A Decoding Method of System Combination Based on Hypergraph in SMT

Yupeng Liu, Sheng Li, and Tiejun Zhao

Machine Transaltion Lab, Harbin Institute of Technology
6F Aoxiao Building, NO.27 Jiaohua Street, Nangang District
Harbin, 150001, China
ypliu@mtlab.hit.edu.cn, lisheng@hit.edu.cn,
tjzhao@mtlab.hit.edu.cn

Abstract. The word level system combination, which is better than phrase level and sentence level, has emerged as a powerful post-processing method for statistical machine translation (SMT). This paper first give the definition of HyperGraph(HG) as a kind of compact data structure in SMT, and then introduce simple bracket transduction grammar(SBTG) for hypergraph decoding. To optimize the more feature weights, we introduce minimum risk (MR) with deterministic annealing (DA) into the training criterion, and compare two classic training procedures in experiment. The deoding approaches of n-gram model based on hypergraph are shown to be superior to conventional cube pruning in the setting of the Chinese-to-English track of the 2008 NIST Open MT evaluation.

Keywords: System combination, HyperGraph, N-gram model, Inside-outside Pruning.

1 Introduction

System combination[2][9][22] aims to find consensus translations among different statistical machine translation (SMT) systems. It has been proven that such consensus translations are usually better than the translation output of individual systems. Confusion network(CN) for word-level combination is a widely adopted approach for combining SMT output, which was shown to outperform sentence re-ranking methods and phrase-level combination[2]. In order to construct confusion network, word alignment between a skeleton (or backbone) and a hypothesis is a key technic in this approach. The important alignment methods include Translation Edit Rate (TER)[19] based alignment, which is proposed in Sim et al. (2007) and often taken as the baseline, and a couple of other approaches, such as the Indirect Hidden Markov Model (IHMM)[23] and the ITG-based alignment[6], which are recently proposed with better results reported. Joint optimization[24] integrates CN construction and CN-based decoding into a decoder without skeleton selection. Lattice-based system combination[10] model normalizes the alignment between the skeleton and the hypothesis CN into the lattice without breaking the phrase structure.

H. Deng et al. (Eds.): AICI 2011, Part III, LNAI 7004, pp. 115–125, 2011.

In this paper, we start from the view of decoding, and introduce a hypergraph decoding algorithm of system combination which runs in a bottom-up manner. This algorithm parses the target words in the order of CN with reordered words and inserted *null* words. The hypergraph decoding based on system combination is different from the conventional decoding algorithm, in particular:

- The formal grammar is used in hypergraph-based system combination. Hypergraph is generally used by parsing and machine translation [17-18]. The algorithm produces the hypergraph by simple bracket transduction grammar (SBTG) with lexical and non-terminal rules.
- Two pass decoding algorithm is adopted in the framework. The first pass uses a 5-gram language model, and the resulting parse hypergraph is used in the second pass to guide search with the re-estimated n-gram probability. We can rescore the derivation through original viterbi model and re-estimated n-gram model, using the product of inside and outside probability.
- MR with/without DA, which attempts the solution of increasingly difficult optimization problem, is introduced in hypergraph decoding, because of fitting the curve of the probability distribute of log-linear model better and solving the large feature number.

This paper is structured as follows. We will first, in Section 2, give the definition of hypergraph; then in Section 3, we show decoding procedure including inside outside pruning, n-gram probablity re-estimation and decoding algorithm. In Section 4, experiment results and analysis are presented.

2 Hypergraph Definition

Formally, a hypergraph[17-18] in system combination is defined as a 4-tuple H=<V, E, G, R> , where V is a finite set of hypernode, E is a finite set of hyperedge, G∈R is the unique goal item in H, and R is a set of weights. For a input sentence of target language $e_1^J = e_1, \ldots e_J$, each hypernode is in the form of X_i^j, which denotes the partial translation of target partial language $e_i, \ldots e_j$ spanning the substring from i-1 to j. Each hyperedge e∈E is a triple tuple e=<T(e), h(e), w(e)>, where T(e)∈V is a vector of tail nodes, h(e)∈V is its head, and w(e) is a weight function from R|T(e)| to R.

Our hypergraph-based system combination is represented by simplified bracket transduction grammar (SBTG). Formally, the set of these hyperedges can be defined as a 3-tuple E=<T, N, P>, where T is a set of the terminal word symbol in target language, N is a set of the non-terminal symbol including three symbols N={S, X1, X2}, P is a set of production rules including two types:

- Lexical rule: X→w, w∈D
- Non-terminal rule : S→X1 X2,
  X→X1 X2

where D is a dictionary including null word (ε) for normalization in system combination, start symbol (G∈R) and single word. Non-terminal rule is like straight reordering in bracket transduction grammar [8].

2.1 Inside and Outside Prunning

The inside-outside algorithm is a way of re-estimating hyperedge probability in synchronous context-free grammar (SCFG). It was introduced as a generalization of the forward-backward algorithm for parameter estimation on Hidden Markov Models (HMM). It is used to compute expections, for example as part of the EM (expectation maximization) algorithm. Standard inside and outside recursion formulation is showed in Equation (1) and Equation (2), in which *I(v)* and *O(v)* is inside score and outside score in hypernode, *f(e)* and *w(e)* are feature and weight vector ,respectively, and *dim(f(e))* is the dimension of feature vector. *IN(v)* and *OUT(v)* are incoming and outgoing hyperedge of *v*.

$$I(v)=\sum_{e\in IN(v)}\left[\sum_{\dim(f(e))} f(e)w(e)\right].\left[\prod_{u\in T(e)} I(u)\right] \tag{1}$$

$$O(v)=\sum_{e\in OUT(v)}\left[\left(\sum_{\dim(f(e))} f(e)w(e)\right)\times\left(O(h(e))\prod_{\substack{u\in T(e)\\ u\neq v}} I(u)\right)\right] \tag{2}$$

Trough Equation (3), we can get the posterior probability of hyperedge including language model feature score, which is computed after its hypernode is generated. After coputing the posterior probability, we can prune these hyperedges whose probability is below the threshold. *P(e|F)* is the posterior probability of specific hyperedge *e*, and p_e is the original weight of hyperedge *e* in hypergraph *F*, and *Z* is normalization factor that equals to the inside probability of the root node in *F*.

$$p(e\mid F)=\frac{1}{Z}p_e.O(h(e)).\prod_{v\in T(e)} I(v) \tag{3}$$

The inside-outside algorithm for pruning and n-gram model is shown in Algorithm 1 and Algorithm 2.

Algorithm 1 Inside Recursion Algorithm
run from **bottom** to **top**
1: **for** $v\in F$
2: $I(v)=1$
3: **for** $e\in IN(v)$

4: $I_e(v) = p_e \cdot I_e(v)$

5: **for** $u \in T(e)$

6: $I_e(v) = I(u) \cdot I_e(v)$

7: $I(v) = I(v) + I_e(v)$

8: **return** $I(v)$

Algorithm 2 Outside Recursion Algorithm
run from **top** to **bottom**

1: **for** $v \in F$

2. $O(v) = 1$

3: **for** $e \in OUT(v)$

4: $O_e(v) = p_e \cdot O(h(e))$

5: **for** $u \in T(e)$

6: **if** $u \neq v$

7: $O_e(v) = I(u) \cdot O_e(v)$

8: $O(v) = O(v) + O_e(v)$

9: **return** $O(v)$

2.2 N-Gram Probability

We adopt three types of n-gram estimation model and compare these models which are proposed by Li[26], which describes an algorithm for computing it through n-gram model, by Kumar[21],which describes an efficient approximate algorithm through the highest edge posterior probability relative to predecessors on all derivations in hypergraph, and by Dereno[12][13], which give the expectation count and exact algorithm. The n-gram estimation algorithm framework is shown in Algortihm 3.

Algorithm 3 N-gram Model/Posterior/Expectation Count Computation
run from **bottom** to **top**

1: run **inside** and **outside** algorithm

2: compute **hyperedge posterior probability** *P(e|F)*

3: for $v \in F$

3: **for** $e \in IN(v)$

4: **if** $w_n \in e$

5: $ec_n(w)$ += $p_w(e \mid F)$

6: $ec_n(h(w))$ += $p_w(e \mid F)$

7: $p_n = \frac{ec_n(w)}{ec_n(h(w))}$

8: $q_n = \frac{ec_n(w)}{Z}$

9: **return** p_n as **n-gram model**

10: **return** $ec_n(w)$ as **n-gram expectation count**

11: **return** q_n as **n-gram posterior probability**

2.3 Decoding Algorithm

Different from conventional cube pruning[7], we use cube growing[16] to exploit the idea of lazy computation, which get n-best from top to bottom in hypergraph. Conceptually, complicate hypergraph decoding incorporates the following procedure:

1. Generating the hypergraph: generate all hypernodes and hyperedges inhypergraph F bottom-up in topological order. Every hypernode have many hyperedges with SBTG rule for phrase structure. According to the generating order of a target sentence, the decoding category is bottom-up[25]. Finally, the hypergraph has a distinguished goal item for convenient decoding.
2. Running inside recursion algorithm: for every hypernode, we compute the inside score from its hyperedges with tail nodes. The algorithm set the inside score of axiom item to zero, and run from bottom to top for the score through axiom item and inference rule[7].
3. Running outside recursion algorithm: for every hypernode, we compute the outside score of the hypernode, which might be the left or right branch of the parent hypernode(two non-terminal in SCFG).
4. Computing hyperedge posterior probability: according to inside and outside probability of hypernode and inside score of goal item and hyperedge weight, we compute the hyperedge posterior probability in hypergraph.
5. Inside-Outside pruning: to reduce the search space and improve the speed, pruning the hyperedges is important in the light of posterior probability. Our pruning strategy is including threshold and histogram. Pruning algorithm shows in above section.
6. Computing n-gram Probability: the method assume n-gram locality of the hypergraph, the property that any n-gram introduced by a hyperedge appears in all derivations that include the hyperedge and thus we apply the rule of edge *e* to n-grams on *T(e)* and propagate n-1 gram prefixes or suffixes to *h(e)*.
7. Assignning scores to hyperedge: if the hyperedge introduce the ngram into the derivation, the n-gram probability is assigned to hyperedge for search k-best translation from hypergraph.
8. Reranking hyperedges in hypernode: we reestimate the n-gram model feature value by n-gram posterior probability and count expectation.
9. Finding the best path in the hypergraph: cube growing[16] compute the k-th best item in every cell lazily. But this algorithm still calculates a full k-th best item for

every hypernode in the hypergraph. We can therefore take laziness to an extreme by delaying the whole k-best calculation until after generating the hypergraph. The algorithm need two phases, which are forward that is same as viterbi decoding, but stores the hypergraph (keep many hyperedges in each hypernode) and backward phase that recursively ask what's your k-th best derivations from top to down.

The word posterior feature $f_s(arc)$ is the same as the one proposed by [2]. Other features used in our log-linear model include language model $f_{lm}=LM(e_i)$, real word count $f_{wc}=N_{word}(e_i)$ and ε word count $f_\varepsilon=N_{null}(e_i)$, where *CN* is confusion network. Equation (4) is decoding framework.

$$\log p(e \mid f) = \sum_{i=1}^{N_{arc}} \Big[\log(\sum_{j=1}^{N_s} \lambda_j f_j(arc_i)) + \alpha LM(e_i) + \beta N_{null}(e_i) + \gamma N_{word}(e_i) \Big] \tag{4}$$

where f denote the source language, e denote the consensus translation generate by system combination. λ_i, α, β and γ is weight of other feature. Cube pruning algorithm with beam search is employed to search for consensus translation [16].

3 Experiments

In our chinese-english translation experiments, the candidate systems participating in the system combination are as listed in Table 1: Sys-1 uses a syntax-based decoder[1], informed by a source language dependency parse (Chinese); Sys-2 is a single-pass phrase-based system. The decoder uses a beam search to produce translation candidates left-to-right, incorporating future distortion penalty estimation and early pruning to limit the search[20]; Sys-3 is essentially the same as Sys-2 except that we apply a syntactic reordering system as a preprocessor to reorder Chinese sentences in training and test data in such a way that the reordered Chinese sentences are much closer to English in terms of word order. For a Chinese sentence, we first parse it using the Stanford Chinese Syntactic Parser [15], and then reorder it by applying a set of reordering rules, proposed by [3], to the parse tree of the sentence; Sys-4 is a syntax-based pre-ordering based MT system using a syntax-based pre-ordering model as described in [4]; Sys-5 is a hierarchical phrase-based system as described by [7]. It uses a statistical phrase-based translation model that uses hierarchical phrases; Sys-6 uses a lexicalized re-ordering model similar to the one described by [8]. It uses a maximum entropy model to predicate reordering of neighbor blocks (phrase pairs); Sys-7 is a two-pass phrase-based system with adapted LM proposed by Foster and Kuhn [11]. This system uses a standard two-pass phrase-based approach; Sys-8 is a hierarchical phrase-based system that uses a 4-gram language model in the first pass to generate n-best lists, which are rescored by three additional language models to generate the final translations via re-ranking.

Table 1. Performance of individual systems on the dev and test set

System ID	Dev(BLEU)	Test(BLEU)
Sys1	31.48	25.63
Sys2	31.41	26.78
Sys3	32.31	26.03
Sys4	30.55	27.38
Sys5	32.60	27.75
Sys6	28.99	22.86
Sys7	27.33	21.45
Sys8	28.91	22.67

3.1 Experimental Setup

The development set of experimental setup is NIST MT06 data set including 1099, and the test set of it is NIST MT08 data set including 1357 from both newswire and web-data genres. Both dev and test sets have four references per sentence. However, to save computation effort, the result on the dev and test set are reported in case insensitive BLEU score instead. The above system generates the 10-best of every sentence as input of system combination through the max-BLEU traning(MERT). The language model used for all models is a 5-gram model trained with Xinhua portion of LDC English Gigaword corpus version 3. We use incremental IHMM as baseline[5]. The parameter of incremental IHMM show in [5]. The lexical translation probabilities used in semantic similarity model are from a small portion (FBIS+GALE) of the constrained track training data. The skeleton is select by MBR. The loss function used for incremental IHMM style is BLEU. As to incremental system, the default order of hypothesis is ascending according to BLEU score against the skeleton. We employ the distortion model in incremental IHMM[5]. When confusion network is built in training process, a set of word confident are added for decoding. Meanwhile, the decoder uses the features of word confident, language model, word penalty and null penalty.

We compare the system combination based on hypergraph decoding with conventional decoding style. The hypergraph decoder was implemented by modifying the classic CKY algorithm with cube growing[16].

3.2 Decoding Performance

During second-pass decoding, we use the same beam size as first pass decoding because the outside probability estimation of the second-pass decoding is discriminative enough to guide second-pass hypergraph decoding. We develop a unified algorithm of three n-gram probabilty which are n-gram model (denoted by ngram_1), n-gram count expection (denoted by ngram_2) and n-gram posterior probability (denoted by ngram_3), and then compare the performance of them.

The Effect of *n*-gram Model: as shown in Table 2, decoding with *1-5*-gram+*wp* (word penalty denoted by *wp*) model of different estimation methods improve (+0.94, +0.63 and +0.57 BLEU score) over baseline on the development set, and we achieve

an absolute improvement (+1.16, +1.17 and +1.13 BLEU score) on the test set. The experimental result proves that *n*-gram feature is effective.

The effect of Viterbi model can be seen through comparing Table 3 with Table 2. The various interpolation models show an improvement of +0.25, +0.3 and +0.27 BLEU points over model without Viterbi on the development set, and +0.25, +0.13 and +0.04 BLEU point on test set. If we compare it with baseline (incremental IHMM), the best performance of three types of *n*-gram probability can be obtained when the setting is *Vi+1-5*gram_1+*wp*. It obtains +1.19 and +1.41 BLEU score on the development and test set respectively.

The experimental results prove the efficiency of *n*-gram and Viterbi+*n*-gram model.

Table 2. The quality of second-pass decoding on the development and test set

n-gram model	NIST06	NIST08
Baseline	39.34	32.82
*1-5*gram_1+*wp*	40.28	33.98
*1-5*gram_2+*wp*	39.97	33.99
*1-5*gram_3+*wp*	39.91	33.95

Table 3. The quality of second-pass decoding with Viterbi baseline on the development and test set

Viterbi+*n*-gram	NIST06	NIST08
*Vi+1-5*gram_1+*wp*	40.53	34.23
*Vi+1-5*gram_2+*wp*	40.27	34.12
*Vi+1-5*gram_3+*wp*	40.18	33.99

The Effect of MR with DA: MR with DA is introduces by [27], but it apply in SMT and don't compare performance of these scheme. We compare the five training schema: MERT vs. MR with different setting, which are with/without DA, with/without quenching scaling factor λ and on hypergraph. With the entropy constrains, starting temperature *T=1000*; quenching temperature *T=0.001*. The temperature is cooled by half at each step; then we double λ at each step. Once *T* is quite cool, it is common in practice to switch to rising λ directly and rapidly until some convergence condition. We optimize feature weight vector $\boldsymbol{\theta}$ and hyperparameter λ through BFGS optimization.

The configures of the experiment use the interpolation between *1-5*gram_1 and Viterbi model. We compare five settings on the development in Figure 1 and the test set in Table 4. MERT, MR without DA&quenching, MR&DA without quenching, MR&DA with quenching and MR&DA with quenching on HG achieve a BLEU score of 40.53, 40.17, 40.37, 40.50 and 40.50 on the development set. The best performance

can be obtained by MERT on the development set, and meanwhile the worst performance can be obtained by it on test set. The fact proves the overfitting of MERT. The reason of decreased performance of 2-th iteration MR with DA is the initialization bias.

Table 4. The MERT and MR with/without DA performance of the test set

Training Criterion	NIST08
MERT	34.23
MR without DA&queching	34.24
MR&DA without quenching	34.28
MR&DA with quenching	34.29
MR&DA with quenching on HG	34.29

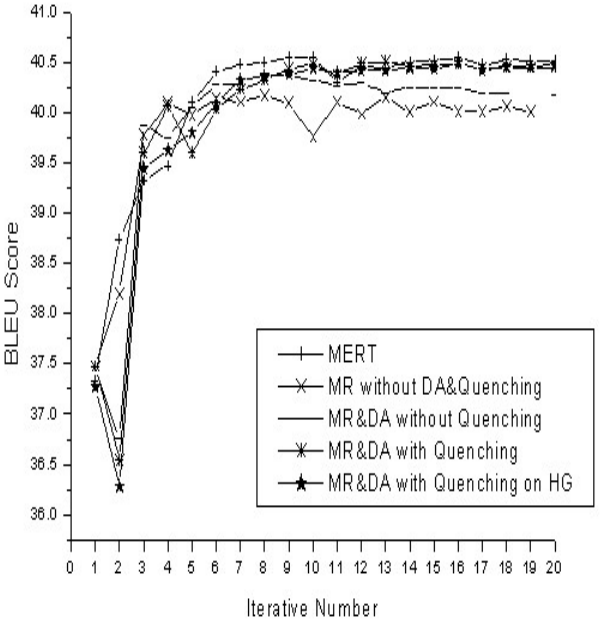

Fig. 1. The MERT and MR with/without DA performance on the development set

Compared to MERT, MR&DA on HG has almost the same performance on test set because of a small number of features[27] or a sparse feature of the non-terminal rule which only includes language model probability. In total, MR&DA on HG outperform baseline (incremental IHMM) using Cube Pruning up to +1.47 in BLEU score.

4 Conclusion

This paper proposed hypergraph_based decoding method in word–level system combination, and then compare a set of n-gram feature on hypergraph for two-pass decoding. The hypergraph decoding includes three types of n-gram probabilities, which are n-gram calculating style. The method is evaluated against state-of-the-art baselines including classic incremental IHMM on the NIST MT06 and MT08 C2E tasks. We also use HG-based MR with/without DA training, which solve the overfitting to objective function and a large number of features. The two-pass hypergraph decoding is shown to outperform cube pruning style decoding significantly. We ontain 1.28 and 1.59 BLEU score improvement in dev and test set through two-pass decoding using HG-based MR training.

Since our training algorithm can cope with a large number of features, in future work, we plan to incorporate more expressive features and hypergraph training in the model. We combine the different system combination model, which has the different expressive ability.

References

1. Menezes, A., Quirk, C.: Using Dependency Order Templates to Improve Generality in Translation. In: Proc. 2nd WMT at ACL, Prague, Czech Republic (2007)
2. Rosti, A.-V.I., Matsoukas, S., Schwartz, R.: Improved Word-level System Combination for Machine Translation. In: Proceedings of ACL (2007)
3. Wang, C., Collins, M., Koehn, P.: Chinese Syntactic Reordering for Statistical Machine Translation. In: EMNLP 2007 (2007)
4. Li, C.-H., Zhang, D., Li, M., Zhou, M., Li, C.-H., Li, M., Guan, Y.: A Probabilistic Approach to Syntax-based Reordering for Statistical Machine Translation. In: Proceedings of ACL (2007)
5. Li, C.-H., He, X., Liu, Y., Xi, N.: Incremental HMM Alignment for MT System Combination. In: Proceedings of ACL (2009)
6. Karakos, D., Eisner, J., Khudanpur, S., Dreyer, M.: Machine Translation System Combination using ITG-based Alignments. In: Proceedings of ACL (2008)
7. Chiang, D.: Hierarchical Phrase-based Translation. Computational Linguistics 33(2) (2007)
8. Xiong, D., Liu, Q., Lin, S.: Maximum Entropy based Phrase Reordering Model for Statistical Machine Translation. In: Proceedings of COLING/ACL 2006, Sydney, Australia, pp. 521–528 (July 2006)
9. Matusov, E., Ueffing, N., Ney, H.: Computing Consensus Translation from Multiple Machine Translation Systems using Enhanced Hypothesis Alignment. In: Proceedings of EACL (2006)
10. Feng, Y., Liu, Y., Mi, H., Liu, Q., Lu, Y.: Lattice-based System Combination for Statistical Machine Translation. In: Proceedings of ACL (2009)
11. Foster, G., Kuhn, R.: Mixture-Model Adaptation for SMT. In: Proc. of the Second ACL Workshop on Statistical Machine Translation, pp. 128–136 (2007)
12. Denero, J., Chiang, D., Knight, K.: Fast Consensus Decoding over Translation Forest. In: Proceedings of ACL (2009)

13. DeNero, J., Kumar, S., Chelba, C., Och, F.: Model Combination for Machine Translation. In: Proceedings of NAACL (2010)
14. Sim, K.C., Byrne, W.J., Gales, M.J.F., Sahbi, H., Woodland, P.C.: Consensus Network Decoding for Statistical Machine Translation System Combination. In: Proc. of ICASSP, pp. 105–108 (2007)
15. Levy, R., Manning, C.: Is It Harder To Parse Chinese, or The Chinese Treebank? Published in Proceedings of ACL 2003 (2003)
16. Huang, L., Chiang, D.: Better k-best Parsing. In: Proceedings of the International Workshop on Parsing Technologies (IWPT), pp. 53–64 (2005)
17. Huang, L., Chiang, D.: Forest Rescoring: Faster Decoding with Integrated Language Models. In: Proceedings of ACL, Prague, Czech Rep. (2007)
18. Huang, L.: Forest reranking: Discriminative parsing with Non-local Features. In: Proc. of (2008)
19. Snover, M., Dorr, B., Schwartz, R., Micciulla, L., Makhoul, J.: A Study of Translation Edit Rate with Targeted Human Annotation. In: Proceedings of AMTA (2006)
20. Moore, R., Quirk, C.: Faster Beam-Search Decoding for Phrasal Statistical Machine Translation. In: Proc. of MT Summit XI (2007)
21. Kumar, S., Macherey, W., Dyer, C., Och, F.: Efficient Minimum Error Rate Training and Minimum Bayes-Risk Decoding for Translation Hypergraphs and Lattices. In: Proceedings of ACL, pp. 163–171 (2009)
22. Bangalore, S., Bordel, G., Riccardi, G.: Computing Consensus Translation from Multiple Machine Translation Systems. In: Workshop on Automatic Speech Recognition and Understanding, Madonna di Campiglio, Italy, pp. 351–354 (2001)
23. He, X., Yang, M., Gao, J., Nguyen, P., Moore, R.: Indirect-HMM based Hypothesis Alignment for Combining Outputs from Machine Translation Systems. In: Proc. of EMNLP (2008)
24. He, X., Toutanova, K.: Joint Optimization for Machine Translation System Combination. In: Proc. of EMNLP (2009)
25. Liu, Y., Mi, H., Feng, Y., Liu, Q.: Joint Decoding with Multiple Translation Models. In: Proc. of ACL, pp. 576–584 (2009)
26. Li, Z., Eisner, J., Khudanpur, S.: Variational Decoding for Statistical Machine Translation. In: Proceedings of ACL (2009a)
27. Li, Z., Eisner, J.: First- and Second-order Expectation Semirings with Applications to Minimum-Risk Training on Translation Forests. In: Proceedings of EMNLP (2009b)