A Novel Method for the N-best Generation of Phrased-Based SMT

Shui Liu, Sheng Li, Tiejun Zhao

Computer Science & Technology
Harbin Institute of Technology
150001, Harbin,China
liushui@mtlab.hit.edu.cn, lisheng@hit.edu.cn, tjzhao@mtlab.hit.edu.cn

Abstract:

Phrase based statistic MT (PBSMT) is an important milestone in SMT. In the PBSMT, finding proper weights is an important issue. Currently, the MERT is widely adapted to optimize the weights of the models for various SMT approaches. For PBSMT, the N-best generation is an important factor affecting the performance of MERT according to our experimental results. In previous method on N-best generation of PBSMT, the word lattice is employed which can reserve exponential in polynomial time. However, in this method, the hypotheses which are similar to the best hypotheses probably obtain high score as well, which may further encourages the similarity of the N-best. In this paper, a novel N-best generation method is proposed to retain considerable divergence for PBSMT. According the experimental results, our method is significant better than previous widely adopted method in Moses.

Keywords:

Statistical Machine Translation, PBSMT, N-best Generation

1 Introduction

Although the syntax based SMT models have been proved to be useful to MT according to recent researches(Wu 1997,Yamada and Knight 2001, Graehl and Knight 2004, Chiang 2005), PBSMT is still widely advocated as a promising model in MT. There are several advantages in PBSMT: first, it is efficient and robust in decoding because of the phrases in PBSMT extracted by simple, unmotivated, and structure constraints; second, even in syntax based SMT models (DeNeefe 2007, Cowan 2006), the phrase is important to translation quality, in which reserving some phrases cross linguistic analysis boundaries can yield significant performance improvement; at last, the linguistic sense features (Marton 2008, Xiong 2009, Zhang 2009, Liu 2010) can be added to this kind of string models (DeNeefe 2007) as soft-constraint to compensate the absent of linguistic analysis.

Decoding of PBSMT is a NP-hard problem according to (Koehn and Och 2003). In the sake of efficiency, pruning is practically necessary. In single best decoding, the complexity is reduced to polynomial time with joint use of histogram pruning, beam-search and

recombine. On the other hand, the pruned hypotheses in single best decoding might appear in the N-best translations, which is required by the training of log-linear model. Fortunately, N-best translation can be reserved in polynomial time by applying word lattice in decoding. In the N-best translations, the hypotheses which are similar to the best hypotheses probably obtain high score as well, which may further encourages the similarity of the N-best. However, theoretically speaking, the tuning of log-linear model can benefit from the N-best with considerable divergence. Thus, we propose a novel N-best generation approach which can skip some similar hypotheses in N-best generation. In this research, the hypotheses with same preconditions of recombine (also mentioned as state and to be explained in section 2) are defined as similar. In order to retain considerable divergence in N-best, we only reserve the top k ($k \leqslant N$) hypotheses of the same state in N-best decoding. By doing so, the performance of baseline is substantially improved according to our experimental results.

The left chapters are organized as follows: the baseline model is proposed in section 1; single best search of PBSMT are discussed briefly in section 2; our N-best generation algorithm is proposed in section 3; the experimental results is proposed with discussion in section 4; at last, the conclusion of this paper is proposed in section 5.

2 Model

According to Koehn (2003), the phrase based model is based on a noisy channel model. Given source language f, the decoder is to find the translation e, which satisfies:

$$e = \mathbf{ARGMAX}_e \, p(f|e) \, p(e) \tag{1}$$

where $p(e)$ is language model and $p(e \mid f)$ is translation model. To avoid the preference of short sentence, the length penalty is induced.

$$e = \mathbf{ARGMAX}_e \, p(f|e) \, pLM(e) \, \omega^{length(e)} \tag{2}$$

In Moses (Koehn 2007) default settings, $p(f|e)$ can be decomposed to phrase model, lexical reordering model, distortion model. Among these models, the phrase model is related to the word translation, while the lexical reordering and the distortion are in charge of the reordering in translation.

The phrase model of Moses have five scores, including: the phrase probability of e conditioned on f , $p(e|f)$; the inverse phrase probability : $p(f|e)$; the lexical weight of e respect to f , $p_{lex}(e|f)$; the lexical weight of f with respect to e , $p_{lex}(f|e)$; the phrase penalty, π.

The distortion models the different ordering between source and target language. The score of the model is the sum of the number of words skipped in decoding. The lexical reordering model divide the reordering type into 3 types: monotone, swap, and discontinuous. The score of each reordering type is computed by the max-likelihood estimation with respect to the lexical occurrence of the source and target phrase.

Log-linear model combines these mentioned models together, in which the weight of the models can be trained by MERT:

$$p(e \mid f) = \frac{\exp \sum_{m=1}^{M} \lambda_m h_m(e, f)}{\sum_{e'} \exp \sum_{m=1}^{M} \lambda_m h_m(e', f)} \tag{3}$$

where λ is the weight of the model score h_m(e,f).

3 Single best search

The decoding algorithm is always considered as a searching problem. According to Kohn (2003), the upper boundary of the number of searching paths (the possible translation) for a given source sentence is $2^{nf}|V_e|^2$, where $|V_e|$ is the size of target language vocabulary, n_f is the number of source word, provided the model is based on trigram language model. To reduce the cost of decoding, risk free and risk prune are introduced.

3.1 risk prune

The translation is generated left to right as in decoding. Since the number of search paths (hypotheses) is large, it is expensive to record all the possible paths in decoding. To make the cost of decoding acceptable, the decoder organizes the hypotheses by stacks with respect to the number of words translated. Hence, in each stack, hypotheses may cover different words. Thus, these hypotheses are not comparable in the translation cost. Then, future cost is introduced to estimate the score of the uncovered words. Using future cost, hypotheses covered different words are comparable.

As mentioned above, the score of hypothesis h can be computed as S(h) , which can be further decomposed as:

$$S(h) = H(h) + \mathcal{F}(h) \tag{4}$$

where H(h) is the cost of the hypothesis and $\mathcal{F}$(h) is future cost. The score of hypothesis is the translation score, i.e. p(e|f) . While the future cost is the estimation of the uncovered words using dynamic programming.

Each stack in decoding retains certain number of hypotheses, the inferior hypotheses are pruned out. By doing this, the decoding cost is reduced to polynomial time. This approach is known as histogram prune.

Moreover, the inferior hypotheses are discarded to further relieve the cost of decoding. Any hypothesis whose score (estimated by equation 4) is lower than a certain threshold is considered as inferior hypothesis. This pruning approach is known as beam-search. In beam-search, the threshold is computed as a relative score which is a factor α (e.g. 0.0001) of the best hypothesis in the same stack.

3.2 risk free prune

In decoding, the decoder keeps on extending partial hypotheses by adding new phrases to the end of the hypotheses. In this process, it is noticed that some hypotheses are always inferior to others. That is: given two hypotheses h_a and h_b which cover the same words in source, it is satisfied that the cost of extending h_a by any phrase is always lower than the

cost of extending h_b by the same phrase. Then pruning out h_a don't change the result in single best search.

In decoding, the score of hypotheses extending contains: the inner scores of the phrase and the hypothesis to be extended, and the score of the extending. The inner score is the cost of the partial score of the hypothesis or the phrase. In risk free prune, h_a is considered as inferior to h_b, if h_a and h_b satisfy the following conditions: the inner score of h_a are lower than h_b, and the extending cost of extending h_a and h_b by a same phrase is the same.

To identify the inferior hypotheses, the hypotheses are organized by state which is the precondition of recombine, and hypotheses with the same state have the same extending cost. In Moses, the state is:

1. the word coverage-vector in source: C
2. the first and end position of last phrase in the hypothesis: i , j
3. the last m gram in target: e_m

Thus, the state F can be represented as a 4-tuple <C, i , j , e_m>. The states of hypotheses with same coverage-vector C can be extended by the same set of phrases. The i and j in state are related to the score of distortion model and lexical reordering model, while the e_m are related with the score of the language model. Thus, hypotheses with same states have the same potential extending cost (the bi-direction lexical reordering model is an exception, and we will discuss this issue in next section). The translation stack of the decoder is a set of states: $\cup_{|C|=i} \{ < C, i , j, e_m >\}$.

In single best decoding, many hypotheses belonged to the same state $F = < C, i, j, e_m >$ are generated in decoding, only the best one is reserved: $h = \mathrm{ARGMAX}_{h \in F} S (h)$

4 N-best search

In Moses, the N-best translations are generated by word lattice. Using the word lattice, the decoder reserves exponential search paths in a compact data structure. The N-best decoding of Moses adopts the approach as in [Ueffing and Och., 2002] which applies A* search in the word lattice, in which the N-best search is considered as a find-K-shortest-paths problem and is carried out in two phases. In the first phase, the decoder build a word lattice. In the second phase, the top one best is found in the lattice and added to the N-best list. Then, the successors of the 1-best are added to a priory queue. This process keeps on popping the top element of the queue to the N-best list and adding the successors of the popped element to the queue, until the N-best list is full.

4.1 Formulation

As discussed in section INTRODUCTION, we consider the similarity of N-best may be encouraged in above method. To retain considerable divergence of N-best, some similar hypotheses in N-best decoding is skipped in our approach. In this paper, we consider the hypotheses with same state are similar. Our original idea is to only reserve the top k hypotheses for each state, and k is mentioned as N-best factor in follows.

However, in a simple way, if the decoder extends all hypotheses in a state, the cost of decoding will increase by k-1 times than single best decoding. Fortunately, the extending cost of hypotheses in the same state are (almost) the same. Thus, we can extend merely one

hypothesis (the best hypothesis) of each state and record the extending history (to be formally defined in follows), then the expansion cost of the rest hypotheses in the state can be estimated using the extending history, rather than estimate the extending cost independently. By doing this, the cost of top-N search is largely reduced.

Formally, the extending of hypotheses can be represent as a deduction. In the deduction, the tail nodes are a hypothesis h and a phrase-pair p, and the head node is the hypothesis h' generating from extending h by p. Each hypothesis in the deduction is proposed as a pair (w, F) where w is the cost of the hypothesis and F is the state of the hypothesis. Thus, as in (Huang et. al, 2006), hypothesis h extended by a phrase-pair p can be formulized as:

$$F^{h'} \xleftarrow{p} F^{h} = \frac{p : (H(h), < \mathrm{C}^{\mathrm{h}}, \mathrm{i}^{\mathrm{h}}, \mathrm{j}^{\mathrm{h}}, \mathrm{e}^{\mathrm{h}}_{\mathrm{m}} >)}{(H(h'), < \mathrm{C}^{\mathrm{h}} \cup \{\mathrm{i}^{\mathrm{p}} .....\mathrm{j}^{\mathrm{p}}\}, \mathrm{i}^{\mathrm{p}}, \mathrm{j}^{\mathrm{p}}, \mathrm{e}^{\mathrm{h}}_{\mathrm{m}} \oplus \mathrm{e}^{\mathrm{p}} >)} \tag{5}$$

where $H_i(F^{h'} \xleftarrow{p} F^{h})$ are the cost of hypothesis h with respect to the model, the function $\oplus$ denotes the update of the language history after extending h by p which is faithful to (Zens and Ney, 2006) , and is the cost of the new generated hypothesis, which could be computed as:

$$H(h') = H(h) + H(p) + \Delta H(F^{h'} \xleftarrow{p} F^{h}) \tag{6}$$

where H(h) and H(p) are called inner cost , and $\Delta H(F^{h'} \xleftarrow{p} F^{h})$ is extending cost which is generated by the deduction and can be computed by:

$$\Delta H(F^{h'} \leftarrow F^{h}) = H(h') - H(h) = \sum_{i} H_i(F^{h'} \xleftarrow{p} F^{h}) \tag{7}$$

where $H_i(F^{h'} \xleftarrow{p} F^{h})$ is the score of i[th] sub-model generated by the transition of states.

The equation (6) divide the deduction scores into three parts, which is based on the assumption that the extending cost is only related with the transition of state . In some models, the extending cost are not only decided by the transition of the state i.e. $F^{h'} \xleftarrow{p} F^{h}$, but also the features which is not included in the state. In this case, the recombine of the hypothesis is not totally risk free, that the pruned paths may be better than the reserved ones because of the model score computed by features which are absent in the state. For example: in the bi-direction lexical reordering model, as proposed in (Koehn 2005), the score of the current phrase-pair's reordering is not only modeled on the phrase-pair, but also on the phrase-pair left to it. The equation (6) should be expanded as:

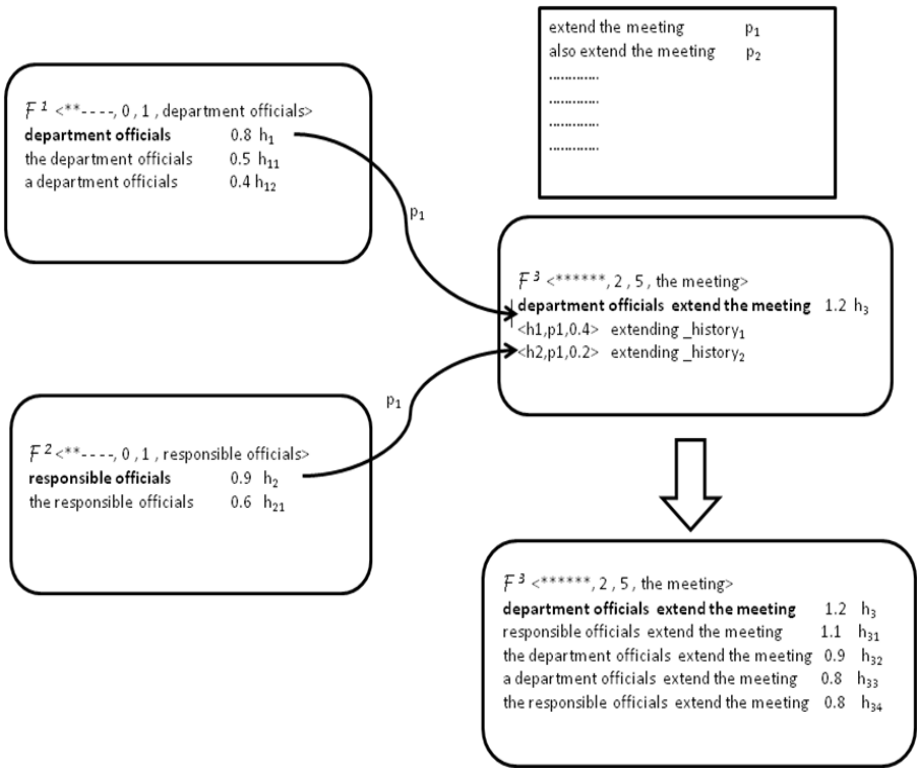
$$H(h') = H(h) + H(p) + \Delta H(F^{h'} \xleftarrow{p} F^{h}) + \xi(h, p) \tag{8}$$

where $\xi(h, p)$ is the score of the model in which the features are not included in state and is also considered as extending cost.

The equation (8) is a generalized model, and models satisfying equation (6) can be considered as a special case if $\xi(h, p) = 0$. In equation (6) and (8), the $\Delta H(F^{h'} \xleftarrow{p} F^{h})$ is the common part of the extending cost. Recoding it can reduce the decoding time to a large extend, especially for model satisfying (6).

Thus, to utilize the common part in N-best decoding, the extending history is recorded, which is a 3-tuple $< h, p, \Delta H(F^{h'} \xleftarrow{p} F^{h}) >$, where h is the previous hypothesis of h', and p is last phrase-pair in the deduction generating h'.

(9)



By using equation (8) and extending history, the cost of extending hypothesis h by phrase p can be easily estimated, if some hypotheses h* has the same state with h and used to be extended by p. In the follows, this method is mentioned as hidden hypothesis generating.

An example of the state based deduction and hidden hypotheses generating for an trigram language model as shown in (9). The rectangles with smooth angles represent states, while the phrase table is shown as rectangle with straight angles. A "*" in the sate item C shows that the source position is covered by the hypothesis. In each state, the best hypothesis is highlighted with bold line.

4.2 N-best Decoding

An example of the hidden hypothesis generating is shown in (9), in which there are 3 hypotheses in the state F^{1} : h_1 ,h_{11} , and h_{12} , of which h_1 is the best hypothesis with hypothesis cost 0.8. The generation of the state F^{3} in N-best search can be divided with 2 phases: extending phase and extending update phase. Firstly, in the extending phase, the h_3 is generated by extending h_1 with p_1, and the extending history is recorded as extending_history1. Then, in the extending update phase, by using $extending\_history_1$, the hidden hypothesis h_{32} and h_{33} are generated (h_{31} and h_{34} are generated in a similar case) according to equation (8). Note that, in extending phase, the best hypothesis is reserved as a hypothesis and an extending history, while the left hypotheses are only reserved as extending history. Another point is that the hypotheses in each state are divided into 2 groups: best hypothesis and inferior hypotheses. Briefly, the decoder first extends the best hypothesis of each state, then uses the extending history of the best and inferior hypotheses to generate all the other hidden hypotheses of each state. The process which generates all the hidden hypotheses of a state is called extending update in this study. After extending update, the N-best hypothesis of state is generated. Thus, the formal description of decoding for the N-best search can be described in (10):

(10)

```
0   Procedure  N-best decoding
1   foreach phrase in phrase-table
2     if distortion_prune(phrase)
3       continue
4     else
5       hypothesisStack[num_of_covered_words(phrase)]←state(phrase)
6     end if
7   end for
8   for i = 0 to n_f-1
9     foreach state in hypothesisStack [ i ]
10      extending_update( state )      // extending update phase
11      prune_state_size ( state)
12      foreach phrase in phrase-table
13        if distortion_prune(phrase, best_hypo_of_state(S))
14          continue
15        else
16          hypo' ← hypothesis_extending(phrase , best_hypo_of_state( S ))
                                                  // extending phase
17          add _to_stack(hypo')
18          prune_stack(i)
19        end if
20      end for
21    end for
22  end for
```

In (10), The function *extending_update*(state) generates all the hidden hypothesis of state as shown in Fig. 2. The function *prune_state_size*(state) only reserves the top the top k

hypotheses of state, where the k is N-best factor. The function *hypothesis_extending*(phrase, hypo) generates new hypothesis h' by extending hypo by phrase, the *prune_stack*(i) prunes the ith stack according to recombine and histogram pruning. The *add_to_stack*(hypo) is as follows:

(11)

```
0  procedure add_to_stack (hypo)
1  num ← num_of_covered_words( hypo )
2  S ← find_equal_state (hypo)
3  If S = NULL
4    hypothesisStack [ num] ← S
5  else
6    if score (hypo) > score (S.best_hypo_of_state)
7      S.extending_history_list ← extending_history(S.best_hypo_of_state)
8      S.best_hypo_of_state←hypo
9    end if
10   extending_history_list(S) ← extending_history(hypo)
11 end if
```

In (11), the function *find_equal_state*(hypo) returns the state of hypo in current stack, and *NULL* indicates that the state doesn't exist in stack. The function *extending_history_list*(h) returns the extending history of h. state.extending_history reserves the extending histories of inferior hypotheses in state. The state.best_hypo_of_state is the best hypo of state.

The (10) and (11) show our decoding algorithm for N-best search. Most part of our algorithm is (very) similar with the sing best search decoding proposed in (Koehn, 2003). The mainly difference is that the hidden paths are generated by using the inferior hypotheses in states instead of word lattice. Furthermore, in the sake of efficiency, the hypotheses in a state are pruned before extending the best hypothesis of the state, as shown in line 11 of (10). After the N-best decoding, we output top N of the hypotheses in last stack, in which the total number of hypotheses is $k \times t$, where t is the number of states in each stack and k is number of hypotheses reserved for each state.

5 Experiment

We perform Chinese-to-English translation task on NIST MT-03 and MT-05 test set, and use NIST MT-02 as our tuning set. FBIS corpus is selected as our training corpus, which contains 7.06M Chinese words and 9.15M English words. We use GIZA++ (Och and Ney, 2000) to make the corpus aligned. A 4-gram language model is trained using Xinhua portion of the English Gigaword corpus (181M words). All models are tuned and evaluated on BLEU with case insensitive.

In decoding, we drop all the OOV words in tuning and test, and the default decoding settings are used: set the distortion limitation with 6, beam-width with 1/100000, stack size

with 200 and max number of phrases for each span with 50. In training, we output 100 best for MERT, i.e. N=100.

In our system, there are 5 groups of features. They are:

1. Language model score
2. word penalty score
3. phrase model scores
4. distortion score
5. lexical RM scores

Using the default setting, we execute our N-best decoding (as shown in Fig. 11-13) with different N-best factor m varying from 20 to 100, the BLEU of Dev. set and test set are proposed:

(12)

	$viterbi_{con}$			$viterbi_{incon}$		
k	NIST02	NIST03	NIST05	NIST02	NIST03	NIST05
20	31.67	25.94	27.30	33.24	26.85	28.45
40	31.61	26.17	27.30	33.25	**28.18**	**28.59**
60	31.75	**26.94**	**27.56**	33.38	27.28	28.57
80	31.27	26.39	27.50	33.33	27.29	28.31
100	31.69	25.34	27.07	33.15	26.96	28.01

In (12), $viterb_{icon}$ contains feature group: 1, 2, 3, 4; $viterbi_{incon}$ contains the feature group: 1, 2, 3, 4, 5. These experimental results shows that the BLEU scores of $viterbi_{con}$ and $viterbi_{incon}$ varies in a large range on NIST-03: the height drop between highest and lowest in $viterb_{icon}$ is 1.00 BLEU score, and in $viterbi_{incon}$ is 1.33 BLEU score. Moreover, when m = 100, $viterb_{icon}$ and $viterbi_{incon}$ nearly achieve the lowest point on both test set, which is because the maximum number of hypotheses in each state are reserved, in other words, the divergence of N-best is the least. On the other hand, the best performance is obtained when k is around 50 on test set with the two models, respectively, from which we can conclude that keeping considerable divergence of N-best can benefit the MERT.

To compare our method with N-best generation method in Moses, we carry out training and test under the same model and decoding settings as shown in (13). Baseline contains feature groups 1-4 and $Baseline_{rm}$ contains feature groups 1-5.

(13)

corpus	02	03	05
baseline	31.58	25.41	27.07
$viterbi_{con\ (k=60)}$	31.75	26.94	27.56
$viterbi_{con\ (k=100)}$	31.69	25.34	27.07
$baseline_{rm}$	33.27	26.54	27.58
$viterbi_{incon\ (k=40)}$	33.25	28.18	28.59

$viterbi_{incon\ (k=100)}$	33.15	26.96	28.01

In (13), our approaches outperform the baseline systems on 2 test set with two different models. The performance of baseline is close to the performance of our N-best search with k = 100. It implies that skipping some similar hypotheses do benefit the MERT, especially for $viterbi_{incon}$.

6 Conclusion

A novel N-best generation method is proposed in this paper, in which considerable divergence of N-best is retained. According to the experimental results, our N-best generation method obtains significant improvement. Then, we conclude that retaining considerable convergence of N-best in PBSMT do benefit the MERT in our corpus.

Exploring the N-best generation for PBSMT is a promising direction in our future work. According to our experimental result, skipping some similar hypotheses can yield substantially performance improvement, which indicates that the similarity of N-best affects the tuning of log-linear model. On this direction, the definition of similar can be extended with different definitions instead of state.

7 Acknowledgement

The work of this paper is funded by National Natural Science Foundation of China (grant no. 60736014), National High Technology Research and Development Program of China (863 Program) (grant no. 2006AA010108), and Microsoft Research Asia IFP (grant no. FY09- RES-THEME-158).

8 References

Chiang, David. (2005). A hierarchical phrase-based model for SMT. ACL-05.263-270.

Chiang, David. (2007). Hierarchical phrase-based translation. Computational Linguistics, 33(2):201-228.

Cowan, B., Kucerova, I., and Collins, M. 2006. A discriminative model for tree-to-tree translation. In Proc. EMNLP.

DeNeefe, S. and Knight, K. (2007). What can syntax-based MT learn from phrase-based MT ? In Proc. EMNLP-CoNULL.

Deyi Xiong, Min Zhang, Aiti AW and Haizhou Li. 2009.A Syntax-Driven Bracket Model for Phrase-Based Translation. ACL-09.315-323.

Koehn, Philipp. (2007) Moses: Open Source Toolkit for Statistical Machine Translation, ACL 2007.

Koehn, Philipp and Och, Franz Joseph. (2003). Statistical Phrase-based Translation. In Proceedings of HLT-NAACL.

Koehn, Philipp and Axelrod, Amittai. (2005). Edinburgh System Description for the 2005 IWSLT Speech Language Translation Evaluation. In Proceedings of International Workshop on Spoken Language.

Liu, S and Li, S. (2010). head-modifier relation based non-lexical reordering model. coling 2010.

Marton, Yuval and Resnik, Philip. (2008). Soft syntactic Constraints for Hierarchical Phrased-based Translation. ACL-08. 1003-1011.

Och, F.J. and Ney, H. (2000). Improved statistical alignment models. In Proceedings of ACL 38.

Ueffing, Nicola and Och, Franz. (2002). Generation of Word Graph in Statistic Machine Translation. In proceedings of EMNLP 2002.

Wu, Dekai. (1997). Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. Computational Lingustics,23(3):377-403.

Zens, Richard and Ney, Hermann. (2006). Improvements in Dynamic Programming Beam Search for Phrase-based Statistical Machine Translation. In Proceedings of Workshop on Computationally Hard Problems and Joint Inference In Speech and Language processing.