sentence pair. Hidden variables F_t and A_t is the word segmentation and alignment of the corpus in the t-th iteration respectively.

Input: D, F_0
Output: A_T, F_T
for $t = 1$ to T : each iteration
Run GIZA++ on (D, F_{t-1}) to obtain A_t
Run GS on (D, F_{t-1}, A_t) to obtain F_t

Using this algorithm, we obtain a new segmentation of the Chinese data and train the translation models using this segmentation as in the baseline machine translation system. To segment the test data for translation, we use a unigram model, trained with maximum likelihood estimation of the final segmentation of the training corpus F_T.

6 Integrated word segmentation in search

We described the alignment derived and semi-supervised Chinese word segmentation methods in Section 4 and 5, where the word segmentation is learned during the training of the word alignment. However, a test corpus is still processed using a unigram segmenter, which does not guarantee optimal segmentations as addressed in Section 1. For instance, 小孩 and 孩 can both mean 'children'. The first one is used more often. Therefore, a segmenter usually puts both characters together rather than separating them. But if only 孩, but not 小孩, appears in the training corpus, 小孩 in the test corpus should be broken into two words, so that 孩 can be recognized and translated into 'children'.

Whenever inconsistencies show up between the expressions in training and test data, considering segmentation alternatives is a good way to adapt the writing style of the test text to that of the training text. Hence, a so-called 'integrated word segmentation' will be described in detail in this section, referring to the paper of (Xu et al., 2005a). The algorithm works as follows: Given a set of character sequences as the input text, we take all possible segmentations of one sentence into account and integrate the segmentation decision into

the search for the translation. Different segmentation possibilities represented as a lattice instead of a single segmentation are translated, and the segmentation decision is only taken during the search for the best translation.

In the conventional method only a single-best word segmentation is employed in the search for the best translation. This approach is not ideal because the segmentation may not be optimal for these translations given the training data segmentation. Making hard decisions in word segmentation may lead to loosing Chinese words that can contribute to find the correct translations. Hence, for one input sentence, we take all possible segmentations into account and represent them as a lattice. The input to the translation system is then a set of lattices instead of the segmented text. In the integrated segmentation, the search decision of the word segmentation is combined with the translation decision as a global decision. The best segmentation of a sentence is only selected while the translation is generated. Using this method, by giving a segmented training corpus, we are able to translate any character-based Chinese text, and it even outperforms a standard approach with manually segmented input text. The comparison of the results will be shown in the coming sections.

6.1 Integrated Chinese word segmentation model

We take the notation in Section 2. In the conventional approach the best translation of c_1^K can be performed by first finding the best segmentation as in Equation 1, then searching for the best translation given fixed word segmentation:

$$\hat{e}_1^{\hat{I}} = \underset{e_1^I}{\operatorname{argmax}} \left\{ Pr(e_1^I | c_1^K, \hat{k}_1^{\hat{J}}) \right\} \quad (10)$$

However, in the transfer of the single-best segmentation from Equation 1 to Equation 10 some segmentations that are potentially optimal for the translation may be lost. Therefore, we combine the two steps. The search is then rewritten as:

$$\begin{aligned}
\hat{e}_1^{\hat{I}} &= \underset{I, e_1^I}{\operatorname{argmax}} \left\{ Pr(e_1^I | c_1^K) \right\} \\
&= \underset{I, e_1^I}{\operatorname{argmax}} \left\{ \sum_{k_1^J, J} Pr(k_1^J, e_1^I | c_1^K) \right\} \\
&\cong \underset{I, e_1^I}{\operatorname{argmax}} \left\{ \max_{k_1^J, J} \left\{ Pr(k_1^J | c_1^K) Pr(e_1^I | c_1^K, k_1^J) \right\} \right\}
\end{aligned}$$

We optimize the segmentation boundaries k_1^J to achieve the best translation directly. In this way, the segmentation model and the translation model are combined into one model. The global decision on Chinese word segmentation and translation are performed together.

6.2 Constructing segmentation lattices

To perform the lattices translation we introduce the weighted finite-state acceptor (Kanthak and Ney, 2004). Now, we will take a short sentence as an example and simulate the segmentation process. The Chinese sentence is selected from the (IWSLT, 2005) development corpus, '在哪里办理登机手续?', which consists of nine characters including a punctuation mark.

There are many approaches to build a segmentation lattice. The aim of the lattice construction is on one hand allowing word segmentation alternatives as candidates for translation and on the other hand avoiding too many ambiguities so that segmentations leading to optimal translations can always be preferred.

The simplest lattices are linear constructed namely a word sequence is taken as the only path in the lattice, and each word marks the input label on the succeeding arcs in its sequential order, which is equivalent to a single-best translation.

In order to introduce segmentation alternatives N-best word segmentations instead of the single best segmentation are used in the translation. Chinese texts processed using different word segmentation methods are concatenated one after another to train the word alignment models. A segmentation lattice offers multiple paths with different segmentation possibilities letting the decoder take the final decision on the optimal word boundaries.

If vocabulary of Chinese words is given, it is possible to construct a lattice with all possible segmentations for a sentence. Allowing all alternative word segmentations tends to be a good idea, if several segmentations are not sufficient to detect proper words that are consistent with the training texts. This is realized by using the operator 'composition' under the concept of finite state acceptor introduced in the beginning of Section 6.2.

We generate the segmentation lattice with the following steps:

1. First, we make a word list from the vocabulary of the Chinese training corpus which contains all the entries that could be translated. Each word in the list is mapped by its characters to be consistent with the input of an unsegmented text. There may be several mapped words for one character sequence.

   In order to avoid the problem of the unknown characters from the unsegmented corpus, the additional characters from the test corpus are also added to the word list.

2. We convert the mapping of the word list into a finite-state transducer for segmentation, as

shown in Figure 3. Here the input labels are the characters from the test corpus, and the output labels are Chinese words to be translated by the translation system. The state 0 is the start and end state.

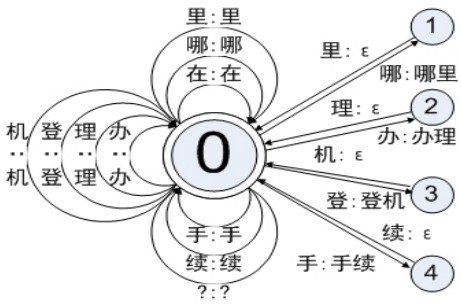

Figure 3: Segmentation transducer.

3. The input character sequence is represented as a linear acceptor in the same way of the single-best segmentation.

4. The linear automata is composed with the segmentation transducer in Figure 3. The result is a lattice which represents all possible segmentations of this sentence as shown in Figure 4. Note that the alphabet in the third step is a subset of the input alphabet in Figure 3, because the unknown characters are added as single words in the word list.

5. Now we get a new finite-state acceptor representing all the alternatives of different word segmentations. We only need to read the segmentation lattice in Figure 4 to have an integrated word segmentation in the translation.

6.3 Weighting segmentation lattices

The availability of introducing segmentation alternatives is based on the assumption that the decoder is 'strong' enough to choose the right segmentation using the translation model costs. However, if there are ambiguities, the decoder might only prefer a path with lower translation costs without considering any context information. As a result, translations differ to a great extent in comparison to the original text. Therefore, we discuss possible features to evaluate different segmentations in this sub-section. Paths in a lattice are weighted by feature costs. Less appeared word segmentations are penalized and more occurred word segmentations gain priority. In this way features in the segmentation lattice and the decoder contribute both to find the best segmentation results.

We will describe two models to weight lattice paths. The first one is the length model based on the observation that single character words are often chosen without context meanings. The other one is a language model measured on the Chinese training text. A unigram language model gives priorities to frequent words used in the training data, and higher order language models also capture the source context information for decisions.

A word segmentation model represents the fluency of a Chinese word sequence and can be built as an n-gram language model of the word-based text. We trained the language model on the Chinese training corpus with the SRILM toolkit (Stolcke, 2002) and used the modified Kneser-Ney discounting. To combine the segmentation lattice with the word based language model we simply transform the language model into a finite-state transducer and compose the lattice with it. Note that after inserting the weights the number of states and arcs in a lattice may increase because of the language model histories.

7 Translation experiments

In this work experiments are performed on two types of datasets: a small data track, where the training corpus is rather clean and contains less than ten thousand words, which is efficient to test the translation algorithms on sparse data; a large data track, where the best translation system is expected, the bilingual training corpus contains hundreds of million words, and the monolingual English data can obtain trillions of words.

We take the IWSLT (International Workshop on Spoken Language Translation) task for the small data track. The IWSLT organization holds an annual evaluation campaign that is carried out using a multilingual speech corpus on a small data track. The provided *Basic Travel Expression Corpus* (BTEC) (Takezawa et al., 2002) is a multilingual speech corpus which contains tourism-related sentences similar to those that are found in phrase books.

After the tokenization and automatic sentence segmentation, the training corpus nearly contains 43K bilingual sentences for each language as shown in Table 7. We calculated the number of words and the vocabulary size as well as the number of singletons of the corpus.

As shown in Table 7, we used three test sets from (IWSLT, 2007) translation evaluations, the Dev2, Dev3 and Eval in 2007. Each of them contains 16 references respectively. For convenience, we only list the statistics of the first reference translation after the tokenization. The Dev2 is selected as the development corpus, Dev3 and Eval are taken as evaluation corpora. We show the statistics using different Chinese word segmentations, which includes translation by taking each character as a single word, ICT (Zhang and Malik, 2003), LDC (LDC, 2003), unigram segmentation as described in Section 3 of our own implemen-

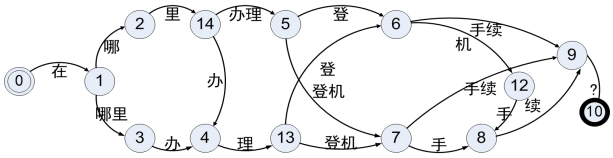

Figure 4: Segmentation lattice without weights including all word segmentation alternatives given a vocabulary.

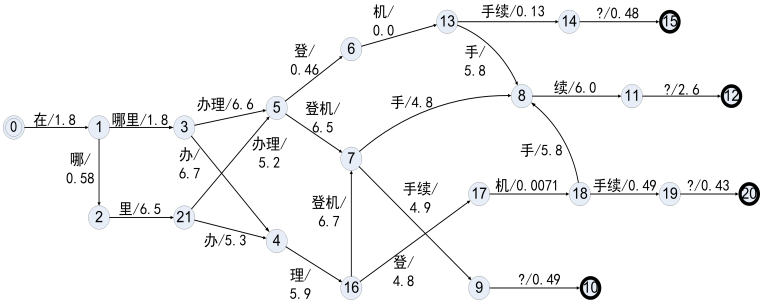

Figure 5: Segmentation lattice weighted by a language model considering all alternatives given a vocabulary.

tation, alignment derived segmentation as introduced in Section 4 where word alignments in both directions are combined using IU approach (Och, 2002), as well as the semi-supervised segmentation using Gibbs sampling described in Section 5. Running words (R.W.), OOVs of running words, i.e. OOVs (R.W.) and OOVs of vocabulary, i.e. OOVs (voc.) are listed, too.

For the large data track we experiment on the GALE 2008 task. The GALE (Global Autonomous Language Exploitation) program is one of the well-known machine translation projects based on a very large amount of training data, which is a set of individual corpora collected from different sources provided by the Linguistic Data Consortium (LDC, 2005) and GALE. The domains of most sub-corpora are news articles. Some sub-corpora contain documents from other domains, such as transcriptions of broadcast conversation, web text and newsgroups. The corpus statistics of the bilingual training data and the test sets are shown in Table 8. The preprocessing step includes the tokenization and the categorization of the numbers and dates. Long sentences are segmented into short sentences using the binary segmentation method in (Xu et al., 2005b) to reduce the training time. After the preprocessing and segmentation, the parallel training data contains more than 7.5 million sentences and more than 90 million words in each language.

Our baseline systems are the official submission systems by the RWTH-Aachen university in evaluations of IWSLT 2007 and GALE 2008. The training corpus (Train) is used to train the word alignment and segmentation models. The feature weights of different translation models are optimized on the development corpus (Dev) using the downhill simplex (Press et al., 2002) algorithm with respect to the BLEU (Papineni et al., 2002) score. The resulting systems are evaluated on the evaluation (Eval) corpora. For convenience we evaluate hypotheses without case information.

7.1 Statistics of word Length in dictionary

The central idea of the learned and semi-supervised CWS methods is to generate automatically the lexicon using bilingual information so that the segmentation is task- and domain- oriented. As there is no unique definition of a ‘correct‘ lexicon, here we will compare the statistics on the word lengths in the learned lexicon and semi-supervised lexicon with the manual lexicon provided by LDC.

Table 9: Statistics of word lengths in the vocabulary of the LDC lexicon, learned lexicon with alignment combination IU and semi-supervised lexicon using GS.

$\mathcal{L}$	LDC lexicon		learned-IU		GS lexicon	
	count	[%]	count	[%]	count	[%]
1	2 334	18.6	2 582	16.5	1 941	29.3
2	8 149	65.1	6 926	44.1	3 599	54.3
3	1 188	9.5	3 670	23.4	508	7.67
4	759	6.1	1 507	9.60	141	2.13
5	70	0.6	490	3.12	24	3.62
6	20	0.2	267	1.70	9	1.36
7	6	0.0	118	0.75	3	0.45
	11	0.0	130	0.82	1	0.01
	12 527	100	15 690	100	6226	100

Table 7: Corpus Statistics of task IWSLT 2007

		Chinese						English
		Chars	ICT	LDC	Unigram	IU	GS	
Train:	Sentences	42942						
	R.W.	519928	380259	385426	393840	343696	396780	420431
	Vocabulary	2776	11760	9425	8802	13309	6226	9933
	Singletons	364	4637	2841	2629	4755	727	3937
Dev2:	Sentences	500						
	R.W.	4825	3578	3607	3682	3318	3740	3905
	Vocabulary	823	950	1021	987	1078	1004	834
	OOVs (R.W.)	7	75	52	49	17	16	216
	OOVs (voc.)	6	73	50	47	15	14	44
Dev3:	Sentences	506						
	R.W.	5155	3835	3845	3930	3583	4009	3965
	Vocabulary	837	936	996	969	1081	980	831
	OOVs (R.W.)	242	72	51	51	18	19	194
	OOVs (voc.)	20	69	48	48	16	15	45
Eval:	Sentences	489						
	R.W.	4365	3256	3268	3334	2994	3387	3836
	Vocabulary	762	885	944	915	1008	904	819
	OOVs (R.W.)	5	60	37	33	9	13	205
	OOVs (voc.)	5	59	36	32	9	12	33

Table 8: Corpus Statistics of task GALE 2007

		Chinese			English
Train:	Sentences	7567237			
	Running Words	93878360	94847485	92778460	101575014
	Vocabulary	111858	109852	121123	347436
	Singletons	38066	38115	38272	152173
Dev:	Sentences	2214			
	Running Words	47954	48584	47803	57735
	Vocabulary	6890	6520	6660	6372
	OOVs (running words)	17	17	18	230
	OOVs (in voc.)	13	13	14	158
Test:	Sentences	1943			
	Running Words	44340	44927	44281	53150
	Vocabulary	6779	6439	6597	6153
	OOVs (running words)	15	15	17	246
	OOVs (in voc.)	13	13	14	158

Table 9 shows the statistics of the word lengths in these three lexicons. We calculate the number of word entries, whose length is from 1 to 7 and larger than 7. For example, there are 2 334 words consisting of a single character in the LDC lexicon, 2 582 words in the learned lexicon and 1 941 words in the semi-supervised lexicon. These single character words represent 18.6% of the total number of entries in the LDC lexicon, 16.5% in the learned lexicon and 29.3% in the GS lexicon.

From this table we see that in the manual LDC lexicon more than 60% of the words consist of two characters and only about 15% of the words consist of three or four characters. Longer words with more than four characters are used seldom. Evidently, there are too many words with more than two characters in the learned dictionary. In the GS lexicon, the length distribution is similar to that in the LDC lexicon. There are about 15% word entries containing more than two characters. Figure 6 visualizes the statistics in Table 9. The horizontal axes show the word lengths and the vertical axes show the percentage of the word entries in the lexicon with a given length.

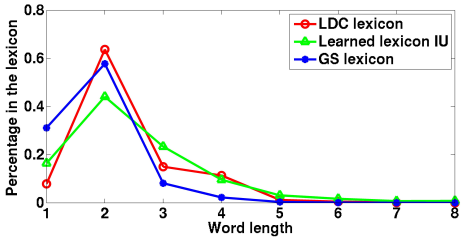

Figure 6: Curves of number of words given a length in the LDC lexicon, learned lexicon with alignment combination IU and semi-supervised lexicon using GS

There are frequencies for each word entries in the lexicon. If we take these frequencies into account, we obtain the expected total number of

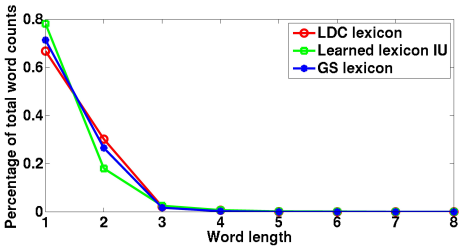

Figure 7: Curves of total number of words given a length in the LDC lexicon, learned lexicon with alignment combination IU and semi-supervised lexicon using GS

words given a length in the lexicon. The percentages are shown in Figure 7. For example, if the frequency of one two-character word entry is ten, we add ten to the two-character words count instead of one. Figure 7 shows that the word distribution in the manual LDC lexicon is closer to that in the GS lexicon than to the learned lexicon.

7.2 Evaluation results

We show the translation results of CWS methods in Table 10. This includes translation on characters, i.e. each character is taken as a word, LDC (LDC, 2003), CRF (Andrew, 2006), ICT (Zhang and Malik, 2003), unigram, 9-gram method described in Section 3, our alignment derived segmentation with three alignment combination approaches in Section 4 and our semi-supervised method (GS) in Section 5 as well as the integrated word segmentation in Section 6. The evaluations are performed under automatic criteria WER, PER, BLEU and TER (Snover et al., 2006) scores.

From Table 7.2 we see IU combination performs the best with respect to the translation result, among all the alignment combination methods, right, intersect and IU (Och, 2002). The integrated Chinese word segmentation performed on the baseline unigram segmented system leads to a consistent improvement in the translation performance with respect to the TER score. As more word segmentation alternatives are introduced, the average sentence length of translation using integrated approach is larger than that of the baseline approach, so that the BLEU scores decrease slightly. The GS is evaluated using our full model with both monolingual and bilingual information according to Section 5.3. The model weights λ are optimized using the Powell (Press et al., 2002) algorithm with respect to the BLEU score. We obtained $\lambda_1 = 1.4$, $\lambda_2 = 1$ and $\lambda_3 = 0.8$ as optimal values and $T = 4$ as the optimal number of iterations of re-alignment with GIZA++. The unigram method is implied to initialize the GS method and to segment the test corpus using a lexicon learned by GS. From the corpus statistics in Table 7 we observe that the vocabulary size of the Chinese training corpus is smaller in GS than in the baseline method, even though the number of running words is similar in both corpora. This shows that the distribution of Chinese words is more concentrated if using GS. In the final translation results, under all test sets and evaluation criteria, GS outperforms the other methods. The absolute WER decreases with 1.2% on Dev3 and with 1.1% on Eval data over baseline unigram method.

Table 11 shows the translation performance on the GALE 2007 task. In this table 'Unigram+GS X' means to linear interpolate the LDC lexicon and the lexicon generated by GS method with a weight X. For instance, 'Unigram+GS 0.5' indicates the combination method of semi-supervised CWS and unigram segmentation, where the probability of each word in the manual lexicon is interpolated with the probability of this word in the GS trained lexicon with a weight of 0.5. 'Unigram+GS 0.6' indicates the combined method with a weight of 0.6 for manual lexicon and 0.4 for GS trained lexicon. The combined method improved the performance with 0.5% in the BLEU score compared to the baseline method.

To measure the diversity of translation systems generated using different Chinese word segmentations, we performed leaving-one-system-out experiment. In Table 12 we show the translation performance by leaving a single system out of system combination and re-optimizing the weights on the Test08 data. The Dev08 data was used as a blind test set. The experiment was performed on the Newswire documents. We see that by adding the system with GS word segmentation, the BLEU score enhances with 0.4% and the TER decreases with 0.3% absolutely, which contributes more than the LDC system does to the final translation performance.

7.3 Effect of segmentation on translation results

We present some examples of translation outputs to show that the segmentation may have effect on the translation quality in Table 13. Three examples are selected from our automatic translations of the Eval corpus. For each of them we show the segmented Chinese source sentence using the baseline unigram, alignment derived segmentation and semi-supervised segmentation method (GS), as well as their corresponding translation and the human reference translation.

In the first example both GS and baseline methods lead to correct segmentations, while the learned segmentation results in an error, because 晚些should be separated into two words. 晚means 'late' and 些means 'a little'.

Table 10: Translation performance with different CWS methods on IWSLT 2007[%]

Test	Method	WER	PER	BLEU	TER
Dev2	Unigram (Baseline)	38.2	31.2	55.4	37.0
	derived-intersect	38.2	31.2	55.4	37.0
	GS	36.8	30.0	56.6	35.5
	integrated	37.4	30.5	55.9	36.1
Dev3	Unigram (Baseline)	33.5	27.5	60.4	32.1
	derived-intersect	32.9	27.0	60.4	31.7
	IU	32.3	26.7	61.1	31.2
	GS	32.3	26.6	61.0	31.4
	integrated	32.8	36.3	60.1	31.0
Eval	Characters	49.3	41.8	35.4	47.5
	LDC	46.2	40.0	39.2	45.0
	CRF	47.0	41.5	37.2	46.0
	ICT	45.9	40.4	40.1	44.9
	Unigram (Baseline)	46.8	40.2	41.6	45.6
	9-gram	46.9	40.4	40.1	45.4
	derived-right	47.4	42.2	38.5	46.5
	derived-intersect	46.5	40.1	39.5	45.3
	derived-IU	46.4	40.5	40.0	45.3
	GS	45.9	40.0	41.6	44.8
	integrated	44.9	39.4	41.0	43.5

Table 11: Translation performance with different CWS methods on GALE 2007[%]

Test	Method	WER	PER	BLEU	TER
Eval	LDC	73.0	49.5	28.2	67.1
	Unigram	73.0	49.7	28.4	67.2
	Unigram+GS 0.5	72.6	48.7	28.5	66.3
	Unigram+GS 0.6	72.5	48.6	28.7	66.3

Table 12: Translation performances of leaving one-out in system combination on GALE 2007[%].

Newswire	Test 08		Dev 08	
	BLEU	TER	BLEU	TER
Best single system	31.0	61.6	32.1	61.8
System combination	34.9	57.8	35.7	57.5
leave out rwth-pbt-LDC	35.2	57.7	35.6	57.6
leave out rwth-pbt-GS	34.6	58.0	35.3	57.8

In the second example the translation results in the learned and GS segmentation are closer to the reference translation. As 金or 额occurs more often than they are combined in the training corpus, it is easier to recognize the single character word in the evaluation text.

In the third example the segmentation with both learned method and baseline method made mistakes. For the learned method 请给 should be in two words, where 请 means 'please' and 给 means 'give'. Though, the baseline segmentation is reasonable for an human evaluation, the translation result is still erroneous, because the sequence of characters '可口可乐' never appears in the training corpus, but 可乐can be found many times. As both of them mean 'coke', we only need the word '可乐' to obtain the correct translation.

We compared all translation outputs on the Eval set using GS with the baseline method. 196 sentences are different out of 489 lines, at which 64 sentences from GS are better, 33 sentences are worse, and the remaining sentences have similar translation qualities.

Table 13: Examples of segmentation and translation outputs with baseline, alignment derived and GS segmentation.

a)	Baseline	请 告诉 我 总 金额 。 please show me the in .
	Learned-IU	请 告诉 我 总 金 额 。 please show me the total price .
	GS	请 告诉 我 总 金 额 。 please show me the total price .
	REF	can you tell me the total amount ?
b)	Baseline	请 给 我 可口可乐 。 please give me .
	Learned-IU	请给 我 可口 可乐 。 please give me a good coke .
	GS	请 给 我 可 口 可乐 。 please give me a coke .
	REF	coke , please .

7.4 Conclusions

We have successfully developed novel Chinese word segmentation methods for statistical machine translation. In the training process, Chinese word boundaries are learned jointly with the word alignments. Both monolingual and bilingual information are employed to derive a segmentation suitable for machine translation. New Chinese words and distributions are generated automatically. In the translation time multiple segmentation alternatives instead of the single-best segmentation are considered, and the segmentation decision is taken during the search for the best translation. Not only in a small, but also in a large data environment, our method outperformed the standard Chinese word segmentation approach in terms of the final Chinese to English translation quality.

References

Aldous, David J. 1985. Exchangeability and related topics. In *École d'été de probabilités de Saint-Flour, XIII-1983*, pages 1–198, Springer, Berlin.

Andrew, Galen. 2006. A hybrid markov/semi-markov conditional random field for sequence segmentation. In *Proceedings of EMNLP*, Sydney, July.

Blunsom, Phil and Miles Osborne. 2008. Probabilistic inference for machine translation. In *Proc. of the 2008 Conf. on Empirical Methods in Natural Language Processing*, pages 215–223, Honolulu, Hawaii, October.

Chen, Aitao, Yiping Zhou, Anne Zhang, and Gordon Sun. 2005. Unigram language model for Chinese word segmentation. *Proceedings of the 4th SIGHAN Workshop on Chinese Language Processing*, pages 138–141.

Goldwater, Sharon, Thomas L. Griffiths, and Mark Johnson. 2006. Contextual dependencies in unsupervised word segmentation. In *Proceedings of Coling/ACL*, Sydney, July.

IWSLT. 2005. International workshop on spoken language translation home page. http://www.is.cs.cmu.edu/iwslt2005/CFP.html.

IWSLT. 2007. International workshop on spoken language translation home page. http://www.slt.atr.jp/IWSLT2007.

Kanthak, Stephan and Hermann Ney. 2004. FSA: An efficient and flexible C++ toolkit for finite state automata using on-demand computation. pages 510–517, Barcelona, Spain, July.

LDC. 2003. Linguistic data consortium Chinese resource home page. http://www.ldc.upenn.edu/Projects/Chinese.

LDC. 2005. Linguistic data consortium resource home page. http://www.ldc.upenn.edu/Projects/TIDES.

Low, J. K., H. T. Ng, and W. Guo. 2005. General formulation and evaluation of agglomerative clustering methods with metric and non-metric distances. pages 161–164.

Luo, Xiaoqiang and Salim Roukos. 1996. An iterative algorithm to build Chinese language models. In *Proc. of the 34th annual meeting of the Associaton for Computational Linguistics*, pages 139–143, Santa Cruz, California.

Ney, Hermann. 1999. Speech translation: Coupling of recognition and translation. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing*, pages 517–520, Phoenix, AR, March.

Och, Franz J. and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):135–244, December.

Och, Franz J. 1999. An efficient method for determining bilingual word classes. In *EACL '99: Ninth Conf. of the Europ. Chapter of the Association for Computational Linguistics*, pages 71–76, Bergen, Norway, June.

Och, Franz J. 2000. Giza++: Training of statistical translation models. http://www-i6.informatik.rwth-aachen.de/~och/software/GIZA++.html.

Och, Franz J. 2002. *Statistical Machine Translation: From Single Word Models to Alignment Templates*. Ph.D. thesis, Computer Science Department, RWTH Aachen, Germany, October.

Papineni, Kishore A., Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, July.

Press, William H., Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. 2002. *Numerical Recipes in C++*. Cambridge University Press, Cambridge, UK.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of AMTA*, pages 223–231, Cambridge, MA, August.

Sproat, Richard and Chilin Shih. 1990. A statistical method for finding word boundaries in Chinese text. *Computer Processing of Chinese and Oriental Languages*, 4:336–351, April.

Stolcke, Andreas. 2002. SRILM - an extensible language modeling toolkit. In *Proceedings of the International Conference On Spoken Language Processing*, pages 901–904, Denver, Colorado, September.

Takezawa, Toshiyuki, Eiichiro 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.

Wang, Zhuoran and Ting Liu. 2005. Chinese unknown word identification based on local bigram model. *International journal of computer processing of oriental languages*, 18(3):185–196.

Xu, Jia, Richard Zens, and Hermann Ney. 2004. Do we need chinese word segmentation for statistical machine translation? In *Proceedings of the Third SIGHAN Workshop on Chinese Language Learning*, pages 122–128, Barcelona, Spain, July.

Xu, Jia, Evgeny Matusov, Richard Zens, and Hermann Ney. 2005a. Integrated chinese word segmentation in statistical machine translation. In *Proceedings of IWSLT 2005 (International Workshop on Spoken Language Translation)*, pages 141–147, Pittsburgh, PA, October.

Xu, Jia, Richard Zens, and Hermann Ney. 2005b. Sentence segmentation using IBM word alignment model 1. In *Proceedings of EAMT 2005 (10th Annual Conference of the European Association for Machine Translation)*, pages 280–287, Budapest, Hungary, May.

Xu, Jia, Jianfeng Gao, Kristina Toutanova, and Hermann Ney. 2008. Bayesian semi-supervised chinese word segmentation for smt. In *Proceedings of COLING*, pages 1017–1024, August.

Zhang, Hao and Jitendra Malik. 2003. Learning a discriminative classifier using shape context distances. In *CVPR 2003, Int. Conf. on Computer Vision and Pattern Recognition*, volume I, pages 242–247, Madison, WI, June.