Akamon: An Open Source Toolkit for Tree/Forest-Based Statistical Machine Translation*

Xianchao Wu[†], **Takuya Matsuzaki**[*], **Jun'ichi Tsujii**[‡]
[†] Baidu Inc.
[*] National Institute of Informatics
[‡] Microsoft Research Asia
wuxianchao@gmail.com, takuya-matsuzaki@nii.ac.jp, jtsujii@microsoft.com

Abstract

We describe **Akamon**, an open source toolkit for tree and forest-based statistical machine translation (Liu et al., 2006; Mi et al., 2008; Mi and Huang, 2008). Akamon implements all of the algorithms required for tree/forest-to-string decoding using tree-to-string translation rules: *multiple-thread* forest-based decoding, n-gram language model integration, beam- and cube-pruning, k-best hypotheses extraction, and minimum error rate training. In terms of tree-to-string translation rule extraction, the toolkit implements the traditional maximum likelihood algorithm using PCFG trees (Galley et al., 2004) and HPSG trees/forests (Wu et al., 2010).

1 Introduction

Syntax-based statistical machine translation (SMT) systems have achieved promising improvements in recent years. Depending on the type of input, the systems are divided into two categories: *string-based* systems whose input is a string to be simultaneously parsed and translated by a synchronous grammar (Wu, 1997; Chiang, 2005; Galley et al., 2006; Shen et al., 2008), and *tree/forest-based* systems whose input is already a parse tree or a packed forest to be directly converted into a target tree or string (Ding and Palmer, 2005; Quirk et al., 2005; Liu et al., 2006; Huang et al., 2006; Mi et al., 2008; Mi and Huang, 2008; Zhang et al., 2009; Wu et al., 2010; Wu et al., 2011a).

Depending on whether or not parsers are explicitly used for obtaining linguistically annotated data during training, the systems are also divided into two categories: *formally syntax-based* systems that do not use additional parsers (Wu, 1997; Chiang, 2005; Xiong et al., 2006), and *linguistically syntax-based* systems that use PCFG parsers (Liu et al., 2006; Huang et al., 2006; Galley et al., 2006; Mi et al., 2008; Mi and Huang, 2008; Zhang et al., 2009), HPSG parsers (Wu et al., 2010; Wu et al., 2011a), or dependency parsers (Ding and Palmer, 2005; Quirk et al., 2005; Shen et al., 2008). A classification[1] of syntax-based SMT systems is shown in Table 1.

Translation rules can be extracted from aligned string-string (Chiang, 2005), tree-tree (Ding and Palmer, 2005) and tree/forest-string (Galley et al., 2004; Mi and Huang, 2008; Wu et al., 2011a) data structures. Leveraging structural and linguistic information from parse trees/forests, the latter two structures are believed to be better than their string-string counterparts in handling non-local reordering, and have achieved promising translation results. Moreover, the tree/forest-string structure is more widely used than the tree-tree structure, presumably because using two parsers on the source and target languages is subject to more problems than making use of a parser on one language, such as the shortage of high precision/recall parsers for languages other than English, compound parse error rates, and inconsistency of errors. In Table 1, note that tree-to-string rules are generic and applicable to many syntax-based models such as tree/forest-to-

*Work done when all the authors were in The University of Tokyo.

[1]This classification is inspired by and extends the Table 1 in (Mi and Huang, 2008).

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 127–132,
Jeju, Republic of Korea, 8-14 July 2012.

Source-to-target	Examples (partial)	Decoding	Rules	Parser
tree-to-tree	(Ding and Palmer, 2005)	↓	dep.-to-dep.	DG
forest-to-tree	(Liu et al., 2009a)	↓ ↑↓	tree-to-tree	PCFG
tree-to-string	(Liu et al., 2006)	↑	*tree-to-string*	PCFG
	(Quirk et al., 2005)	↑	dep.-to-string	DG
forest-to-string	(Mi et al., 2008)	↓ ↑↓	*tree-to-string*	PCFG
	(Wu et al., 2011a)	↓ ↑↓	*tree-to-string*	HPSG
string-to-tree	(Galley et al., 2006)	CKY	*tree-to-string*	PCFG
	(Shen et al., 2008)	CKY	string-to-dep.	DG
string-to-string	(Chiang, 2005)	CKY	string-to-string	none
	(Xiong et al., 2006)	CKY	string-to-string	none

Table 1: A classification of syntax-based SMT systems. Tree/forest-based and string-based systems are split by a line. All the systems listed here are linguistically syntax-based except the last two (Chiang, 2005) and (Xiong et al., 2006), which are formally syntax-based. DG stands for dependency (abbreviated as dep.) grammar. ↓ and ↑ denote top-down and bottom-up traversals of a source tree/forest.

string models and string-to-tree model.

However, few tree/forest-to-string systems have been made open source and this makes it difficult and time-consuming to testify and follow existing proposals involved in recently published papers. The Akamon system[2], written in Java and following the tree/forest-to-string research direction, implements all of the algorithms for both tree-to-string translation rule extraction (Galley et al., 2004; Mi and Huang, 2008; Wu et al., 2010; Wu et al., 2011a) and tree/forest-based decoding (Liu et al., 2006; Mi et al., 2008). We hope this system will help related researchers to catch up with the achievements of tree/forest-based translations in the past several years without re-implementing the systems or general algorithms from scratch.

2 Akamon Toolkit Features

Limited by the successful parsing rate and coverage of linguistic phrases, Akamon currently achieves comparable translation accuracies compared with the most frequently used SMT baseline system, Moses (Koehn et al., 2007). Table 2 shows the automatic translation accuracies (case-sensitive) of Akamon and Moses. Besides BLEU and NIST score, we further list RIBES score[3], , i.e., the software implementation of Normalized Kendall's τ as proposed by (Isozaki et al., 2010a) to automatically evaluate the translation between distant language pairs based on rank correlation coefficients and significantly penalizes word order mistakes.

In this table, Akamon-Forest differs from Akamon-Comb by using different configurations: Akamon-Forest used only 2/3 of the total training data (limited by the experiment environments and time). Akamon-Comb represents the system combination result by combining Akamon-Forest and other phrase-based SMT systems, which made use of pre-ordering methods of head finalization as described in (Isozaki et al., 2010b) and used the total 3 million training data. The detail of the pre-ordering approach and the combination method can be found in (Sudoh et al., 2011) and (Duh et al., 2011).

Also, Moses (hierarchical) stands for the hierarchical phrase-based SMT system and Moses (phrase) stands for the flat phrase-based SMT system. For intuitive comparison (note that the result achieved by Google is only for reference and not a comparison, since it uses a different and unknown training data) and following (Goto et al., 2011), the scores achieved by using the Google online translation system[4] are also listed in this table.

Here is a brief description of Akamon's main features:

- *multiple-thread* forest-based decoding: Akamon first loads the development (with source and reference sentences) or test (with source sentences only) file into memory and then perform parameter tuning or decoding in a parallel way. The forest-based decoding algorithm is alike that described in (Mi et al., 2008),

[2]Code available at https://sites.google.com/site/xianchaowu2012

[3]Code available at http://www.kecl.ntt.co.jp/icl/lirg/ribes

[4]http://translate.google.com/

Systems	BLEU	NIST	RIBES
Google online	0.2546	6.830	0.6991
Moses (hierarchical)	0.3166	7.795	0.7200
Moses (phrase)	0.3190	7.881	0.7068
Moses (phrase)*	0.2773	6.905	0.6619
Akamon-Forest*	0.2799	7.258	0.6861
Akamon-Comb	0.3948	8.713	0.7813

Table 2: Translation accuracies of Akamon and the baseline systems on the NTCIR-9 English-to-Japanese translation task (Wu et al., 2011b). * stands for only using 2 million parallel sentences of the total 3 million data. Here, HPSG forests were used in Akamon.

i.e., first construct a *translation forest* by applying the tree-to-string translation rules to the original parsing forest of the source sentence, and then collect k-best hypotheses for the root node(s) of the translation forest using Algorithm 2 or Algorithm 3 as described in (Huang and Chiang, 2005). Later, the k-best hypotheses are used both for parameter tuning on additional development set(s) and for final optimal translation result extracting.

- language models: Akamon can make use of one or many n-gram language models trained by using SRILM[5] (Stolcke, 2002) or the Berkeley language model toolkit, berkeleylm-1.0b3[6] (Pauls and Klein, 2011). The weights of multiple language models are tuned under minimum error rate training (MERT) (Och, 2003).

- pruning: traditional beam-pruning and cube-pruning (Chiang, 2007) techniques are incorporated in Akamon to make decoding feasible for large-scale rule sets. Before decoding, we also perform the marginal probability-based inside-outside algorithm based pruning (Mi et al., 2008) on the original parsing forest to control the decoding time.

- MERT: Akamon has its own MERT module which optimizes weights of the features so as to maximize some automatic evaluation metric, such as BLEU (Papineni et al., 2002), on a development set.

[5]http://www.speech.sri.com/projects/srilm/
[6]http://code.google.com/p/berkeleylm/

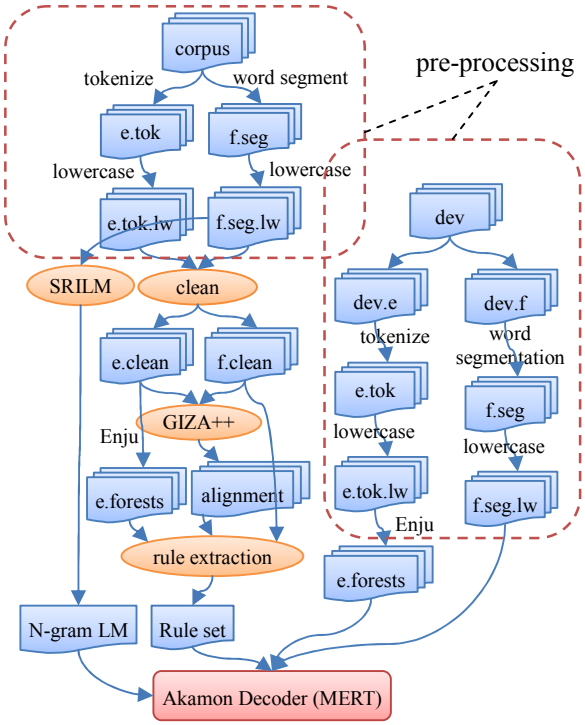

Figure 1: Training and tuning process of the Akamon system. Here, e = source English language, f = target foreign language.

- translation rule extraction: as former mentioned, we extract tree-to-string translation rules for Akamon. In particular, we implemented the GHKM algorithm as proposed by Galley et al. (2004) from word-aligned tree-string pairs. In addition, we also implemented the algorithms proposed by Mi and Huang (2008) and Wu et al. (2010) for extracting rules from word-aligned PCFG/HPSG forest-string pairs.

3 Training and Decoding Frameworks

Figure 1 shows the training and tuning progress of the Akamon system. Given original bilingual parallel corpora, we first tokenize and lowercase the source and target sentences (e.g., word segmentation of Chinese and Japanese, punctuation segmentation of English).

The pre-processed monolingual sentences will be used by SRILM (Stolcke, 2002) or BerkeleyLM (Pauls and Klein, 2011) to train a n-gram language model. In addition, we filter out too long sentences

here, i.e., only relatively short sentence pairs will be used to train word alignments. Then, we can use GIZA++ (Och and Ney, 2003) and symmetric strategies, such as *grow-diag-final* (Koehn et al., 2007), on the tokenized parallel corpus to obtain a word-aligned parallel corpus.

The source sentence and its packed forest, the target sentence, and the word alignment are used for tree-to-string translation rule extraction. Since a 1-best tree is a special case of a packed forest, we will focus on using the term 'forest' in the continuing discussion. Then, taking the target language model, the rule set, and the preprocessed development set as inputs, we perform MERT on the decoder to tune the weights of the features.

The Akamon forest-to-string system includes the decoding algorithm and the rule extraction algorithm described in (Mi et al., 2008; Mi and Huang, 2008).

Feature	Description
CAT	phrasal category
XCAT	fine-grained phrasal category
SCHEMA	name of the schema applied in the node
HEAD	*pointer* to the head daughter
SEM_HEAD	*pointer* to the semantic head daughter
CAT	syntactic category
POS	Penn Treebank-style part-of-speech tag
BASE	base form
TENSE	tense of a verb (past, present, untensed)
ASPECT	aspect of a verb (none, perfect, progressive, perfect-progressive)
VOICE	voice of a verb (passive, active)
AUX	auxiliary verb or not (minus, modal, have, be, do, to, copular)
LEXENTRY	lexical entry, with supertags embedded
PRED	type of a predicate
ARG$\langle x \rangle$	*pointer* to semantic arguments, $x = 1..4$

Table 3: Syntactic/semantic features extracted from HPSG signs that are included in the output of Enju. Features in phrasal nodes (top) and lexical nodes (bottom) are listed separately.

4 Using Deep Syntactic Structures

In Akamon, we support the usage of *deep syntactic structures* for obtaining fine-grained translation rules as described in our former work (Wu et al., 2010)[7]. Similarly, Enju[8], a state-of-the-art and freely available HPSG parser for English, can be used to generate packed parse forests for source sentences[9]. Deep syntactic structures are included in the HPSG trees/forests, which includes a fine-grained description of the syntactic property and a semantic representation of the sentence. We extract fine-grained rules from aligned HPSG forest-string pairs and use them in the forest-to-string decoder. The detailed algorithms can be found in (Wu et al., 2010; Wu et al., 2011a). Note that, in Akamon, we also provide the codes for generating HPSG forests from Enju.

Head-driven phrase structure grammar (HPSG) is a lexicalist grammar framework. In HPSG, linguistic entities such as words and phrases are represented by a data structure called a *sign*. A sign gives a factored representation of the syntactic features of a word/phrase, as well as a representation of their semantic content. Phrases and words represented by signs are composed into larger phrases by applications of *schemata*. The semantic representation of the new phrase is calculated at the same time. As such, an HPSG parse tree/forest can be considered as a tree/forest of signs (c.f. the HPSG forest in Figure 2 in (Wu et al., 2010)).

An HPSG parse tree/forest has two attractive properties as a representation of a source sentence in syntax-based SMT. First, we can carefully control the condition of the application of a translation rule by exploiting the fine-grained syntactic description in the source parse tree/forest, as well as those in the translation rules. Second, we can identify sub-trees in a parse tree/forest that correspond to basic units of the semantics, namely sub-trees covering a predicate and its arguments, by using the semantic representation given in the signs. Extraction of translation rules based on such *semantically-connected* sub-trees is expected to give a compact and effective set of translation rules.

A sign in the HPSG tree/forest is represented by a typed feature structure (TFS) (Carpenter, 1992). A TFS is a directed-acyclic graph (DAG) wherein the edges are labeled with feature names and the nodes

[7]However, Akamon still support PCFG tree/forest based translation. A special case is to yield PCFG style trees/forests by ignoring the rich features included in the nodes of HPSG trees/forests and only keep the POS tag and the phrasal categories.

[8]http://www-tsujii.is.s.u-tokyo.ac.jp/enju/index.html

[9]Until the date this paper was submitted, Enju supports generating English and Chinese forests.

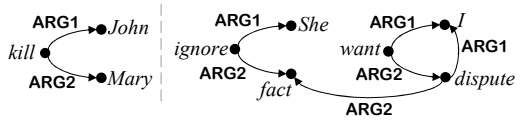

Figure 2: Predicate argument structures for the sentences of "*John killed Mary*" and "*She ignored the fact that I wanted to dispute*".

(feature values) are typed. In the original HPSG formalism, the types are defined in a hierarchy and the DAG can have arbitrary shape (e.g., it can be of any depth). We however use a simplified form of TFS, for simplicity of the algorithms. In the simplified form, a TFS is converted to a (flat) set of pairs of feature names and their values. Table 3 lists the features used in our system, which are a subset of those in the original output from Enju.

In the Enju English HPSG grammar (Miyao et al., 2003) used in our system, the semantic content of a sentence/phrase is represented by a predicate-argument structure (PAS). Figure 2 shows the PAS of a simple sentence, "*John killed Mary*", and a more complex PAS for another sentence, "*She ignored the fact that I wanted to dispute*", which is adopted from (Miyao et al., 2003). In an HPSG tree/forest, each leaf node generally introduces a predicate, which is represented by the pair of LEXENTRY (lexical entry) feature and PRED (predicate type) feature. The arguments of a predicate are designated by the pointers from the ARG$\langle x \rangle$ features in a leaf node to non-terminal nodes. Consequently, Akamon includes the algorithm for extracting compact composed rules from these PASs which further lead to a significant fast tree-to-string decoder. This is because it is not necessary to exhaustively generate the subtrees for all the tree nodes for rule matching any more. Limited by space, we suggest the readers to refer to our former work (Wu et al., 2010; Wu et al., 2011a) for the experimental results, including the training and decoding time using standard English-to-Japanese corpora, by using deep syntactic structures.

5 Content of the Demonstration

In the demonstration, we would like to provide a brief tutorial on:

- describing the format of the packed forest for a source sentence,
- the training script on translation rule extraction,
- the MERT script on feature weight tuning on a development set, and,
- the decoding script on a test set.

Based on Akamon, there are a lot of interesting directions left to be updated in a relatively fast way in the near future, such as:

- integrate target dependency structures, especially target dependency language models, as proposed by Mi and Liu (2010),
- better pruning strategies for the input packed forest before decoding,
- derivation-based combination of using other types of translation rules in one decoder, as proposed by Liu et al. (2009b), and
- taking other evaluation metrics as the optimal objective for MERT, such as NIST score, RIBES score (Isozaki et al., 2010a).

Acknowledgments

We thank Yusuke Miyao and Naoaki Okazaki for their invaluable help and the anonymous reviewers for their comments and suggestions.

References

Bob Carpenter. 1992. *The Logic of Typed Feature Structures*. Cambridge University Press.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of ACL*, pages 263–270, Ann Arbor, MI.

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Lingustics*, 33(2):201–228.

Yuan Ding and Martha Palmer. 2005. Machine translation using probabilistic synchronous dependency insertion grammers. In *Proceedings of ACL*, pages 541–548, Ann Arbor.

Kevin Duh, Katsuhito Sudoh, Xianchao Wu, Hajime Tsukada, and Masaaki Nagata. 2011. Generalized minimum bayes risk system combination. In *Proceedings of IJCNLP*, pages 1356–1360, November.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *Proceedings of HLT-NAACL*.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of COLING-ACL*, pages 961–968, Sydney.

Isao Goto, Bin Lu, Ka Po Chow, Eiichiro Sumita, and Benjamin K. Tsou. 2011. Overview of the patent machine translation task at the ntcir-9 workshop. In *Proceedings of NTCIR-9*, pages 559–578.

Liang Huang and David Chiang. 2005. Better k-best parsing. In *Proceedings of IWPT*.

Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proceedings of 7th AMTA*.

Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. 2010a. Automatic evaluation of translation quality for distant language pairs. In *Proc.of EMNLP*, pages 944–952.

Hideki Isozaki, Katsuhito Sudoh, Hajime Tsukada, and Kevin Duh. 2010b. Head finalization: A simple reordering rule for sov languages. In *Proceedings of WMT-MetricsMATR*, pages 244–251, July.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the ACL 2007 Demo and Poster Sessions*, pages 177–180.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment templates for statistical machine transaltion. In *Proceedings of COLING-ACL*, pages 609–616, Sydney, Australia.

Yang Liu, Yajuan Lü, and Qun Liu. 2009a. Improving tree-to-tree translation with packed forests. In *Proceedings of ACL-IJCNLP*, pages 558–566, August.

Yang Liu, Haitao Mi, Yang Feng, and Qun Liu. 2009b. Joint decoding with multiple translation models. In *Proceedings of ACL-IJCNLP*, pages 576–584, August.

Haitao Mi and Liang Huang. 2008. Forest-based translation rule extraction. In *Proceedings of EMNLP*, pages 206–214, October.

Haitao Mi and Qun Liu. 2010. Constituency to dependency translation with forests. In *Proceedings of ACL*, pages 1433–1442, July.

Haitao Mi, Liang Huang, and Qun Liu. 2008. Forest-based translation. In *Proceedings of ACL-08:HLT*, pages 192–199, Columbus, Ohio.

Yusuke Miyao, Takashi Ninomiya, and Jun'ichi Tsujii. 2003. Probabilistic modeling of argument structures including non-local dependencies. In *Proceedings of RANLP*, pages 285–291, Borovets.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of ACL*, pages 160–167.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL*, pages 311–318.

Adam Pauls and Dan Klein. 2011. Faster and smaller n-gram language models. In *Proceedings of ACL-HLT*, pages 258–267, June.

Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal smt. In *Proceedings of ACL*, pages 271–279.

Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of ACL-08:HLT*, pages 577–585.

Andreas Stolcke. 2002. Srilm-an extensible language modeling toolkit. In *Proceedings of International Conference on Spoken Language Processing*, pages 901–904.

Katsuhito Sudoh, Kevin Duh, Hajime Tsukada, Masaaki Nagata, Xianchao Wu, Takuya Matsuzaki, and Jun'ichi Tsujii. 2011. Ntt-ut statistical machine translation in ntcir-9 patentmt. In *Proceedings of NTCIR-9 Workshop Meeting*, pages 585–592, December.

Xianchao Wu, Takuya Matsuzaki, and Jun'ichi Tsujii. 2010. Fine-grained tree-to-string translation rule extraction. In *Proceedings of ACL*, pages 325–334, July.

Xianchao Wu, Takuya Matsuzaki, and Jun'ichi Tsujii. 2011a. Effective use of function words for rule generalization in forest-based translation. In *Proceedings of ACL-HLT*, pages 22–31, June.

Xianchao Wu, Takuya Matsuzaki, and Jun'ichi Tsujii. 2011b. Smt systems in the university of tokyo for ntcir-9 patentmt. In *Proceedings of NTCIR-9 Workshop Meeting*, pages 666–672, December.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.

Deyi Xiong, Qun Liu, and Shouxun Lin. 2006. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of COLING-ACL*, pages 521–528, July.

Hui Zhang, Min Zhang, Haizhou Li, Aiti Aw, and Chew Lim Tan. 2009. Forest-based tree sequence to string translation model. In *Proceedings of ACL-IJCNLP*, pages 172–180, Suntec, Singapore, August.