

Microsoft Research Treelet Translation System: NIST MT Evaluation 06

*Xiaodong He, Arul Menezes, Chris Quirk, Anthony Aue,
Simon Corston-Oliver, Jianfeng Gao and Patrick Nguyen*

Microsoft Research
One Microsoft Way
Redmond, WA 98052

{xiaohe, arulm, chrisq, anthaue, simonco, jfgao, panguyen}@microsoft.com

Abstract

The Microsoft Research Treelet Translation System is a syntax-based SMT system employing a source language dependency parser. The system translates by composing dependency treelet translation pairs, ranked by a log-linear combination of models including a global reordering model informed by a source language dependency parser. The source language analysis is produced by a Chinese Treebank parser. The log-linear combination also includes standard channel models, language models, and common feature functions. These weights are optimized by maximizing BLEU on a development set.

1. Introduction

The dependency treelet translation system developed at MSR is a statistical MT system that takes advantage of linguistic tools, namely a source language dependency parser, as well as a word alignment component.

To train a translation system, we require a sentence-aligned parallel corpus. First both sides are word-segmented and the source side is parsed to obtain dependency trees. Next the corpus is word-aligned, and the source dependencies are projected onto the target sentences using the word alignments. From the aligned dependency corpus we extract all treelet translation pairs, and train an order model and a bi-lexical dependency model.

To translate, we parse the input sentence, and employ a decoder to find a combination and ordering of treelet translation pairs that cover the source tree and are optimal according to a set of models. In a now-common generalization of the classic noisy-channel framework, we use a log-linear combination of models (Och and Ney 2002), as in below:

$$\text{translation}(\mathbf{S}, \mathbf{F}, \Lambda) = \arg \max_{\mathbf{T}} \left\{ \sum_{f \in \mathbf{F}} \lambda_f f(\mathbf{S}, \mathbf{T}) \right\}$$

Such an approach toward translation scoring has proven very effective in practice, as it allows a translation system to incorporate information from a variety of probabilistic or non-probabilistic sources. The weights $\Lambda = \{ \lambda_f \}$ are selected to maximize the BLEU score (Papineni et al. 2002) of held-out development data using the method proposed by Och (2003).

2. Data

For training, we used exclusively those resources listed in the large training data track. No additional LDC datasets, web data, or other resources were used.

2.1. Parallel data

We used all available Chinese-English data. For most datasets we used the provided sentence separation; with the FBIS data, however, we used a slight variation on the standard Church and Gale approach. In addition to length features, we used lexical correspondence information: the score for a each bead in a sentence alignment was determined by the lexical overlap between the source sentence(s) and target sentence(s) in the bead using a modified cosine similarity was used. Normally bag of words cosine similarity is defined in terms of dot product. Since the lexical items here are drawn from different languages, we instead defined the similarity between a Chinese bag of words \mathbf{c} and an English bag of words \mathbf{e} as $\mathbf{cM}(\mathbf{e}^T)$, where \mathbf{M} is a 0-1 matrix defined as $M_{i,j} = 1$ if the Chinese word i and the English word j are present in the Chinese-English dictionary LDC2002L27; 0 otherwise.

After sentence extraction, some cleanup was performed on these datasets: we removed duplicate sentence pairs and removed certain noisy sentences (as determined by regular expressions). Table 1 notes the size of each dataset in terms of sentences. This full corpus was then word aligned.

For development testing and tuning, we primarily focused on MTC-1 (LDC2002T01), MTC-2 (LDC2003T17), and the 2005 Tides test set (LDC2006E38). To speed development and bias toward the most relevant data in such a large dataset, we used a greedy data selection method to pick a useful subset of the full training data. We began by indexing each Chinese substring in all of the available test sets. Next we performed a greedy selection of the data from the full corpus to produce the subset S in the following manner. We initialize S to be empty. Then, given each sentence pair P in turn, we look at each substring of P in common with the test data, and find its frequency in S . If any such substring has a frequency below a threshold T , then P is added to S . The resulting dataset is therefore representative of the overall data albeit much smaller. Table 1 also contains details about this selected subset.

We also commissioned a word alignment of 250 sentences from the Chinese Treebank translations (LDC2003E07) by a bilingual Chinese-English speaker. To ease the task for this annotator, we selected sentences of thirty words or less. This small dataset contained 8,033 Chinese words and 10,075

Dataset	Sentences	Original data		Selected subset		
		Chinese Characters	English Words	Sentences	Chinese Characters	English Words
LDC2002E18	86,471	3,927,759	2,582,478			
LDC2003E07	4,172	155,998	118,868			
LDC2003E14	152,002	6,282,206	5,140,765			
LDC2004T08	1,940,823	51,052,863	40,038,865			
LDC2004E12	4,684,611	195,566,043	133,259,359			
LDC2005T06	10,317	430,905	286,407			
LDC2005T10	156,811	4,799,765	3,271,875			
Total	7,035,207	262,215,539	184,698,617			

Table 1: Characteristics of both the full dataset and the representative subset selected on the basis of the test data.

English words. The annotator produced an alignment containing 7,148 sure and 2,843 possible links.

2.2. Monolingual target language data

For training target language models, we used both the target side of the training data as well as the English Gigaword corpus (LDC2005T12). Both the target side of the training data and the Gigaword corpus were normalized to match MT evaluation, notably counting punctuation as separate words. All data were converted to lower case.

In the Gigaword corpus, articles from February and March 2002 were blacked out to allow development on 2002 data. The resulting corpus contained a total of 2.7 billion tokens. We selected the 120k most frequent words as the vocabulary, achieving a 0.77% out-of-vocabulary rate on MTC-02.

2.3. Other translation data

We did attempt to incorporate the LDC dictionary (LDC2002L27) and named entity lists (LDC2005T34) to aid in translation of unfound words, though the lists appeared to be somewhat noisy and of limited use. Transliteration for Chinese names was performed by consulting a table providing the most likely Pinyin Romanization of each Chinese character.

2.4. Linguistic resources

The linguistic resources in the competition, including the word segmentation component, the part-of-speech tagger, and the dependency parser, were all trained on the Penn Chinese Treebank V5 (LDC2005T01).

3. System Details

A brief word on notation: s and t represent source and target lexical nodes; \mathbf{S} and \mathbf{T} represent source and target trees; \mathbf{s} and \mathbf{t} represent source and target treelets (connected subgraphs of the dependency tree). Where the intent is clear, we will disregard the structure of these elements and consider these structures to be sets of lexical items: the expression $\forall t \in \mathbf{T}$ refers to all the lexical items in the target language tree \mathbf{T} . Similarly, $|\mathbf{T}|$ refers to the count of lexical items in \mathbf{T} . We use subscripts to indicate selected words: \mathbf{T}_n represents the n^{th} lexical item in an in-order traversal of \mathbf{T} .

3.1. Word segmentation

To wordbreak the training corpus, test data, and evaluation data, we employed the semi-CRF word segmentation approach (Andrew 2006). The segmenter was trained on the word segmented data in the Chinese Treebank, achieving F-measure of 95.64%. We experimented with some variants of feature sets, but found that word segmentation accuracy did not correlate particularly well with translation accuracy. On the other hand, incorporating a small set of simple rules (ensure punctuation and numbers are separate tokens, make strings of Roman characters be a single token) improved BLEU scores significantly, despite a degradation in the segmentation F-measure to 94.80%.

3.2. Part-of-speech tagging

Words were annotated with parts-of-speech using a Cyclic Dependency Network tagger (Toutanova et al. 2003). We used a rather small feature set, which achieved a per token accuracy of 94.1% on the Chinese Treebank development test set when trained on the Chinese Treebank training set.

3.3. Dependency parsing

Dependency analysis is an alternative to constituency analysis (Tesnière 1959, Mel'čuk 1988). In a dependency analysis of syntax, words directly modify other words, with no intervening non-lexical nodes. We use the terms child node and parent node to denote the tokens in a dependency relation. Each child has a single parent, with the lexical root of the sentence dependent on a synthetic ROOT node.

We use the parsing approach described in (Corston-Oliver et al. 2006). The parser is trained on dependencies extracted from the Chinese Penn Treebank version 5.0 (Xue et al. 2005) by using the head-percolation rules of Collins (1999). For training the parser we used oracle part-of-speech tags and oracle word-breaking.

Given a sentence x , the goal of the parser is to find the highest-scoring parse \hat{y} among all possible parses $y \in Y$. The score of a given parse y is the sum of the scores of all its dependency links $(i, j) \in y$:

$$s(x, y) = \sum_{(i, j) \in y} d(i, j) = \sum_{(i, j) \in y} \mathbf{w} \cdot \mathbf{f}(i, j)$$

where the link (i, j) indicates a parent-child dependency between the token at position i and the token at position j . The score $d(i, j)$ of each dependency link (i, j) is further decomposed as the weighted sum of its features $\mathbf{f}(i, j)$.

The feature vector $\mathbf{f}(i, j)$ computed for each possible parent-child dependency includes the part-of-speech (POS), lexeme and stem of the parent and child tokens, the POS of tokens adjacent to the child and parent, and the POS of each token that intervenes between the parent and child. Various combinations of these features are used, for example a new feature is created that combines the POS of the parent, lexeme of the parent, POS of the child and lexeme of the child. Each feature is also conjoined with the direction and distance of the parent, e.g. does the child precede or follow the parent, and how many tokens intervene?

To set the weight vector \mathbf{w} , we train twenty averaged perceptrons (Collins 2002) on different shuffles of data drawn from sections 02--21 of the Penn Treebank. The averaged perceptrons are then combined to form a Bayes Point Machine (Herbrich et al. 2001, Harrington et al 2003), resulting in a linear classifier that is competitive with wide margin techniques. To find the optimal parse given the weight vector \mathbf{w} and feature vector $\mathbf{f}(i, j)$ we use the decoder described by Eisner (1996).

3.4. Word alignment

We used GIZA++ (Och and Ney 2003) to produce word alignments of the dataset. Parameters and training regimen were selected to optimize AER on the word-aligned test set.

3.5. Decoding

We use a tree-based decoder, inspired by dynamic programming. It searches for an approximation of the n -best translations of each subtree of the input dependency tree. Translation candidates are composed from treelet translation pairs extracted from the training corpus. This process is described in more detail in (Quirk et al. 2005). To mitigate the impact of parser error, we translated n -best parses.

3.6. Models

3.6.1. Channel models

We use several channel models: relative frequency estimates of the probability of target given source, as well as lexical weighting scores for source given target and target given source using the word translation table from IBM Model 1 [6]. Probabilities estimated by relative frequency are smoothed using absolute discounting:

$$P_{MLE}(\mathbf{t} | \mathbf{s}) = \frac{c(\mathbf{s}, \mathbf{t}) - \lambda}{c(\mathbf{s}, *)}$$

Here, c represents the count of instances of the treelet pair $\langle \mathbf{s}, \mathbf{t} \rangle$ in the training corpus, and λ is selected to maximize BLEU on a held out test set.

For lexical weighting we compute the sum over all possible alignments of the treelet without normalizing for length. The calculation of source given target is presented below; target given source is calculated symmetrically.

$$P_{M1}(\mathbf{t} | \mathbf{s}) = \prod_{t \in \mathbf{t}} \sum_{s \in \mathbf{s}} P(t | s)$$

We also experimented with other methods of lexical weighting (Ayan and Dorr 2006) but found they had little

3.6.2. Target language models

We use both surface level n -gram language models and a dependency-based bigram language model [7], similar to the bilingual dependency modes used in some English Treebank parsers (e.g. [8]).

$$P_{dep}(\mathbf{T}) = \prod_{i=1}^{|\mathbf{T}|} P_{billex}(\mathbf{T}_i | parent(\mathbf{T}_i))$$

Here P_{billex} is a Kneser-Ney smoothed bigram language model trained on target language dependencies extracted from the aligned parallel dependency tree corpus.

We use two surface n -gram language models integrated with the decoder. One is a word trigram language model trained on the target side of the parallel training data using modified Kneser-Ney smoothing (Chen and Goodman 1999).

The other model was trained on the English Gigaword corpus, as described in 2.2. It is a 5-gram model, trained using modified absolute discount smoothing (Gao et al. 2001), which is a variant of modified Kneser-Ney smoothing. We use count cutoffs of 1 for every n -gram order. The model contains 21 million 2-grams, 951 million 3-grams, 1.6 billion 4-grams, and 1.77 billion 5-grams. It achieves a perplexity of 112 on MT02.

3.6.3. Order model

The order model attempts to assign a probability to the position (pos) of each target node relative to its head based on information in both the source and target trees:

$$P_{order}(order(\mathbf{T}) | \mathbf{S}, \mathbf{T}) = \prod_{t \in \mathbf{T}} P(pos(t, parent(t)) | \mathbf{S}, \mathbf{T})$$

Here, position is modeled in terms of closeness to head in the dependency tree. The closest pre-modifier of given head has position -1; the closest post-modifier has a position 1. Figure 1 shows an example dependency tree pair annotated with head-relative positions.

We use a small set of features reflecting local information in the dependency tree to model $P(pos(t, parent(t)) | \mathbf{S}, \mathbf{T})$:

- Lexical items of t and $parent(t)$.
- Lexical items of the source nodes aligned to t and $head(t)$.
- Part-of-speech ("cat") of the source nodes aligned to the head and modifier.
- Head-relative position of the source node aligned to the source modifier.

These features along with the target feature are gathered from the word-aligned parallel dependency tree corpus and used to train a decision tree (Chickering et al. 2002).

3.6.4. Other models

In addition to these basic models, we also incorporate a variety of other information to gather information about the translation process.

- **Treelet count.** This feature is a count of treelets used to construct the candidate. It acts as a bias toward translations that use a smaller number of treelets; hence toward larger sized treelets incorporating more context.
- **Word count.** We also include a count of the words in the target sentence. This feature helps to offset the bias of the target language model toward shorter sentences.
- **Whole sentence Model 1 scores.** We provide the system with both the probability of the whole source sentence given the whole target sentence and vice versa, as described in [10]:

$$P(\mathbf{S}|\mathbf{T}) = \frac{\epsilon}{|\mathbf{T}|+1} \prod_{s \in \mathbf{S}} \sum_{t \in \mathbf{T}} P(s|t)$$

- **Deletion penalty.** As in [11], we approximate the number of deleted words using Model 1 probabilities using the following formula, where d is an empirically determined threshold:

$$D(\mathbf{S}, \mathbf{T}) = |\{s \in \mathbf{S} \mid \forall t \in \mathbf{T}. P(t|s) < d\}|$$

- **Insertion penalty.** An approximation of the number of insertions can be counted in the same manner:

$$I(\mathbf{S}, \mathbf{T}) = |\{t \in \mathbf{T} \mid \forall s \in \mathbf{S}. P(s|t) < i\}|$$

3.7. Parameter training and tuning

There are several types of tunable parameters. First we have Λ , a multi-dimensional weight vector: one real-valued weight for each feature function. This weight vector is determined by maximization of the BLEU score using n-best lists (Och 2003). However, there are a variety of other parameters that cannot be tuned via n-best lists; instead these are optimized by grid search on BLEU score.

- **Maximum treelet size:** keep treelet translation pairs up to s nodes.
- **Treelet translation pair pruning cutoff:** explore only the top k treelet translation pairs per input node.
- **Decoder beam width:** keep only n best translated subtrees per input subtree.
- **Exhaustive ordering threshold:** fall back to greedy ordering when the count of children whose order is unspecified exceeds this limit; see Quirk et al. (2006) for details.
- **MLE channel model discount.**
- **Deletion and insertion penalty cutoffs.**
- **Default NULL translation probability.**

3.8. Post-processing

As a final step, we restored case to the 1-best translation using a trigram target language model using modified Kneser-Ney smoothing along interpolated with a unigram prior on the case variant of each word. The weight for this prior was selected via a grid search to maximize BLEU score.

A small set of rules handled translation of Chinese numbers and dates. Also a regular expression detected Chinese news datelines and bylines; another small set of rules was used to translate these pieces according to the format shown seen prior NIST competitions.

4. Acknowledgements

Our thanks to Robert C. Moore and Kristina Toutanova for many informative discussions, and to Galen Andrew for graciously providing his word segmentation component.

5. References

- [1] Quirk, C., Menezes, A., and Cherry, C., "Dependency Tree Translation: Syntactically Informed Phrasal SMT", *Proceedings of ACL 2005*, Ann Arbor, MI, USA, 2005.
- [2] Och, F. J., and Ney, H., "Discriminative Training and Maximum Entropy Models for Statistical Machine Translation", *Proceedings of ACL 2002*, Philadelphia, PA, USA, 2002.
- [3] Heidorn, G., "Intelligent writing assistance", in Dale et al. *Handbook of Natural Language Processing*, Marcel Dekker, 2000.
- [4] Och, F. J., and Ney H., "A Systematic Comparison of Various Statistical Alignment Models", *Computational Linguistics*, 29(1):19-51, March 2003.
- [5] Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J., "BLEU: a method for automatic evaluation of machine translation", *Proceedings of ACL 2002*, Philadelphia, PA, USA, 2002.
- [6] Brown, P. F., Della Pietra, S., Della Pietra, V. J., and Mercer, R. L., "The Mathematics of Statistical Machine Translation: Parameter Estimation", *Computational Linguistics* 19(2): 263-311, 1994.
- [7] Aue, A., Menezes, A., Moore, R., Quirk, C., and Ringger, E., "Statistical Machine Translation Using Labeled Semantic Dependency Graphs." *Proceedings of TMI 2004*, Baltimore, MD, USA, 2004.
- [8] Collins, M., "Three generative, lexicalised models for statistical parsing", *Proceedings of ACL 1997*, Madrid, Spain, 1997.
- [9] Chickering, D.M., "The WinMine Toolkit", Microsoft Research Technical Report MSR-TR-2002-103, Redmond, WA, USA, 2002.
- [10] Och, F. J., Gildea, D., Khudanpur, S., Sarkar, A., Yamada, K., Fraser, A., Kumar, S., Shen, L., Smith, D., Eng, K., Jain, V., Jin, Z., and Radev, D., "A Smorgasbord of Features for Statistical Machine Translation". *Proceedings of HLT/NAACL 2004*, Boston, MA, USA, 2004.
- [11] Bender, O., Zens, R., Matsuov, E. and Ney, H., "Alignment Templates: the RWTH SMT System". *IWSLT Workshop at INTERSPEECH 2004*, Jeju Island, Korea, 2004.
- [12] Och, F. J., "Minimum Error Rate Training for Statistical Machine Translation", *Proceedings of ACL 2003*, Sapporo, Japan, 2003.
- [13] Andrew, G., "A Hybrid Markov/Semi-Markov Conditional Random Field for Sequence Segmentation," *Proceedings of EMNLP 2006*, pages 465-472, Sydney, July 2006

- [14] Tesnière, L., *Éléments de syntaxe structurale*, Librairie C. Klincksieck, 1959.
- [15] Corston-Oliver, S. H., Aue, A., Duh, K. and Ringger, E., “Multilingual Dependency Parsing Using Bayes Point Machines”. *Proceedings of HLT/NAACL 2006*, New York, USA.
- [16] Mel’čuk, I. A., *Dependency Syntax: Theory and Practice*, State University of New York Press, 1988.
- [17] Marcus, M., Santorini, B., Marcinkiewicz, M., “Building a Large Annotated Corpus of English: The Penn Treebank”, *Computational Linguistics* 19(2):313-330, 1993.
- [18] Collins, M., “Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms”, *Proceedings of EMNLP*, 2002.
- [19] Eisner, J. M., “Three New Probabilistic Models for Dependency Parsing: An Exploration”, *Proceedings of COLING*, 340-345, 1996.
- [20] Harrington, E., Herbrich, R., Kivinen, J., Platt, J. C., and Williamson, R. C., “Online Bayes Point Machines”, *Proceedings of the 7th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 241-252, 2003.
- [21] Herbrich, R., Graepel, T., Campbell, C., “Bayes Point Machines”, *Journal of Machine Learning Research* 2001:245-278.
- [22] Xue, N., Xia, F., Chiou, F., Palmer, M., “The Penn Chinese Treebank: Phrase Structure Annotation of a Large Corpus”, *Natural Language Engineering* 11(2), 2005.
- [23] Collins, M. J., *Head-driven Statistical Models for Natural Language Processing*, Ph.D. Thesis, University of Pennsylvania, 1999.
- [24] Gao, Jianfeng, Joshua Goodman and Jiangbo Miao. 2001. The use of clustering techniques for language model – application to Asian language. *Computational Linguistics and Chinese Language Processing*. Vol. 6, No. 1, pp.27-60.
- [25] Chen, S. F., Goodman, J. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech and Language* 13 (Oct.), 359-394.