KyotoEBMT System Description for the 1st Workshop on Asian Translation

John Richardson[†] **Fabien Cromières**[‡] **Toshiaki Nakazawa**[‡] **Sadao Kurohashi**[†]
[†]Graduate School of Informatics, Kyoto University, Kyoto 606-8501
[‡]Japan Science and Technology Agency, Kawaguchi-shi, Saitama 332-0012
john@nlp.ist.i.kyoto-u.ac.jp, {fabien, nakazawa}@pa.jst.jp,
kuro@i.kyoto-u.ac.jp

Abstract

This paper introduces the KyotoEBMT Example-Based Machine Translation framework. Our system uses a *tree-to-tree* approach, employing syntactic dependency analysis for both source and target languages in an attempt to preserve non-local structure. The effectiveness of our system is maximized with online example matching and a flexible decoder. Evaluation demonstrates BLEU scores competitive with state-of-the-art SMT baselines. The system implementation is available as open-source.

1 Introduction

Corpus-based approaches have become a major focus of Machine Translation research. We present here a fully-fledged Example-Based Machine Translation (EBMT) platform making use of both source-language and target-language dependency structure. This paradigm has been explored comparatively less, as studies on Syntactic-based SMT/EBMT tend to focus on constituent trees rather than dependency trees, and on tree-to-string rather than tree-to-tree approaches. Furthermore, we employ separate dependency parsers for each language rather than projecting the dependencies from one language to another, as in (Quirk et. al, 2005).

The dependency structure information is used end-to-end: for improving the quality of the alignment of the translation examples, for constraining the translation rule extraction and for guiding the decoding. We believe that dependency structure, which considers more than just local context, is important in order to generate fluent and accurate translations of complex sentences across distant language pairs.

Our experiments focus on technical domain translation for Japanese-Chinese and Japanese-English, however our implementation is applicable to any domain and language pair for which there exist translation examples and dependency parsers.

A further unique characteristic of our system is that, again contrary to the majority of similar systems, it does not rely on precomputation of translation rules. Instead it matches each input sentence to the full database of translation examples before extracting translation rules online. This has the merit of maximizing the information available when creating and combining translation rules, while retaining the ability to produce excellent translations for input sentences similar to an existing translation example.

The system is mostly developed in C++ and incorporates a web-based translation interface for ease of use. The web interface (see Figure 1) also displays information useful for error analysis such as the list of translation examples used. Experiments are facilitated through the inclusion of a curses-based graphical interface for performing tuning and evaluation. The decoder supports multiple threads.

The system has been made available as open-source. The code can be downloaded from http://nlp.ist.i.kyoto-u.ac.jp/kyotoebmt/.

2 System Overview

Figure 2 shows the basic structure of the Kyoto EBMT translation pipeline.

The training process begins with parsing and aligning parallel sentences from the training corpus. Alignment uses a Bayesian subtree alignment model based on dependency

Proceedings of the 1st Workshop on Asian Translation (WAT2014), pages 83–88,
Tokyo, Japan, 4th October 2014.

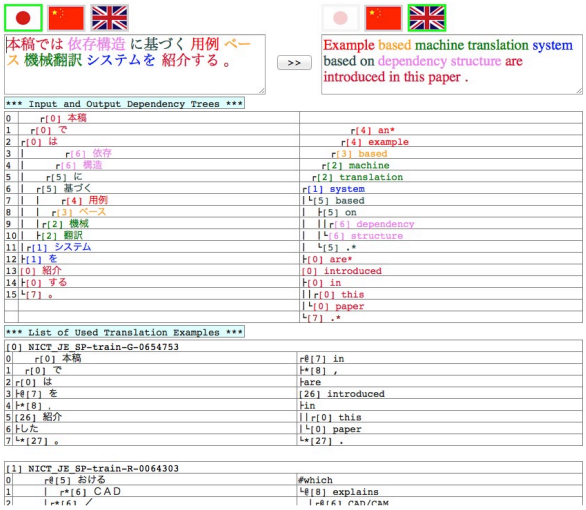

Figure 1: A screenshot of the web interface showing a Japanese-English translation. The interface provides the source and target side dependency tree, as well as the list of examples used with their alignments. The web interface facilitates easy and intuitive error analysis, and can be used as a tool for computer-aided translation.

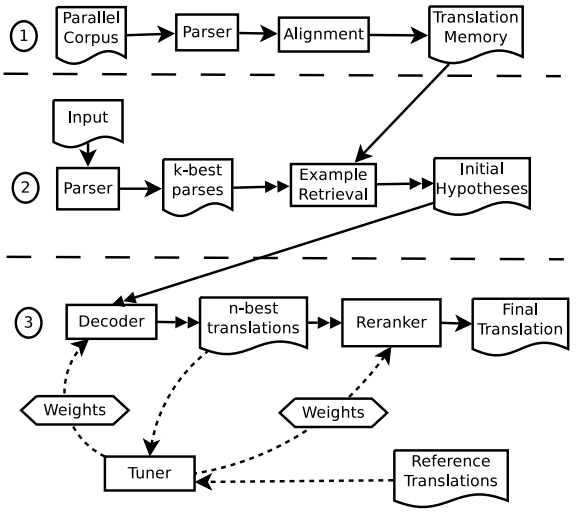

Figure 2: The translation pipeline can be roughly divided in 3 steps. Step 1 is the creation of the example database, trained from a parallel corpus. Step 2 is the k-best parsing of an input sentence and the generation of k sets of initial hypotheses. Step 3 consists in decoding and reranking. The tuning of the weights for decoding and reranking is done by a modified version of step 3. The double arrows indicate the parallel processing of each of the k parse of the input sentence.

trees. This contains a tree-based reordering model and can capture non-local reorderings, which sequential word-based models often cannot handle effectively. The alignments are then used to build an example database ('translation memory') containing 'examples' or 'treelets' that form the hypotheses to be combined during decoding.

Translation is performed by first parsing an input sentence then searching for treelets matching entries in the example database. The retrieved treelets are combined by a decoder that optimizes a log linear model score. Finally, a reranker select an optimal translation in the n-best list provided by the decoder using additional non-local features (see section 6).

In order to be more robust with respect to parse errors, the input sentence can be parsed into a k-best list of posible parses. Each parse is then translated separately and all translations are merged before the final reranking step (see section 7).

The example retrieval and decoding steps are explained in more detail in sections 3 and 4 respectively. The choice of features and the tuning of the log linear model is described in section 5.

Figure 3 shows the process of combining examples matching the input tree to create an output sentence.

3 Example retrieval and translation hypothesis construction

An important characteristic of our system is that we do not extract and store translation rules in advance: the alignment of translation examples is performed offline. However, for a given input sentence i, the steps for finding examples partially matching i and extracting their translation hypotheses is an online process. This approach could be considered to be more faithful to the original EBMT approach advocated by Nagao (1984). It has already been proposed for phrase-based (Callison-Burch et al., 2005), hierarchical (Lopez, 2007), and syntax-based (Cromières and Kurohashi, 2011) systems. It does not however, seem to be very commonly integrated in syntax-based MT.

This approach has several benefits. The first

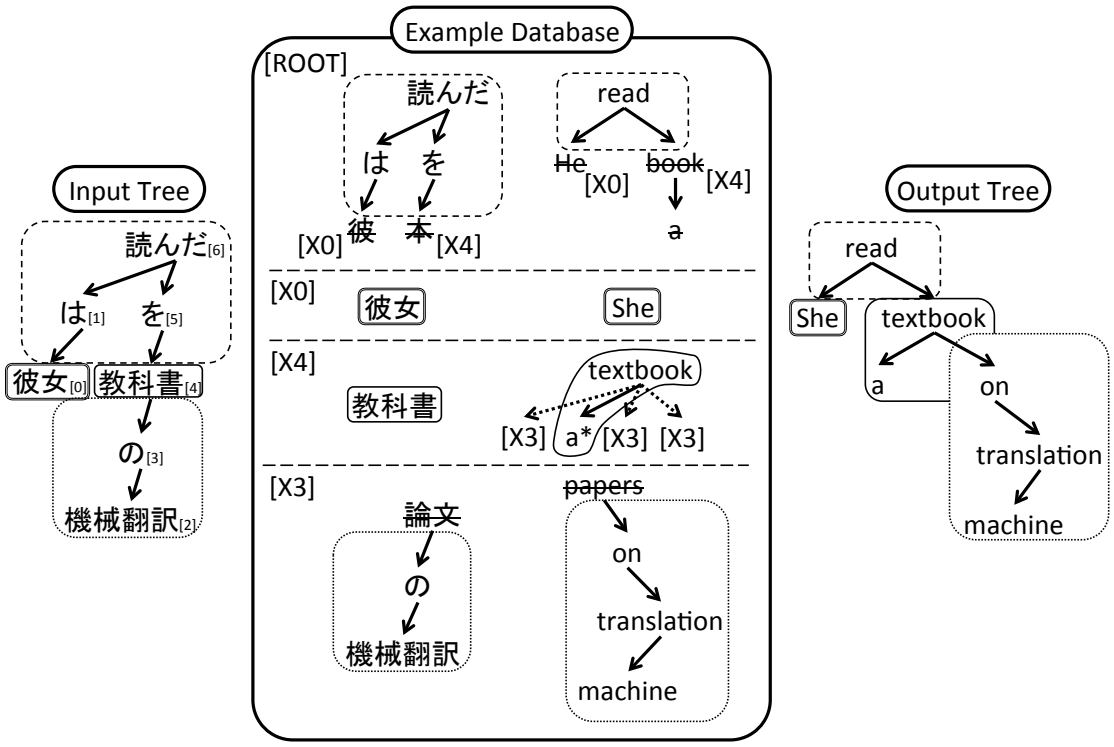

Figure 3: The process of translation. The source sentence is parsed and matching subtrees from the example database are retrieved. From the examples, we extract translation hypotheses than can contain optional target words and several position for each non-terminals. For example the translation hypothesis containing "textbook" has three possible position for the non-terminal X3 (as a left-child before "a", as a left-child after "a" or as a right-child). The translation hypotheses are then combined during decoding. Choice of optional words and final Non-Terminal positions is also done during decoding.

is that we are not required to impose a limit on the size of translation hypotheses. Systems extracting rules in advance typically restrict the size and number of extracted rules for fear of becoming unmanageable. In particular, if an input sentence is the same or very similar to one of our translation examples, we will be able to retrieve a perfect translation. A second advantage is that we can make use of the full context of the example to assign features and scores to each translation hypothesis.

The main drawback of our approach is that it can be computationally more expensive to retrieve arbitrarily large matchings in the example database online than it is to match precomputed rules. We use the techniques described in (Cromières and Kurohashi, 2011) to perform this step as efficiently as possible.

Once we have found an example translation (s, t) for which s partially matches i, we proceed to extract a translation hypothesis from it. A translation hypothesis is defined as a generic translation rule for a part p of the input sentence that is represented as a target-language treelet, with non-terminals representing the insertion positions for the translations of other parts of the sentence. A translation hypothesis is created from a translation example as follows:

1. We project the part of s that is matched into the target side t using the alignment of s and t. This is trivial if each word of s and t is aligned, but this is not typically the case. Therefore our translation hypotheses will often have some target words/nodes marked as *optionals*: this means that we will decide if they should be added to the final translation only at the moment of combination.

2. We insert the non-terminals as child nodes of the projected subtree. This is

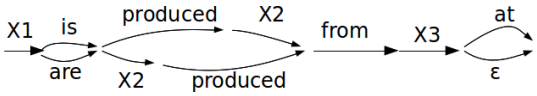

Figure 4: A translation hypothesis endoded as a lattice. This representation allows us to handle efficiently the ambiguities of our translation rules. Note that each path in this lattice corresponds to different choices of insertion position for X2, morphological forms of "be", and the optional insertion of "at".

simple if i, s and t have the same structure and are perfectly aligned, but again this is not typically the case. A consequence is that we will sometimes have *several possible insertion positions* for each non-terminal. The choice of insertion position is again made during combination.

4 Decoding

After having extracted translation hypotheses for as many parts of the input tree as possible, we need to decide how to select and combine them. Our approach here is similar to what has been proposed for Corpus-Based Machine Translation. We first choose a number of features and create a linear model scoring each possible combination of hypotheses (see Section 5). We then attempt to find the combination that maximizes this model score.

The combination of rules is constrained by the structure of the input dependency tree. If we only consider local features[1], then a simple bottom-up dynamic programming approach can efficiently find the optimal combination with linear $O(|\mathcal{H}|)$ complexity[2]. However, non-local features (such as language models) will force us to prune the search space. This pruning is done efficiently through a variation of cube-pruning (Chiang, 2007). We use KenLM[3] (Heafield, 2011) for computing the target language model score. Decoding is made more efficient by using some of the more advanced features of KenLM such as state-reduction ((Li and Khudanpur, 2008), (Heafield et al., 2011)) and rest-cost estimations(Heafield et al., 2012).

[1]The score of a combination will be the sum of the local scores of each translation hypothesis.

[2]$\mathcal{H}$ = set of translation hypotheses

[3]http://kheafield.com/code/kenlm/

Compared with the original cube-pruning algorithm, our decoder is designed to handle an arbitrary number of non-terminals. In addition, as we have seen in Section 3, the translation hypotheses we initially extract from examples are ambiguous in term of which target word is going to be used and which will be the final position of each non-terminal. In order to handle such ambiguities, we use a lattice-based internal representation that can encode them efficiently (see Figure 4). This lattice representation also allows the decoder to make choices between various morphological variations of a word (e.g. be/is/are). We use the decoding algorithm described in (Cromieres and Kurohashi, 2014).

5 Features and Tuning

During decoding we use a linear model to score each possible combination of hypotheses. This linear model is based on a linear combination of both local features (local to each translation hypothesis) and non-local features (such as a 5-gram language model score of the final translation). The decoder considers in total a combination of 42 features, a selection of which are given below.

- Example penalty and example size
- Translation probability
- Language model score
- Optional words added/removed

The optimal weights for each feature are estimated using the implementation of k-best batch MIRA (Cherry and Foster, 2012) included in Moses.

6 Reranking

We reranked the n-best output of our system using an additional two language models: a standard 7-gram language model with Modified Kneser-Ney smoothing and a Recurrent Neural Network Language Model (RNNLM) (Mikolov et. al, 2010). The RNNLM model was trained with hidden layer size 200, and 5000 sentences from the training fold were used as validation data.

Reranking was conducted by first calculating the various language model scores for each

translation in the n-best list. These features were added to those used in the first round of tuning, then one final iteration of tuning was run. The tuning algorithm and settings were the same as for standard tuning. This retuning step was added in order to find an optimal combination of the two additional LMs with related features such as sentence length and the score given by the 5-gram language model used inside the decoder.

7 K-best parsing

We found that the quality of the source-side dependency parsing had an important impact on translation quality. Unfortunately, parsing errors are unavoidable. Chinese parsing is maybe especially challenging and our Chinese parser still produces a significant number of parsing errors. In order to mitigate this problem, we use a k-best list of parses of the input sentence. Each parse is translated in parallel, and the list of translation thus obtained is merged before the reranking step. In the future, we intend to move from a k-best list representation of multiple parses to a more compact and efficient forest representation. Furthermore, we will also have to consider multiple parses for all the translation examples (and not just the input sentence).

8 Experiments

The following dependency parsers were used. The scores in parentheses are the approximate parsing accuracies (micro-average), which were evaluated by hand on a random subset of sentences from the test data. The parsers were trained on domains different to those used in the experiments.

- English: NLParser[4] (92%) (Charniak and Johnson, 2005)
- Japanese: KNP (96%) (Kawahara and Kurohashi, 2006)
- Chinese: SKP (88%) (Shen et al., 2012)

K-best dependency parsing was done with k set to 20.

[4]Converted to dependency parses with in-house tool.

8.1 Results

The results shown are for evaluation on the test set after tuning. Tuning was conducted over 20 iterations on the development set using an n-best list of length 500.

9 Conclusion

We have described the Kyoto example-based translation system. It uses both source and target dependency analysis, as well as online example retrieving, allowing the availability of full translation examples at translation time.

We believe that the use of dependency parsing is important for accurate translation across distant language pairs, especially in settings such as ours with many long sentences. We have designed a complete translation framework around this idea, using dependency-parsed trees at each step from alignment to example retrieval to example combination.

References

Eugene Charniak and Mark Johnson. 2005. Coarse-to-Fine n-Best Parsing and MaxEnt Discriminative Reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics, ACL 2005.*

Fabien Cromières and Sadao Kurohashi. 2011. Efficient retrieval of tree translation examples for syntax-based machine translation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing.*

Isao Goto, Ka Po Chow, Bin Lu, Eiichiro Sumita and Benjamin Tsou. 2013. Overview of the Patent Machine Translation Task at the NTCIR-10 Workshop. In *Proceedings of the 10th NTCIR Workshop Meeting on Evaluation of Information Access Technologies (NTCIR-10).*

Daisuke Kawahara and Sadao Kurohashi. 2006. A Fully-Lexicalized Probabilistic Model for Japanese Syntactic and Case Structure Analysis. In *Proceedings of the Human Language Technology Conference of the NAACL.*

Makoto Nagao. 1984. A framework of a mechanical translation between Japanese and English by analogy principle. In *A. Elithorn and R. Banerji. Artificial and Human Intelligence.*

Toshiaki Nakazawa and Sadao Kurohashi. 2012. Alignment by bilingual generation and monolingual derivation. In *Proceedings of COLING 2012.*

Language Pair	Reranking	BLEU	RIBES	HUMAN
JA–EN	No	20.60	70.12	21.50
	Yes	21.07	69.90	25.00
EN–JA	No	29.76	75.21	33.75
	Yes	31.09	75.96	38.00
JA–ZH	No	27.21	79.13	-0.75
	Yes	27.67	78.83	-8.75
ZH–JA	No	33.57	80.10	6.00
	Yes	34.75	80.26	7.50

Table 1: Scores

Mo Shen, Daisuke Kawahara and Sadao Kurohashi. 2012. A Reranking Approach for Dependency Parsing with Variable-sized Subtree Features. In *Proceedings of 26th Pacific Asia Conference on Language Information and Computing*.

Chris Callison-Burch, Colin Bannard, and Josh Schroeder. 2005. Scaling phrase-based statistical machine translation to larger corpora and longer phrases. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 255–262. Association for Computational Linguistics, 2005.

Colin Cherry and George Foster. 2012. Batch Tuning Strategies for Statistical Machine Translation. In *HLT-NAACL*, 2012.

David Chiang. 2007. Hierarchical phrase-based translation. In *Computational Linguistics*.

Cromieres Fabien and Sadao Kurohashi. 2014. Translation Rules with Right-Hand Side Lattices. In *Proceedings of EMNLP 2014*

Kenneth Heafield. 2011. KenLM: faster and smaller language model queries. In *Proceedings of the EMNLP 2011 Sixth Workshop on Statistical Machine Translation*, 2011.

Kenneth Heafield, Hieu Hoang, Philipp Koehn, Tetsuo Kiso, and Marcello Federico. 2011. Left language model state for syntactic machine translation. In *Proceedings of the International Workshop on Spoken Language Translation*, 2011.

Kenneth Heafield, Philipp Koehn, and Alon Lavie. 2012. Language model rest costs and space-efficient storage. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 2012.

Zhifei Li and Sanjeev Khudanpur. 2008. A scalable decoder for parsing-based machine translation with equivalent language model state maintenance. In *Proceedings of the Second Workshop on Syntax and Structure in Statistical Translation*. Association for Computational Linguistics, 2008.

Adam Lopez. 2007. Hierarchical phrase-based translation with suffix arrays. In *EMNLP-CoNLL*, 2007.

Tomas Mikolov, Martin Karafiat, Lukas Burget, Jan Cernocky and Sanjeev Khudanpur. 2010. Recurrent Neural Network Based Language Model. In *Proceedings of the 11th Annual Conference of the International Speech Communication Association*, 2010.

Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency Treelet Translation: Syntactically Informed Phrasal SMT. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2005.