Forest-to-String Translation using Binarized Dependency Forest for IWSLT 2012 OLYMPICS Task

Hwidong Na and Jong-Hyeok Lee

Department of Computer Science and Engineering
Pohang University of Science and Technology (POSTECH), Republic of Korea
{ leona, jhlee } @postech.ac.kr

Abstract

We participated in the OLYMPICS task in IWSLT 2012 and submitted two formal runs using a forest-to-string translation system. Our primary run achieved better translation quality than our contrastive run, but worse than a phrase-based and a hierarchical system using Moses.

1. Introduction

Syntax-based SMT approaches incorporate tree structures of sentences to the translation rules in the source language [10, 14, 23, 22], the target language [1, 7, 12, 18, 26], or both [2, 3, 28]. Due to the structural constraint, the transducer grammar extracted from parallel corpora tends to be quite large and flat. Hence, the extracted grammar consists of translation rules that appear few times, and it is difficult to apply most translation rules in the decoding stage.

For generalization of transducer grammar, binarization methods of a phrase structure grammar have been suggested [1, 12, 20, 26]. Binarization is a process that transforms an n-ary grammar into a binary grammar. During the transformation, a binarization method introduces the virtual nodes which is not included in the original tree. The virtual nodes in a binarized phrase structure grammar are annotated using the phrasal categories in the original tree. Unfortunately, these approaches are available only for string-to-tree models, because we are not aware of the correct binarization of the source tree at the decoding stage. To take the advantage of binarization in tree-to-string models, a binarized forest of phrase structure trees has been proposed [25]. Since the number of all possible binarized trees are exponentially many, the author encode the binarized trees in a packed forest, which was originally proposed to encode the multiple parse trees [14].

In contrast to previous studies, we propose to use a novel binarized forest of dependency trees for syntax-based SMT. A dependency tree represents the grammatical relations between words as shown in Figure 1. Dependency grammar has that holds the best phrasal cohesion across the languages [6]. We utilize dependency labels for the annotation of the virtual nodes in a binarized dependency tree. To the best of our knowledge, this is the first attempt to binarize the dependency grammar.

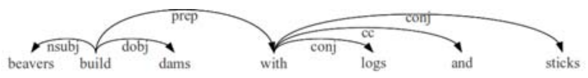

Figure 1: An example dependency tree with dependency labels

2. Binarized Dependency Forest

Forest-to-string translation approaches construct a packed forest for a source sentence, and find the mapping between the source forest and the target sentence. A packed forest is a compact representation of exponentially many trees. Most studies focused on the forest of multiple parse trees in order to reduce the side effect of the parsing error [13, 14, 15, 19, 27, 28]. On the other hand, Zhang et. al. [25] attempted to binarize the best phrase structure tree. A binarization method comprises the conversion of the possibly non-binary tree into a binarized tree. The authors suggested a binarized forest, which is a packed forest that compactly encodes multiple binarized trees. It improves generalization by breaking downs the rules into the smallest possible parts. Thus, a binarized forest that the authors suggested covers non-constituent phrases by introducing a virtual node, for example, "beavers build" or "dams with" in Figure 1.

In this paper, we propose a binarized forest analogous to but two differences. First, we binarize the best *dependency* tree instead of the best phrase structure tree. Because dependency grammar does not have non-terminal symbols, it is not trivial to construct a binarized forest from a dependency tree. Second, we annotate the virtual nodes using the dependency labels instead of the phrase categories.

2.1. Construction of binarized dependency forest

We utilize the concept of the well-formed dependency proposed by Shen et. al. [18]. A well-formed dependency refers to either a connected sub-graph in a dependency tree (treelet) or a floating dependency, i.e., a sequence of treelets that have a common head word. For example, "beavers build" is a treelet and "dams with" is a floating dependency.

Since the number of all possible binarized trees are exponentially many, we encode a binarized forest $\mathcal{F}$ in a chart analogous to Zhange et. al. [25]. Let π be the best dependency tree of a source sentence from w_1 to w_n. π consists of a set of information for each word w_j, i.e. the head word $HEAD(w_j)$ and the dependency label $LABEL(w_j)$. For each word w_j, we initialize the chart with a binary node v. For each span $s^{begin:end}$ that ranges from $w_{begin+1}$ to w_{end}, we check whether the span consists of a well-formed dependency. For each pair of sub-spans $s^{begin:mid}$ and $s^{mid:end}$, which are rooted at v_l and v_r respectively, we add an incoming binary edge e if:

- Sibling (SBL): v_l and v_r consist of a floating dependency, or
- Left dominates right (LDR): v_l has no right child $RIGHT(v_l)$ and v_l dominates v_r, or
- Right dominates left (RDL): v_r has no left child $LEFT(v_r)$ and v_r dominates v_l.

Note that the root node of the SBL case is a virtual node, and we extend the incoming binary edge of v for LDR and RDL cases by attaching v_r and v_l, respectively. For example, $\{dobj, prep\}^{2:4}$ is the root node for the SBL case where v_l is "dams" and v_r is "with", and build$^{0:4}$ is the root node for the LDR case where v_l is build$^{0:2}$ and v_r is $\{dobj, prep\}^{2:4}$.

Algorithm 1 shows the pseudo code, and Figure 2 shows a part of the binarized forest for the example dependency tree in Figure 1. Although the worst time complexity of the construction is $O(n^3)$, the running time is negligible when we extract translation rules and decode the source sentence in practice (less than 1 ms). Because we restrict the combination, a binary node has a constant number of incoming binary edges. Thus, the space complexity is $O(n^2)$.

2.2. Augmentation of phrasal node

We also augment phrasal nodes for word sequences, i.e. phrases in PBSMT. A phrasal node p is a virtual node corresponding to a span $s^{begin:end}$, yet it does not consist of a well-formed dependency. Hence, augmenting phrasal nodes in $\mathcal{F}$ leads to including all word sequences covered in PBSMT. Because phrases capture more specific translation patterns, which are not linguistically justified, we expect that the coverage of the translation rules will increase as we augment phrasal nodes.

We augment phrasal nodes into the chart that we built for the binarized forest. For each span $s^{begin:end}$, we introduce a phrasal node if the chart cell is not defined, i.e. the span does not consist of well-formed dependency. We restrict the maximum length of a span covered by a phrasal node to L. For each pair of sub-spans $s^{begin:mid}$ and $s^{mid:end}$, where they are rooted at v_l and v_r respectively we add an incoming binary edge e if any of v, v_l, or v_r is a phrasal node. Algorithm 2 shows the pseudo code.

Algorithm 1: Construct Binarized Dependency Forest

```
1  function Construct($\pi$)
   input  : A dependency tree $\pi$ for the sentence
            $w_1 \dots w_J$
   output: A binarized forest $\mathcal{F}$ stored in $chart$
2  for $col = 1 \dots J$ do
3      create a binary node $v$ for $w_{col}$
4      $chart[1, col] \leftarrow v$
5  end
6  for $row = 2 \dots J$ do
7      for $col = row \dots J$ do
8          if a span $s^{col-row:col}$ consists of a
           well-formed dependency then
9              create a binary node $v$
10             for $i = 1 \dots row$ do
11                 $v_l \leftarrow chart[i, col - row + i]$
12                 $v_r \leftarrow chart[row - i - 1, col]$
13                 if $v_l$ and $v_r$ consist of a
                   floating dependency then
14                     create an incoming binary node
                       $e = \langle v, v_l, v_r \rangle$
15                 end
16                 else if $v_l$ has no right child
                   and $v_l$ dominates $v_r$ then
17                     create an incoming binary node
                       $e = \langle v_l, LEFT(v_l), v_r \rangle$
18                 end
19                 else if $v_r$ has no left child
                   and $v_r$ dominates $v_l$ then
20                     create an incoming binary node
                       $e = \langle v_r, v_l, RIGHT(v_r) \rangle$
21                 end
22                 else
23                     continue // combination is
                           not allowed
24                 end
25                 $IN(v) \leftarrow IN(v) \cup \{e\}$
26             end
27             $chart[row, col] \leftarrow v$
28         end
29     end
30 end
```

2.3. Annotation of virtual node using dependency label

The translation probability of fine-grained translation rules is more accurate than that of a coarse one [21]. It is also beneficial in terms of efficiency because fine-grained translation rules reduce the search space by constraining the applicable rules. Therefore, we annotate the virtual nodes in $\mathcal{F}$ using dependency labels that represent the dependency relation between the head and the dependent word.

An annotation of a virtual node v for a span $s^{begin:end}$ is a set of dependency labels $ANN(v) =$

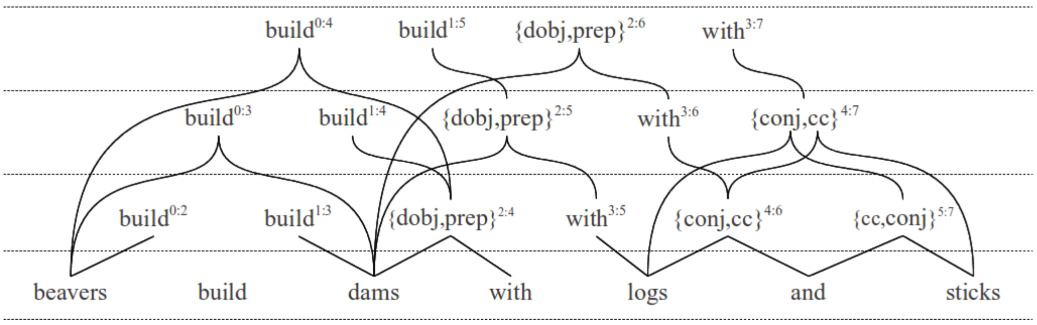

Figure 2: A part of the chart of the binarized dependency forest for the example dependency tree in Figure 1. The dotted lines represent the rows in a chart, and the nodes in a row represent the cells the rooted at these nodes. The solid lines are the incoming binary edges of the binary nodes. For each root node v which covers more than two words, we denote the covered span to $v^{begin:end}$ for clarity. The virtual nodes have annotation using dependency labels as explained in Section 2.3. Note that a binary node can have more than one incoming binary edges, e.g. $\{conj, cc\}^{4:7}$.

Algorithm 2: Augment Phrasal Nodes

```
1  function Augment($\mathcal{F}$, $L$, $n$)
   input  : A binarized forest $\mathcal{F}$, the maximum phrase
            length $L$, the sentence length $n$
   output: A binarized forest $\mathcal{F}'$ with phrasal nodes
2  for $row = 2 \ldots min(L, n)$ do
3      for $col = row \ldots n$ do
4          if $row \leq L$ and $chart[row, col]$ is not
           defined then
5              create a phrasal node $v$
6              $chart[row, col] \leftarrow v$
7          end
8          else
9              $v \leftarrow chart[row, col]$
10         end
11         for $i = 0 \ldots row$ do
12             $v_l \leftarrow chart[i, col - row + i]$
13             $v_r \leftarrow chart[row - i - 1, col]$
14             if any of $v$, $v_l$, or $v_r$ is a
               phrasal node then
15                 create an incoming binary node
                   $e = \langle v, v_l, v_r \rangle$
16                 $IN(v) \leftarrow IN(v) \cup \{e\}$
17             end
18         end
19     end
20 end
```

$\bigcup_{j=begin+1}^{end} LABEL(w_j)$. Note that we merge duplicated relations if there are more than two modifiers. Thus it abstracts the dependency relations of the covered words, for example, the modifiers consist of a coordination structure such as "logs and sticks" in the example. When there exist more than two preposition phrases, our proposed method also takes advantage of the abstraction. Since a coordination structure or a the number of preposition phrases can be long arbitrarily, merging duplicated relations minimizes the variation of the annotations, and increases the degree of f the generalization.

2.4. Extraction of translation rule

We extract tree-to-string translation rules from the binarized forest as proposed in [13] after we identify the substitution sites, i.e., frontier nodes. A binary node is a frontier node if a word in the corresponding source span has a consistent word alignment, i.e. there exists at least one alignment to the target and any word in the target span does not aligned to the source word out of the source span. For example, since build$^{0:2}$ has inconsistent word alignment in Figure 3, it is not a frontier node. The identification of the frontier nodes in $\mathcal{F}$ is done by a single post-order traversal.

After we identify the frontier nodes, we extract the minimal rules from each frontier node [8]. Figure 4 shows the minimal rules extracted from the example sentence. For each frontier node v, we expand the tree fragment until it reaches the other frontier nodes. For each tree fragment, we compile the corresponding target words, and substitute the frontier nodes with the labels. If a virtual node is the root of a tree fragment, we do not substitute the frontier nodes that cover length-1 spans. For example, R2, R5, R6, R8 and R9 have length-1 spans that is not substituted. The extraction of the minimal rules takes linear time to the number of the nodes in $\mathcal{F}$, thus the length of the sentence.

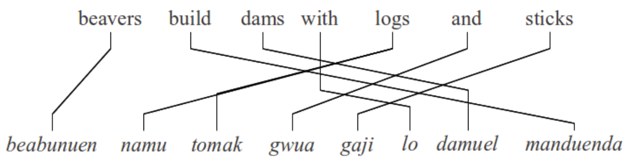

Figure 3: An example of word alignment and target sentence.

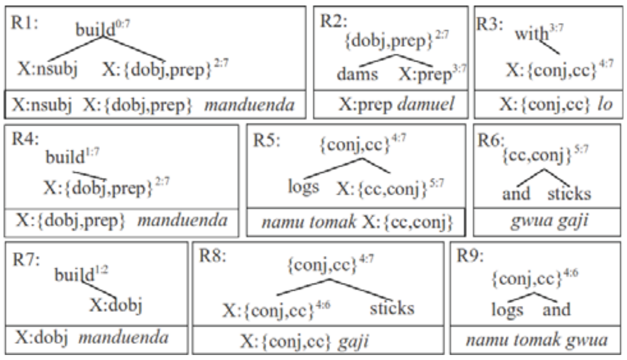

Figure 4: The minimal translation rules. Each box represents the source tree fragment (above) and the corresponding target string (below) with mapping for substitution sites (X).

We also extract composed rules in order to increase the coverage of the extracted translation rules [7]. We believe that the composed rules also prevents the over-generalization of the binarized dependency forest. For each tree fragment in the minimal rules, we extend the the tree fragment beyond the frontier nodes until the size of the tree fragment is larger than a threshold. When we restrict the size, we do not count the non-leaf virtual nodes. We also restrict the number of the extension for each tree fragment in practice. Figure 5 shows two composed rules that extend the tree fragments in R1 and R8, respectively.

3. Experiments

We performed the experiments in the OLYMPICS task in IWSLT 2012. The task provided two parallel corpora, one from the HIT Olypic Trilingual Corpus (HIT) and the other from the Basic Tranvel Expression Corpus (BTEC). We only carried out our experiment with the official condition, i.e. training data limited to supplied data only. As the size of training data sets in the HIT and BTEC is relatively small, we regards the 8 development data sets in the BTEC corpus also as training corpora. Each development corpus in the BTEC corpus has multiple references and we duplicated the source sentences in Chinese for the reference sentences in English. One development set (Dev) was used for tuning the weights in the log-linear model and the other development set (DevTest) was used for testing the translation quality. Finally, the formal runs were submitted by translating the evaluation corpus. Table 1 summarizes the statistics of corpora we used.

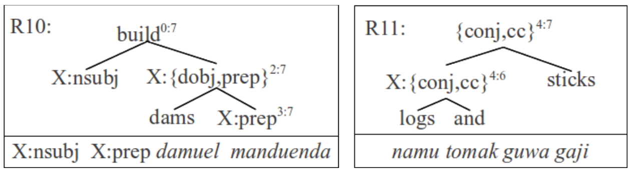

Figure 5: Two composed translation rules.

Table 1: Corpus statistics of the corpora. Sentence column shows the number of sentence pairs, and Source and Target column shows the number of words in Chinese and English, respectively.

	Sentence	Source	Target
Train	111,064	911,925	1,007,611
Dev	1,050	9,499	10,125
DevTest	1,007	9,623	10,083
Test	998	9,902	11,444

Table 2: The official evaluation results of the submitted runs. P is the primary run and C is the contrastive run. M is a phrase-based SMT using Moses with lexicalized reordering and H is Hierarchical phrase-based SMT using Moses-chart.

	BLEU	NIST	TER	GTM	METEOR
P	0.1203	3.7176	0.7999	0.4352	0.3515
C	0.1031	3.4032	0.8627	0.4207	0.3163
M	0.1666	4.3703	0.6892	0.4754	0.4168
H	0.1710	4.4841	0.6817	0.4803	0.4182

We compared the effectiveness of our proposed methods in two different settings. The primary run fully utilized the methods described in Section 2. The contrastive run, on the other hand, skipped the augmentation of phrasal nodes described in Section 2.2. Therefore, the translation rules used in the contrastive run only included tree fragments that satisfies the well-formed dependency. We denoted the contrastive run as the baseline in the next section. We also compared the submitted runs with a phrase-base SMT with lexicalized reordering and a hierarchical phrase-based SMT using Moses. Table 2 shows the evaluation results using various metrics following the instruction provided by the task organizer (README.OLYMPICS.txt). Please refer the details in the overview paper [5].

For both primary and contrastive runs, we implemented a forest-to-string translation system using cube pruning [11] in Java. The implementation of our decoder is based on a log-linear model. The feature functions are similar to hierarchical PBSMT including a penalty for a glue rule, as well as bidirectional translation probabilities, lexical probabilities, and word and rule counts. For the translation probabilities, we applied Good-Turing discounting smoothing in order to prevent over-estimation of sparse rules. We also restricted the maximum size of a tree fragment to 7, and the number of the extension to 10,000.

For an Chinese sentence, we used a CRFTagger to obtain POS tags, and a chart parser to obtain a dependency tree developed in our laboratory. The F-measure of the CRFTagger is 95% and the unlabelled arc score (UAS) of the parser is 87%. We used GIZA++[17] to obtain bidirectional word alignments for each segmented parallel corpus, and applied the grow-diag-final-and heuristics. For tuning the parameter of a log-linear model, we utilized an implementation of minimum error rate training [16], Z-MERT [24]. We built the n-gram language model using the IRSTLM toolkit 5.70.03 [4], and converted in binary format using KenLM toolkit [9].

4. Discussion

The augmentation of the phrasal nodes (primary run) outperformed the baseline (contrastive run) in all evaluation metrics. However, both our approaches underperformed any of Moses systems. We suspected the reasons as follows:

- Over-generalization of the dependency structure causes a lot of incorrect reordering, although we annotate the virtual nodes using dependency labels.
- Over-constraint of the tree structure makes a lot of translations impossible that are possible with phrase-based models.
- Parsing error affects the extraction of translation rules and decoding, which are inevitable.

Besides, there are many out-of-vocabulary in all systems due to the relatively small size of the training data. We hope more data in the HIT and BTEC corpora will be available in the future.

5. Conclusion

We participated in the OLYMPICS task in IWSLT 2012 and submitted two formal runs using a forest-to-string translation system. Our primary run achieved better translation quality than our contrastive run, but worse than a phrase-based and a hierarchical system using Moses.

6. Acknowledgements

This work was supported in part by the Korea Ministry of Knowledge Economy (MKE) under Grant No.10041807 and under the "IT Consilience Creative Program" support program supervised by the NIPA(National IT Industry Promotion Agency)" (C1515-1121-0003), in part by the Korea Science and Engineering Foundation (KOSEF) grant funded by the Korean government (MEST No. 2012-0004981), in part by the BK 21 Project in 2012.

7. References

[1] DeNero, J., Bansal, M., Pauls, A., and Klein, D. (2009). Efficient parsing for transducer grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 227–235, Boulder, Colorado. Association for Computational Linguistics.

[2] Ding, Y. and Palmer, M. (2005). Machine translation using probabilistic synchronous dependency insertion grammars. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 541–548, Ann Arbor, Michigan. Association for Computational Linguistics.

[3] Eisner, J. (2003). Learning non-isomorphic tree mappings for machine translation. In *The Companion Volume to the Proceedings of 41st Annual Meeting of the Association for Computational Linguistics*, pages 205–208, Sapporo, Japan. Association for Computational Linguistics.

[4] Federico, M., Bertoldi, N., and Cettolo, M. (2008). Irstlm: an open source toolkit for handling large scale language models. In *Proceedings of Interspeech*, Brisbane, Australia.

[5] Federico, M., Cettolo, M., Bentivogli, L., Paul, M., and Stüker, S. (2012). Overview of the IWSLT 2012 Evaluation Campaign. Proc. of the International Workshop on Spoken Language Translation.

[6] Fox, H. J. (2002). Phrasal cohesion and statistical machine translation. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - Volume 10*, EMNLP '02, pages 304–3111, Stroudsburg, PA, USA. Association for Computational Linguistics.

[7] Galley, M., Graehl, J., Knight, K., Marcu, D., DeNeefe, S., Wang, W., and Thayer, I. (2006). Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 961–968, Sydney, Australia. Association for Computational Linguistics.

[8] Galley, M., Hopkins, M., Knight, K., and Marcu, D. (2004). What's in a translation rule? In Susan Dumais, D. M. and Roukos, S., editors, *HLT-NAACL 2004: Main Proceedings*, pages 273–280, Boston, Massachusetts, USA. Association for Computational Linguistics.

[9] Heafield, K. (2011). Kenlm: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland. Association for Computational Linguistics.

[10] Huang, L. (2006). Statistical syntax-directed translation with extended domain of locality. In *In Proc. AMTA 2006*, pages 66–73.

[11] Huang, L. and Chiang, D. (2005). Better k-best parsing. In *Proceedings of the Ninth International Workshop on Parsing Technology*, Parsing '05, pages 53–64, Stroudsburg, PA, USA. Association for Computational Linguistics.

[12] Huang, L., Zhang, H., Gildea, D., and Knight, K. (2009). Binarization of synchronous context-free grammars. *Comput. Linguist.*, 35(4):559–595.

[13] Mi, H. and Huang, L. (2008). Forest-based translation rule extraction. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 206–214, Honolulu, Hawaii. Association for Computational Linguistics.

[14] Mi, H., Huang, L., and Liu, Q. (2008). Forest-based translation. In *Proceedings of ACL-08: HLT*, pages 192–199, Columbus, Ohio. Association for Computational Linguistics.

[15] Mi, H. and Liu, Q. (2010). Constituency to dependency translation with forests. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1433–1442, Uppsala, Sweden. Association for Computational Linguistics.

[16] Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 160–167, Stroudsburg, PA, USA. Association for Computational Linguistics.

[17] Och, F. J. and Ney, H. (2000). Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, pages 440–447, Stroudsburg, PA, USA. Association for Computational Linguistics.

[18] Shen, L., Xu, J., and Weischedel, R. (2008). A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of ACL-08: HLT*, pages 577–585, Columbus, Ohio. Association for Computational Linguistics.

[19] Tu, Z., Liu, Y., Hwang, Y.-S., Liu, Q., and Lin, S. (2010). Dependency forest for statistical machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1092–1100, Beijing, China. Coling 2010 Organizing Committee.

[20] Wang, W., May, J., Knight, K., and Marcu, D. (2010). Re-structuring, re-labeling, and re-aligning for syntax-based machine translation. *Comput. Linguist.*, 36(2):247–277.

[21] Wu, X., Matsuzaki, T., and Tsujii, J. (2010). Fine-grained tree-to-string translation rule extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 325–334, Uppsala, Sweden. Association for Computational Linguistics.

[22] Xie, J., Mi, H., and Liu, Q. (2011). A novel dependency-to-string model for statistical machine translation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 216–226, Edinburgh, Scotland, UK. Association for Computational Linguistics.

[23] Xiong, D., Liu, Q., and Lin, S. (2007). A dependency treelet string correspondence model for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 40–47, Stroudsburg, PA, USA. Association for Computational Linguistics.

[24] Zaidan, O. F. (2009). Z-mert: A fully configurable open source tool for minimum error rate training of machine translation systems. *Prague Bulletin of Mathematical Linguistics*, 91:79–88.

[25] Zhang, H., Fang, L., Xu, P., and Wu, X. (2011). Binarized forest to string translation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 835–845, Portland, Oregon, USA. Association for Computational Linguistics.

[26] Zhang, H., Huang, L., Gildea, D., and Knight, K. (2006). Synchronous binarization for machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 256–263, New York City, USA. Association for Computational Linguistics.

[27] Zhang, H., Zhang, M., Li, H., Aw, A., and Tan, C. L. (2009). Forest-based tree sequence to string translation model. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 172–180, Suntec, Singapore. Association for Computational Linguistics.

[28] Zhang, H., Zhang, M., Li, H., and Chng, E. S. (2010). Non-isomorphic forest pair translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 440–450, Cambridge, MA. Association for Computational Linguistics.