Optimizing Word Alignment Combination For Phrase Table Training

Yonggang Deng *and* **Bowen Zhou**
IBM T.J. Watson Research Center
Yorktown Heights, NY 10598, USA
{ydeng,zhou}@us.ibm.com

Abstract

Combining word alignments trained in two translation directions has mostly relied on heuristics that are not directly motivated by intended applications. We propose a novel method that performs combination as an optimization process. Our algorithm explicitly maximizes the effectiveness function with greedy search for phrase table training or synchronized grammar extraction. Experimental results show that the proposed method leads to significantly better translation quality than existing methods. Analysis suggests that this simple approach is able to maintain accuracy while maximizing coverage.

1 Introduction

Word alignment is the process of identifying word-to-word links between parallel sentences. It is a fundamental and often a necessary step before linguistic knowledge acquisitions, such as training a phrase translation table in phrasal machine translation (MT) system (Koehn et al., 2003), or extracting hierarchial phrase rules or synchronized grammars in syntax-based translation framework.

Most word alignment models distinguish translation direction in deriving word alignment matrix. Given a parallel sentence, word alignments in two directions are established first, and then they are combined as knowledge source for phrase training or rule extraction. This process is also called symmetrization. It is a common practice in most state of the art MT systems. Widely used alignment models, such as IBM Model serial (Brown et al., 1993) and HMM , all assume one-to-many alignments. Since many-to-many links are commonly observed in natural language, symmetrization is able to make up for this modeling limitation. On the other hand, combining two directional alignments practically can lead to improved performance. Symmetrization can also be realized during alignment model training (Liang et al., 2006; Zens et al., 2004).

Given two sets of word alignments trained in two translation directions, two extreme combination are intersection and union. While intersection achieves high precision with low recall, union is the opposite. A right balance of these two extreme cases would offer a good coverage with reasonable accuracy. So starting from intersection, gradually adding elements in the union by heuristics is typically used. Koehn et al. (2003) grow the set of word links by appending neighboring points, while Och and Hey (2003) try to avoid both horizontal and vertical neighbors. These heuristic-based combination methods are not driven explicitly by the intended application of the resulting output. Ayan (2005) exploits many advanced machine learning techniques for general word alignment combination problem. However, human annotation is required for supervised training in those techniques.

We propose a new combination method. Like heuristics, we aim to find a balance between intersection and union. But unlike heuristics, combination is carried out as an optimization process driven by an effectiveness function. We evaluate the impact of each alignment pair w.r.t. the target application, say phrase table training, and gradually add or remove the word link that currently can maximize the predicted benefit measured by the effectiveness function. More specifically, we consider the goal of word alignment combination is for phrase table training, and we directly motivate word alignment combination as a process of maximizing the number of phrase translations that can be extracted within a sentence pair.

2 Combination As Optimization Process

Given a parallel sentence $(\mathbf{e} = e_1^I, \mathbf{f} = f_1^J)$, a word link is represented by a pair of indices (i, j),

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 229–232,
Suntec, Singapore, 4 August 2009.

which means that Foreign word f_j is aligned with English word e_i. The direction of word alignments is ignored. Since the goal of word alignment combination is for phrase table training, we first formally define a phrase translation. Provided with a set of static word alignments $\mathbf{A}$, a phrase pair $(e_{i_1}^{i_2}, f_{j_1}^{j_2})$ is considered translation of each other if and only if there exists at least one word link between them and no cross phrase boundary links exist in $\mathbf{A}$, i.e., for all $(i,j) \in \mathbf{A}, i \in [i_1, i_2]$ iff $j \in [j_1, j_2]$. Notice that by this definition, it does not matter whether boundary words of the phrase pairs should be aligned or not. Let $PP_n(\mathbf{A})$ denote the set of phrase pairs that can be extracted with $\mathbf{A}$ where up to n boundary words are allowed to be not-aligned, i.e., aligned to empty word NULL. As can be imagined, increasing n would improve recall of phrase table but likely to hurt precision. For word alignment combination, we focus on the set with high accuracy where $n = 0$.

Let $\mathbf{A}_1$, $\mathbf{A}_2$ denote two sets of word alignments to be combined for the given sentence pair. For instance, $\mathbf{A}_1$ could be word alignments from English to foreign while $\mathbf{A}_2$ the other direction. On different setup, $\mathbf{A}_1$ could be Model-4 alignments, while $\mathbf{A}_2$ is from HMM. In the first combination method we presented in Algorithm 1, we start with intersection $\mathbf{A}_I$. $\mathbf{A}_c$ is the candidate link set to be evaluated and appended to the combined set $\mathbf{A}$. Its initial value is the difference between union and intersection. We assume that there is an effectiveness function $g(\cdot)$ which quantitatively measures the 'goodness' of a alignment set for the intended application. A higher number indicates a better alignment set. We use the function g to drive the process. Each time, we identify the best word link $(\hat{i}, \hat{j})$ in the candidate set that can maximize the function g and append it to the current set $\mathbf{A}$. This process is repeated until the candidate set is empty or adding any link in the set would lead to degradation. Finally (line 15 to 21), we pickup word links in the candidate set to align those uncovered words. This is applied to maximize coverage, which is similar as the 'final' in (Koehn et al., 2003). Again, we use the function $g(\cdot)$ to rank the word links in $\mathbf{A}_c$ and sequentially append them to $\mathbf{A}$ depending on current word coverage.

The algorithm clearly is a greedy search procedure that maximizes the function g. Since we plan to take the combined word alignments for phrase translation training, a natural choice for g is the number of phrase pairs that can be extracted with the given alignment set. We choose $g(\mathbf{A}) = |PP_0(\mathbf{A})|$, where we only count phrase pairs that all boundary words are aligned. The reason of putting a tight constraint is to maintain phrase table accuracy while improving the coverage. By keeping track of the span of currently aligned words, we can have efficient implementation of the function g.

Algorithm 1 Combination of $\mathbf{A}_1$ and $\mathbf{A}_2$ as an Optimized Expanding Process

1: $\mathbf{A}_I = \mathbf{A}_1 \cap \mathbf{A}_2$, $\mathbf{A}_U = \mathbf{A}_1 \cup \mathbf{A}_2$
2: $\mathbf{A} = \mathbf{A}_I$, $\mathbf{A}_c = \mathbf{A}_U - \mathbf{A}_I$
3: $total = g(\mathbf{A})$
4: **while** $\mathbf{A}_c \neq \emptyset$ **do**
5: $\quad curMax = \max_{(i,j)\in\mathbf{A}_c} g(\mathbf{A} \cup \{(i,j)\})$
6: $\quad$ **if** $curMax \geq total$ **then**
7: $\quad\quad (\hat{i}, \hat{j}) = \text{argmax}_{(i,j)\in\mathbf{A}_c} g(\mathbf{A} \cup \{(i,j)\})$
8: $\quad\quad \mathbf{A} = \mathbf{A} \cup \{(\hat{i}, \hat{j})\}$
9: $\quad\quad \mathbf{A}_c = \mathbf{A}_c - \{(\hat{i}, \hat{j})\}$
10: $\quad\quad total = curMax$
11: $\quad$ **else** {adding any link will make it worse}
12: $\quad\quad$ break
13: $\quad$ **end if**
14: **end while**
15: **while** $\mathbf{A}_c \neq \emptyset$ **do**
16: $\quad (\hat{i}, \hat{j}) = \text{argmax}_{(i,j)\in\mathbf{A}_c} g(\mathbf{A} \cup \{(i,j)\})$
17: $\quad$ **if** $e_{\hat{i}}$ is not aligned or $f_{\hat{j}}$ is not aligned **then**
18: $\quad\quad \mathbf{A} = \mathbf{A} \cup \{(\hat{i}, \hat{j})\}$
19: $\quad$ **end if**
20: $\quad \mathbf{A}_c = \mathbf{A}_c - \{(\hat{i}, \hat{j})\}$
21: **end while**
22: return $\mathbf{A}$

Alternatively, the optimization can go in opposite direction. We start with the union $\mathbf{A} = \mathbf{A}_U$, and gradually remove the worse word link $(\hat{i}, \hat{j}) = \text{argmax}_{(i,j)\in\mathbf{A}_c} g(\mathbf{A} - \{(i,j)\})$ that could maximize the effectiveness function. Similarly, this shrinking process is repeated until either candidate set is empty or removing any link in the candidate set would reduce the value of function g.

Other choice of 'goodness' function g is possible. For instance, one could consider syntactic constraints, or weight phrase pairs differently according to their global co-occurrence. The basic idea is to implement the combination as an iterative customized optimization process that is driven by the application.

3 Experimental Results

We test the proposed new idea on Persian Farsi to English translation. The task is to translate spoken Farsi into English. We decode reference transcription so recognition is not an issue. The training

data was provided by the DARPA TransTac program. It consists of around 110K sentence pairs with 850K English words in the military force protection domain. We train IBM Model-4 using GIZA++ toolkit (Och and Ney, 2003) in two translation directions and perform different word alignment combination. The resulting alignment set is used to train a phrase translation table, where Farsi phrases are limited to up to 6 words.

The quality of resulting phrase translation table is measured by translation results. Our decoder is a phrase-based multi-stack implementation of the log-linear model similar to Pharaoh (Koehn et al., 2003). Like other log-linear model based decoders, active features in our translation engine include translation models in two directions, lexicon weights in two directions, language model, lexicalized reordering models, sentence length penalty and other heuristics. These feature weights are tuned on the dev set to achieve optimal translation performance evaluated by automatic metric. The language model is a statistical 4-gram model estimated with Modified Kneser-Ney smoothing (Chen and Goodman, 1996) using only English sentences in the parallel training data.

3.1 Phrase Table Comparison

We first study the impact of different word alignment combination methods on phrase translation table, and compare our approaches to heuristic based methods. The same English to Farsi and Farsi to English Model-4 word alignments are used, but we try different combination methods and analysis the final alignment set and the resulting phase translation table. Table 1 presents some statistics. Each row corresponds to a particular combination. The first two are intersection (I) and union (U). The next two methods are heuristic (H) in (Och and Ney, 2003) and grow-diagonal (GD) proposed in (Koehn et al., 2003). Our proposed methods are presented in the following two rows: one is optimization as an expanding process (OE), the other is optimization as an shrinking process (OS). In the last four rows, we add 'final' operation (line 15 to 21 in Algorithm 1).

For each method, we calculate the output alignment set size as a percentage of the union (the 2nd column) and resulting phrase table ($PP_n(\mathbf{A})$) size (in thousand) with different constrain on the maximum number of unaligned boundary words $n = 0, 1, 2$ (the next 3 columns). As we can see, the intersection has less than half of all word links in the pool. This implies the underlying word alignment quality leaves much room for improvements, mainly due to data sparseness. Not surprisingly, when relaxing unaligned boundary word number from 0 to 2, the phrase table size increases more than 7 times. This is the result of very low recall of word alignments, consequently the estimated phrase table $PP_2(\mathbf{A})$ has very low accuracy. Union suffers from the opposite problem: many incorrect word links prevent good phrase pairs from being extracted.

The two heuristic methods and our proposed optimization approaches achieve somewhat a balance between I and U. By comparing size of $PP_0(\mathbf{A})$ (3rd column), optimization methods are able to identify much more phrase pairs with similar size of alignment set. This confirms that the new method is indeed moving to the desired direction of extracting as many accurate (all boundary words should be aligned) phrase pairs as possible. We still notice that ratio of $|PP_2(\mathbf{A})|$ and $|PP_0(\mathbf{A})|$ (the last column) is high. We suspect that the ratio of this two phrase table size might somewhat be indicative of the phrase table accuracy, which is hard to estimate without manual annotation though.

Method	$\frac{\lvert\mathbf{A}\rvert}{\lvert\mathbf{A}_U\rvert}$	$\lvert PP_0\rvert$	$\lvert PP_1\rvert$	$\lvert PP_2\rvert$	$\frac{\lvert PP_2\rvert}{\lvert PP_0\rvert}$
I	45%	424	2047	3658	8.63
U	100%	354	555	578	1.63
H	78%	538	1225	1519	2.82
GD	82%	499	1081	1484	2.97
OS	84%	592	1110	1210	2.04
OE	78%	659	1359	1615	2.45
HF	95%	427	670	697	1.63
GDF	97%	412	647	673	1.63
OSF	89%	484	752	781	1.61
OEF	89%	476	739	768	1.61

Table 1: Statistics of word alignment set and the resulting phrase table size (number of entries in thousand (K)) with different combination methods

3.2 Translation Results

The ultimate goal of word alignment combination is for building translation system. The quality of resulting phrase tables is measured by automatic translation metric. We have one dev set (1430 sentences with 11483 running words), test set 1 (1390 sentences with 10334 running words) and test set 2 (417 sentences with 4239 running words). The dev set and test set 1 are part of all available Farsi-

English parallel corpus. They are holdout from training data as tuning and testing. The test set 2 is the standard NIST offline evaluation set, where 4 references are available for each sentence. The dev and test set 1 are much closer to the training set than the standard test set 2. We tune all feature weights automatically (Och, 2003) to maximize the BLEU (Papineni et al., 2002) score on the dev set.

Table 2 shows BLEU score of different combination methods on all three sets. Union performs much worse on the dev and test1 than intersection, while intersection achieved the same performance on test2 as union but with more than 6 times of phrase table size. Grow-diagonal (GD) has more than 1 bleu point on test2 than intersection but with less than half of phrase table size. The proposed new method OE is consistently better than both heuristic methods GD and H, with more than 1 point on dev/teset1 and 0.7 point on test2. Comparing the last group to the middle one, we can see the effect of the 'final' operation on all four methods. Tabel 1 shows that after applying the final operation, phrase table size is cut into half. When evaluated with automatic translation metric, all four methods generally perform much worse on dev and test1 that are close to training data, but better on NIST standard test2. We observe half BLEU point improvement for optimization method but marginal gain for heuristic-based approaches. This suggest that the phrase table accuracy get improved with the final operation. Optimization method directly tries to maximize the number of phrase pairs that can be extracted. We observe that it (OEF) is able to find more than 14% more phrase pairs than heuristic methods and achieve 1 BLEU point gain than the best heuristic method (GDF).

Method	dev	test1	test2
I	0.396	0.308	0.348
U	0.341	0.294	0.348
H	0.400	0.314	0.341
GD	0.391	0.314	0.360
OS	0.383	0.316	0.356
OE	0.410	0.329	0.367
HF	0.361	0.297	0.343
GDF	0.361	0.301	**0.362**
OSF	0.372	0.305	0.361
OEF	0.370	0.306	**0.372**

Table 2: Translation results (BLEU score) with phrase tables trained with different word alignment combination methods

4 Conclusions

We presented a simple yet effective method for word alignment symmetrization and combination in general. The problem is formulated as an optimization with greedy search driven by an effectiveness function, which can be customized directly to maximum benefit for intended applications such as phrase table training or synchronized grammar extraction in machine translation. Experimental results demonstrated consistent better BLEU scores than the best heuristic method. The optimization process can better maintain accuracy while improving coverage.

The algorithm is generic and leaves much space for variations. For instance, designing a better effectiveness function g, or considering a soft link with some probability rather than binary 0/1 connection would potentially be opportunities for further improvement. On the other hand, the search space of current algorithm is limited by the pool of candidate set, it is possible to suggest new links while driven by the target function.

Acknowledgments We thank the DARPA TransTac program for funding and the anonymous reviewers for their constructive suggestions.

References

N. F. Ayan. 2005. *Combining Linguistic and Machine Learning Techniques for Word Alignment Improvement*. Ph.D. thesis, University of Maryland, College Park, November.

P. Brown, S. Della Pietra, V. Della Pietra, and R. Mercer. 1993. The mathematics of machine translation: Parameter estimation. *Computational Linguistics*, 19:263–312.

S. F. Chen and J. Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proc. of ACL*, pages 310–318.

P. Koehn, F. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *Proc. of HLT-NAACL*, pages 48–54.

P. Liang, B. Taskar, and D. Klein. 2006. Alignment by agreement. In *Proc. of HLT-NAACL*, pages 104–111.

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

F. J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of ACL*, pages 160–167.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proc. of ACL*, pages 311–318.

R. Zens, E. Matusov, and H. Ney. 2004. Improved word alignment using a symmetric lexicon model. In *Proc. of COLING*, pages 36–42.