Learning Translation Consensus with Structured Label Propagation

†Shujie Liu*,
† Harbin Institute of Technology
Harbin, China
shujieliu@mtlab.hit.edu.cn

‡Chi-Ho Li, ‡Mu Li and ‡Ming Zhou
‡Microsoft Research Asia
Beijing, China
{chl, muli, mingzhou}@microsoft.com

Abstract

In this paper, we address the issue for learning better translation consensus in machine translation (MT) research, and explore the search of translation consensus from similar, rather than the same, source sentences or their spans. Unlike previous work on this topic, we formulate the problem as structured labeling over a much smaller graph, and we propose a novel structured label propagation for the task. We convert such graph-based translation consensus from similar source strings into useful features both for n-best output re-ranking and for decoding algorithm. Experimental results show that, our method can significantly improve machine translation performance on both IWSLT and NIST data, compared with a state-of-the-art baseline.

1 Introduction

Consensus in translation has gained more and more attention in recent years. The principle of consensus can be sketched as "a translation candidate is deemed more plausible if it is supported by other translation candidates." The actual formulation of the principle depends on whether the translation candidate is a complete sentence or just a span of it, whether the candidate is the same as or similar to the supporting candidates, and whether the supporting candidates come from the same or different MT system.

Translation consensus is employed in those minimum Bayes risk (MBR) approaches where the loss function of a translation is defined with respect to all other translation candidates. That is, the translation with the minimal Bayes risk is the one to the greatest extent similar to other candidates. These approaches include the work of Kumar and Byrne (2004), which re-ranks the n-best output of a MT decoder, and the work of Tromble et al. (2008) and Kumar et al. (2009), which does MBR decoding for lattices and hypergraphs.

Others extend consensus among translations from the same MT system to those from different MT systems. Collaborative decoding (Li et al., 2009) scores the translation of a source span by its n-gram similarity to the translations by other systems. Hypothesis mixture decoding (Duan et al., 2011) performs a second decoding process where the search space is enriched with new hypotheses composed out of existing hypotheses from multiple systems.

All these approaches are about utilizing consensus among translations for the same (span of) source sentence. It should be noted that consensus among translations of similar source sentences/spans is also helpful for good candidate selection. Consider the examples in Figure 1. For the source (Chinese) span "*五百 元 以下 的 茶*", the MT system produced the correct translation for the second sentence, but it failed to do so for the first one. If the translation of the first sentence could take into consideration the translation of the second sentence, which is similar to but not exactly the same as the first one, the final translation output may be improved.

Following this line of reasoning, a discriminative learning method is proposed to constrain the translation of an input sentence using

* This work has been done while the first author was visiting Microsoft Research Asia.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 302–310,
Jeju, Republic of Korea, 8-14 July 2012.

IWSLT Chinese to English Translation Task	
Src	*你 有没有 五百 元 以下 的 茶 ?*
Ref	*Do you have any tea under five hundred dollars ?*
Best1	*Do you have any **less than five hundred dollars tea** ?*
Src	*我 想要 五百 元 以下 的 茶 .*
Ref	*I would like some tea under five hundred dollars .*
Best1	*I would like **tea under five hundred dollars** .*

Figure 1. Two sentences from IWSLT (Chinese to English) data set. "**Src**" stands for the source sentence, and "**Ref**" means the reference sentence. "**Best1**" is the final output of the decoder.

the most similar translation examples from translation memory (TM) systems (Ma et al., 2011). A classifier is applied to re-rank the n-best output of a decoder, taking as features the information about the agreement with those similar translation examples. Alexandrescu and Kirchhoff (2009) proposed a graph-based semi-supervised model to re-rank n-best translation output. Note that these two attempts are about translation consensus for similar sentences, and about re-ranking of n-best output. It is still an open question whether translation consensus for similar sentences/spans can be applied to the decoding process. Moreover, the method in Alexandrescu and Kirchhoff (2009) is formulated as a typical and simple label propagation, which leads to very large graph, thus making learning and search inefficient. (c.f. Section 3.)

In this paper, we attempt to leverage translation consensus among similar (spans of) source sentences in bilingual training data, by a novel graph-based model of translation consensus. Unlike Alexandrescu and Kirchhoff (2009), we reformulate the task of seeking translation consensus among source sentences as structured labeling. We propose a novel label propagation algorithm for structured labeling, which is much more efficient than simple label propagation, and derive useful MT decoder features out of it. We conduct experiments with IWSLT and NIST data, and experimental results show that, our method can improve the translation performance significantly on both data sets, compared with a state-of-the-art baseline.

2 Graph-based Translation Consensus

Our MT system with graph-based translation consensus adopts the conventional log-linear model. For the source string f, the conditional probability of a translation candidate e is defined as:

$$p(e|f) = \frac{\exp\left(\sum_i (\lambda_i \psi_i(e,f))\right)}{\sum_{e' \in H(f)} \left(\exp\left(\sum_i (\lambda_i \psi_i(e',f))\right)\right)} \quad (1)$$

where ψ is the feature vector, λ is the feature weights, and $H(f)$ is the set of translation hypotheses in the search space.

Based on the commonly used features, two kinds of feature are added to equation (1), one is graph-based consensus features, which are about consensus among the translations of *similar* sentences/spans; the other is local consensus features, which are about consensus among the translations of the *same* sentence/span. We develop a structured label propagation method, which can calculate consensus statistics from translation candidates of similar source sentences/spans.

In the following, we explain why the standard, simple label propagation is not suitable for translation consensus, and then introduce how the problem is formulated as an instance of structured labeling, with the proposed structured label propagation algorithm, in section 3. Before elaborating how the graph model of consensus is constructed for both a decoder and N-best output re-ranking in section 5, we will describe how the consensus features and their feature weights can be trained in a semi-supervised way, in section 4.

3 Graph-based Structured Learning

In general, a graph-based model assigns labels to instances by considering the labels of similar instances. A graph is constructed so that each instance is represented by a node, and the weight of the edge between a pair of nodes represents the similarity between them. The gist of graph-based model is that, if two instances are connected by a strong edge, then their labels tend to be the same (Zhu, 2005).

In MT, the instances are source sentences or spans of source sentences, and the possible labels are their translation candidates. This scenario differs from the general case of graph-based model in two aspects. First, there are an indefinite, or even intractable, number of labels. Each of them is a string of words rather than a simple category. In the following we will call these labels as structured labels (Berlett et al., 2004). Second, labels are highly 'instance-dependent'. In most cases, for any two different (spans of) source sentences, however small their difference is, their correct labels (translations) are not exactly the same. Therefore, the principle of graph-based translation consensus must be reformulated as, if two instances (source spans) are similar, then their labels (translations) tend to be similar (rather than the same).

Note that Alexandrescu and Kirchhoff (2009) do not consider translation as structured labeling. In their graph, a node does not represent only a source sentence but a pair of source sentence and its candidate translation, and there are only two possible labels for each node, namely, 1 (this is a good translation pair) and 0 (this is not a good translation pair). Thus their graph-based model is a normal example of the general graph-based model. The biggest problem of such a perspective is inefficiency. An average MT decoder considers a vast amount of translation candidates for each source sentence, and therefore the corresponding graph also contains a vast amount of nodes, thus rendering learning over a large dataset is infeasible.

3.1 Label Propagation for General Graph-based Models

A general graph-based model is iteratively trained by label propagation, in which $p_{i,l}$, the probability of label l for the node i, is updated with respect to the corresponding probabilities for i's neighboring nodes $N(i)$. In Zhu (2005), the updating rule is expressed in a matrix calculation. For convenience, the updating rule is expressed for each label here:

$$p_{i,l}^{t+1} = \sum_{j \in N(i)} T(i,j) p_{j,l}^{t} \quad (2)$$

where $T(i,j)$, the propagating probability, is defined as:

$$T(i,j) = \frac{w_{i,j}}{\sum_{j' \in N(i)} w_{i,j'}} \quad (3)$$

$w_{i,j}$ defines the weight of the edge, which is a similarity measure between nodes i and j.

Note that the graph contains nodes for training instances, whose correct labels are known. The probability of the correct label to each training instance is reset to 1 at the end of each iteration. With a suitable measure of instance/node similarity, it is expected that an unlabeled instance/node will find the most suitable label from similar labeled nodes.

3.2 Structured Label Propagation for Graph-based Learning

In structured learning like MT, different instances would not have the same correct label, and so the updating rule (2) is no longer valid, as the value of $p_{i,l}$ should not be calculated based on $p_{j,l}$. Here we need a new updating rule so that $p_{i,l}$ can be updated with respect to $p_{j,l'}$, where in general $l \neq l'$.

Let us start with the model in Alexandrescu and Kirchhoff (2009). According to them, a node in the graph represents the pair of some source sentence/span f and its translation candidate e. The updating rule (for the label 1 or 0) is:

$$p_{(f,e)}^{t+1} = \sum_{(f',e') \in NP(f,e)} T\big((f,e),(f',e')\big) p_{(f',e')}^{t} \quad (4)$$

where $NP(f,e)$ is the set of neighbors of the node (f,e).

When the problem is reformulated as structured labeling, each node represents the source sentence/span only, and the translation candidates become labels. The propagating probability $T((f,e),(f',e'))$ has to be reformulated accordingly. A natural way is to decompose it into a component for nodes and a component for labels. Assuming that the two components are independent, then:

$$T\big((f,e),(f',e')\big) = T_s(f,f') T_l(e,e') \quad (5)$$

where $T_s(f,f')$ is the propagating probability from source sentence/span f' to f, and $T_l(e,e')$ is that from translation candidate e' to e.

The set of neighbors $NP(f,e)$ of a pair (f,e) has also to be reformulated in terms of the set of neighbors $N(f)$ of a source sentence/span f:

$$NP(f,e) = \{(f',e') | f' \in N(f), e' \in H(f')\} \quad (6)$$

where $H(f')$ is the set of translation candidates for source f'. The new updating rule will then be:

$$p_{f,e}^{t+1} = \sum_{f' \in N(f), e' \in H(f')} T_s(f,f') T_l(e,e') p_{f',e'}^t$$
$$= \sum_{f' \in N(f)} \sum_{e' \in H(f')} T_s(f,f') T_l(e,e') p_{f',e'}^t$$
$$= \sum_{f' \in N(f)} T_s(f,f') \sum_{e' \in H(f')} T_l(e,e') p_{f',e'}^t \quad (7)$$

The new rule updates the probability of a translation e of a source sentence/span f with probabilities of similar translations e's of some similar source sentences/spans f's.

Propagation probability $T_s(f,f')$ is as defined in equation (3), and $T_l(e,e')$ is defined given some similarity measure $sim(e,e')$ between labels e and e':

$$T_l(e,e') = \frac{sim(e,e')}{\sum_{e'' \in H(f')} sim(e,e'')} \quad (8)$$

Note that rule (2) is a special case of rule (7), when $sim(e,e')$ is defined as:

$$sim(e,e') = \begin{cases} 1 & if\ e = e'; \\ 0 & otherwise; \end{cases}$$

4 Features and Training

The last section sketched the structured label propagation algorithm. Before elaborating the details of how the actual graph is constructed, we would like to first introduce how the graph-based translation consensus can be used in an MT system.

4.1 Graph-based Consensus Features

The probability as estimated in equation (7) is taken as a group of new features in either a decoder or an n-best output re-ranker. We will call these features collectively as graph-based consensus features (GC):

$$GC(e,f) = \quad (9)$$
$$\log \left(\sum_{f' \in N(f)} T_s(f,f') \sum_{e' \in H(f')} T_l(e,e') p_{f',e'} \right)$$

Recall that, $N(f)$ refers to source sentences/spans which are similar with f, and $H(f')$ refers to translation candidates of f'. $p_{f',e'}$ is initialized with the translation posterior of e' given f'.The translation posterior is normalized in the n-best list. For the nodes representing the training sentence pairs, this posterior is fixed. $T_l(e,e')$ is the propagating probability in equation (8), with the similarity measure $sim(e,e')$ defined as the Dice co-efficient over the set of all n-grams in e and those in e'. That is,

$$sim(e,e') = Dice(NGr_n(e), NGr_n(e'))$$

where $NGr_n(x)$ is the set of *n*-grams in string x, and $Dice(A,B)$ is the Dice co-efficient over sets A and B:

$$Dice(A,B) = \frac{2|A \cap B|}{|A| + |B|}$$

We take $1 \le n \le 4$ for similarity between translation candidates, thus leading to four features. The other propagating probability $T_s(f,f')$, as defined in equation (3), takes symmetrical sentence level BLEU as similarity measure[1]:

$$w_{f,f'} = \frac{1}{2}(BLEU_{sent}(f,f') + BLEU_{sent}(f',f))$$

where $BLEU_{sent}(f,f')$ is defined as follows (Liang et al., 2006):

$$BLEU_{sent}(f,f') = \sum_{i=1}^{4} \frac{i - BLEU(f,f')}{2^{4-i+1}} \quad (10)$$

where $i - BLEU(f,f')$ is the IBM BLEU score computed over *i*-grams for hypothesis f using f' as reference.

In theory we could use other similarity measures such as edit distance, string kernel. Here simple n-gram similarity is used for the sake of efficiency.

4.2 Other Features

In addition to graph-based consensus features, we also propose local consensus features, defined over the n-best translation candidates as:

$$LC(e,f) = \log \left(\sum_{e' \in H(f)} p(e'|f) T_l(e,e') \right) \quad (11)$$

[1] BLEU is not symmetric, which means, different scores are obtained depending on which one is reference and which one is hypothesis.

where $p(e'|f)$ is translation posterior. Like GC, there are four features with respect to the value of n in n-gram similarity measure.

We also use other fundamental features, such as translation probabilities, lexical weights, distortion probability, word penalty, and language model probability.

4.3 Training Method

When graph-based consensus is applied to an MT system, the graph will have nodes for training data, development (dev) data, and test data (details in Section 5). There is only one label/translation for each training data node. For each dev/test data node, the possible labels are the n-best translation candidates from the decoder. Note that there is mutual dependence between the consensus graph and the decoder. On the one hand, the MT decoder depends on the graph for the GC features. On the other hand, the graph needs the decoder to provide the translation candidates as possible labels, and their posterior probabilities as initial values of various $p_{f,e}$. Therefore, we can alternatively update graph-based consensus features and feature weights in the log-linear model.

Algorithm 1 Semi-Supervised Learning

$GC^0 = 0$;
$\lambda^t = MERT(S^{dev}, T^{dev}, GC^0)$;
while not converged **do**
  $G^t = CreatG(S^{train}, T^{train}, S^{dev}, S^{test}, \lambda^t)$.
  $GC^{t+1} = StructLP(G^t)$.
  $\lambda^{t+1} = MERT(S^{dev}, T^{dev}, GC^{t+1})$
end while
return last (GC^t, λ^t)

Algorithm 1 outlines our semi-supervised method for such alternative training. The entire process starts with a decoder without consensus features. Then a graph is constructed out of all training, dev, and test data. The subsequent structured label propagation provides GC feature values to the MT decoder. The decoder then adds the new features and re-trains all the feature weights by Minimum Error Rate Training (MERT) (Och, 2003). The decoder with new feature weights then provides new n-best candidates and their posteriors for constructing another consensus graph, which in turn gives rise to next round of MERT. This alternation of structured label propagation and MERT stops when the BLEU score on dev data converges, or a pre-set limit (10 rounds) is reached.

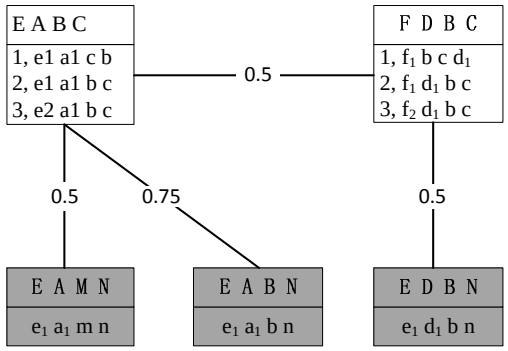

Figure 2. A toy graph constructed for re-ranking.

5 Graph Construction

A technical detail is still needed to complete the description of graph-based consensus, namely, how the actual consensus graph is constructed. We will divide the discussion into two sections regarding how the graph is used.

5.1 Graph Construction for Re-Ranking

When graph-based consensus is used for re-ranking the n-best outputs of a decoder, each node in the graph corresponds to a complete sentence. A separate node is created for each source sentence in training data, dev data, and test data. For any node from training data (henceforth training node), it is labeled with the correct translation, and $p_{f,e}$ is fixed as 1. If there are sentence pairs with the same source sentence but different translations, all the translations will be assigned as labels to that source sentence, and the corresponding probabilities are estimated by MLE. There is no edge between training nodes, since we suppose all the sentences of the training data are correct, and it is pointless to re-estimate the confidence of those sentence pairs.

Each node from dev/test data (henceforth test node) is unlabeled, but it will be given an n-best list of translation candidates as possible labels from a MT decoder. The decoder also provides translation posteriors as the initial confidences of

the labels. A test node can be connected to training nodes and other test nodes. If the source sentences of a test node and some other node are sufficiently similar, a similarity edge is created between them. In our experiment we measure similarity by symmetrical sentence level BLEU of source sentences, and 0.3 is taken as the threshold for edge creation.

Figure 2 shows a toy example graph. Each node is depicted as rectangle with the upper half showing the source sentence and the lower half showing the correct or possible labels. Training nodes are in grey while test nodes are in white. The edges between the nodes are weighted by the similarities between the corresponding source sentences.

5.2 Graph Construction for Decoding

Graph-based consensus can also be used in the decoding algorithm, by re-ranking the translation candidates of not only the entire source sentence but also every source span. Accordingly the graph does not contain only the nodes for source sentences but also the nodes for all source spans. It is needed to find the candidate labels for each source span.

It is not difficult to handle test nodes, since the purpose of MT decoder is to get all possible segmentations of a source sentence in dev/test data, search for the translation candidates of each source span, and calculate the probabilities of the candidates. Therefore, the cells in the search space of a decoder can be directly mapped as test nodes in the graph.

Training nodes can be handled similarly, by applying forced alignment. Forced alignment performs phrase segmentation and alignment of each sentence pair of the training data using the full translation system as in decoding (Wuebker et al., 2010). In simpler term, for each sentence pair in training data, a decoder is applied to the source side, and all the translation candidates that do not match any substring of the target side are deleted. The cells of in such a reduced search space of the decoder can be directly mapped as training nodes in the graph, just as in the case of test nodes. Note that, due to pruning in both decoding and translation model training, forced alignment may fail, i.e. the decoder may not be able to produce target side of a sentence pair. In such case we still map the cells in the search space as training nodes.

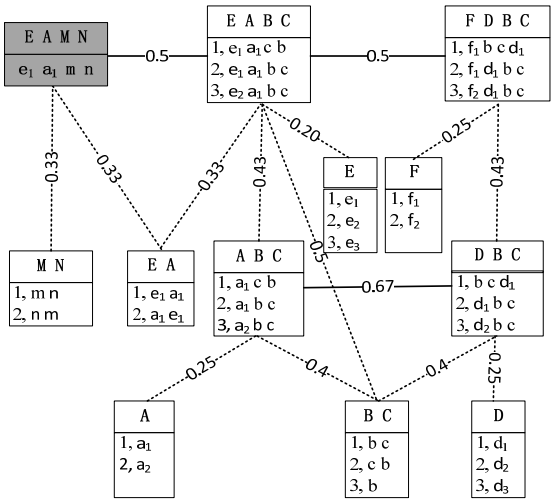

Figure 3. A toy example graph for decoding. Edges in dash line indicate relation between a span and its sub-span, whereas edges of solid line indicate source side similarity.

Note also that the shorter a source span is, the more likely it appears in more than one source sentence. All the translation candidates of the same source span in different source sentences are merged.

Edge creation is the same as that in graph construction for n-best re-ranking, except that two nodes are always connected if they are about a span and its sub-span. This exception ensures that shorter spans can always receive propagation from longer ones, and vice versa.

Figure 3 shows a toy example. There is one node for the training sentence "E A M N" and two nodes for the test sentences "E A B C" and "F D B C". All the other nodes represent spans. The node "M N" and "E A" are created according to the forced alignment result of the sentence "E A M N". As we see, the translation candidates for "M N" and "E A" are not the sub-strings from the target sentence of "E A M N". There are two kinds of edges. Dash lines are edges connecting nodes of a span and its sub-span, such as the one between "E A B C" and "E". Solid lines are edges connecting nodes with sufficient source side n-gram similarity, such as the one between "E A M N" and "E A B C".

6 Experiments and Results

In this section, graph-based translation consensus is tested on the Chinese to English translation tasks. The evaluation method is the case insensitive IBM BLEU-4 (Papineni et al., 2002). Significant testing is carried out using bootstrap re-sampling method proposed by Koehn (2004) with a 95% confidence level.

6.1 Experimental Data Setting and Baselines

We test our method with two data settings: one is IWSLT data set, the other is NIST data set. Our baseline decoder is an in-house implementation of Bracketing Transduction Grammar (Dekai Wu, 1997) (BTG) in CKY-style decoding with a lexical reordering model trained with maximum entropy (Xiong et al., 2006). The features we used are commonly used features as standard BTG decoder, such as translation probabilities, lexical weights, language model, word penalty and distortion probabilities.

Our IWSLT data is the IWSLT 2009 dialog task data set. The training data include the BTEC and SLDB training data. The training data contains 81k sentence pairs, 655k Chinese words and 806 English words. The language model is 5-gram language model trained with the target sentences in the training data. The test set is devset9, and the development set for MERT comprises both devset8 and the Chinese DIALOG set. The baseline results on IWSLT data are shown in Table 1.

	devset8+dialog	devset9
Baseline	48.79	44.73

Table 1. Baselines for IWSLT data

For the NIST data set, the bilingual training data we used is NIST 2008 training set excluding the Hong Kong Law and Hong Kong Hansard. The training data contains 354k sentence pairs, 8M Chinese words and 10M English words. The language model is 5-gram language model trained with the Giga-Word corpus plus the English sentences in the training data. The development data utilized to tune the feature weights of our decoder is NIST'03 evaluation set, and test sets are NIST'05 and NIST'08 evaluation sets. The baseline results on NIST data are shown in Table 2.

	NIST'03	NIST'05	NIST'08
Baseline	38.57	38.21	27.52

Table 2. Baselines for NIST data

6.2 Experimental Result

Table 3 shows the performance of our consensus-based re-ranking and decoding on the IWSLT data set. To perform consensus-based re-ranking, we first use the baseline decoder to get the n-best list for each sentence of development and test data, then we create graph using the n-best lists and training data as we described in section 5.1, and perform semi-supervised training as mentioned in section 4.3. As we can see from Table 3, our consensus-based re-ranking (G-Re-Rank) outperforms the baseline significantly, not only for the development data, but also for the test data.

Instead of using graph-based consensus confidence as features in the log-linear model, we perform structured label propagation (Struct-LP) to re-rank the n-best list directly, and the similarity measures for source sentences and translation candidates are symmetrical sentence level BLEU (equation (10)). Using Struct-LP, the performance is significantly improved, compared with the baseline, but not as well as G-Re-Rank.

	devset8+dialog	devset9
Baseline	48.79	44.73
Struct-LP	**49.86**	**45.54**
G-Re-Rank	**50.66**	**46.52**
G-Re-Rank-GC	**50.23**	**45.96**
G-Re-Rank-LC	**49.87**	**45.84**
G-Decode	**51.20**	**47.31**
G-Decode-GC	**50.46**	**46.21**
G-Decode-LC	**50.11**	**46.17**

Table 3. Consensus-based re-ranking and decoding for IWSLT data set. The results in bold type are significantly better than the baseline.

We use the baseline system to perform forced alignment procedure on the training data, and create span nodes using the derivation tree of the forced alignment. We also saved the spans of the sentences from development and test data, which will be used to create the responding nodes for consensus-based decoding. In such a way, we create the graph for decoding, and perform semi-

supervised training to calculate graph-based consensus features, and tune the weights for all the features we used. In Table 3, we can see that our consensus-based decoding (G-Decode) is much better than baseline, and also better than consensus-based re-ranking method. That is reasonable since the neighbor/local similarity features not only re-rank the final n-best output, but also the spans during decoding.

To test the contribution of each kind of features, we first remove all the local consensus features and perform consensus-based re-ranking and decoding (G-Re-Rank-GC and G-Decode-GC), and then we remove all the graph-based consensus features to test the contribution of local consensus features (G-Re-Rank-LC and G-Decode-LC). Without the graph-based consensus features, our consensus-based re-ranking and decoding is simplified into a consensus re-ranking and consensus decoding system, which only re-rank the candidates according to the consensus information of other candidates in the same n-best list.

From Table 3, we can see, the G-Re-Rank-LC and G-Decode-LC improve the performance of development data and test data, but not as much as G-Re-Rank and G-Decode do. G-Re-Rank-GC and G-Decode-GC improve the performance of machine translation according to the baseline. G-Re-Rank-GC does not achieve the same performance as G-Re-Rank-LC does. Compared with G-Decode-LC, the performance with G-Decode-GC is much better.

	NIST'03	NIST'05	NIST'08
Baseline	38.57	38.21	27.52
Struct-LP	38.79	38.52	28.06
G-Re-Rank	**39.21**	**38.93**	**28.18**
G-Re-Rank-GC	38.92	38.76	**28.21**
G-Re-Rank-LC	38.90	38.65	27.88
G-Decode	**39.62**	**39.17**	**28.76**
G-Decode-GC	**39.42**	**39.02**	**28.51**
G-Decode-LC	39.17	38.70	**28.20**

Table 4. Consensus-based re-ranking and decoding for NIST data set. The results in bold type are significantly better than the baseline.

We also conduct experiments on NIST data, and results are shown in Table 4. The consensus-based re-ranking methods are performed in the same way as for IWSLT data, but for consensus-based decoding, the data set contains too many sentence pairs to be held in one graph for our machine. We apply the method of Alexandrescu and Kirchhoff (2009) to construct separate graphs for each development and test sentence without losing global connectivity information. We perform modified label propagation with the separate graphs to get the graph-based consensus for n-best list of each sentence, and the graph-based consensus will be recorded for the MERT to tune the weights.

From Table 4, we can see that, Struct-LP improves the performance slightly, but not significantly. Local consensus features (G-Re-Rank-LC and G-Decode-LC) improve the performance slightly. The combination of graph-based and local consensus features can improve the translation performance significantly on SMT re-ranking. With graph-based consensus features, G-Decode-GC achieves significant performance gain, and combined with local consensus features, G-Decode performance is improved farther.

7 Conclusion and Future Work

In this paper, we extend the consensus method by collecting consensus statistics, not only from translation candidates of the same source sentence/span, but also from those of similar ones. To calculate consensus statistics, we develop a novel structured label propagation method for structured learning problems, such as machine translation. Note that, the structured label propagation can be applied to other structured learning tasks, such as POS tagging and syntactic parsing. The consensus statistics are integrated into the conventional log-linear model as features. The features and weights are tuned with an iterative semi-supervised method. We conduct experiments on IWSLT and NIST data, and our method can improve the performance significantly.

In this paper, we only tried Dice co-efficient of n-grams and symmetrical sentence level BLEU as similarity measures. In the future, we will explore other consensus features and other similarity measures, which may take document level information, or syntactic and semantic information into consideration. We also plan to introduce feature to model the similarity of the source

sentences, which are reflected by only one score in our paper, and optimize the parameters with CRF model.

References

Andrei Alexandrescu, Katrin Kirchhoff. 2009. Graph-based learning for statistical machine translation. In *Proceedings of Human Language Technologies and Annual Conference of the North American Chapter of the ACL*, pages 119-127.

Peter L. Bertlett, Michael Collins, Ben Taskar and David McAllester. 2004. Exponentiated gradient algorithms for large-margin structured classification. In *Proceedings of Advances in Neural Information Processing Systems*.

John DeNero, David Chiang, and Kevin Knight. 2009. Fast consensus decoding over translation forests. In *Proceedings of the Association for Computational Linguistics*, pages 567-575.

John DeNero, Shankar Kumar, Ciprian Chelba and Franz Och. 2010. Model combination for machine translation. In *Proceedings of the North American Association for Computational Linguistics*, pages 975-983.

Nan Duan, Mu Li, Dongdong Zhang, and Ming Zhou. 2010. Mixture model-based minimum bayes risk decoding using multiple machine translation Systems. In *Proceedings of the International Conference on Computational Linguistics*, pages 313-321.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing*, pages 388-395.

Shankar Kumar and William Byrne. 2004. Minimum bayes-risk decoding for statistical machine translation. In *Proceedings of the North American Association for Computational Linguistics*, pages 169-176.

Shankar Kumar, Wolfgang Macherey, Chris Dyer, and Franz Och. 2009. Efficient minimum error rate training and minimum bayes-risk decoding for translation hypergraphs and lattices. In *Proceedings of the Association for Computational Linguistics*, pages 163-171.

Mu Li, Nan Duan, Dongdong Zhang, Chi-Ho Li, and Ming Zhou. 2009. Collaborative decoding: partial hypothesis re-ranking using translation consensus between decoders. In *Proceedings of the Association for Computational Linguistics*, pages 585-592.

Percy Liang, Alexandre Bouchard-Cote, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *Proceedings of the International Conference on Computational Linguistics and the ACL*, pages 761-768

Yanjun Ma, Yifan He, Andy Way, Josef van Genabith. 2011. Consistent translation using discriminative learning: a translation memory-inspired approach. In *Proceedings of the Association for Computational Linguistics*, pages 1239-1248.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the Association for Computational Linguistics*, pages 160-167.

Kishore Papineni, Salim Roukos, Todd Ward and Wei-jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the Association for Computational Linguistics*, pages 311-318.

Roy Tromble, Shankar Kumar, Franz Och, and Wolfgang Macherey. 2008. Lattice minimum bayes-risk decoding for statistical machine translation. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing*, pages 620-629.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3).

Joern Wuebker, Arne Mauser and Hermann Ney. 2010. Training phrase translation models with leaving-one-out. In *Proceedings of the Association for Computational Linguistics*, pages 475-484.

Deyi Xiong, Qun Liu and Shouxun Lin. 2006. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of the Association for Computational Linguistics*, pages 521-528.

Xiaojin Zhu. 2005. Semi-supervised learning with graphs. *Ph.D. thesis*, Carnegie Mellon University. CMU-LTI-05-192.