Preference Grammars and Soft Syntactic Constraints for GHKM Syntax-based Statistical Machine Translation

Matthias Huck and **Hieu Hoang** and **Philipp Koehn**
School of Informatics
University of Edinburgh
10 Crichton Street
Edinburgh EH8 9AB, UK
{mhuck,hhoang,pkoehn}@inf.ed.ac.uk

Abstract

In this work, we investigate the effectiveness of two techniques for a feature-based integration of syntactic information into GHKM string-to-tree statistical machine translation (Galley et al., 2004): (1.) *Preference grammars* on the target language side promote syntactic well-formedness during decoding while also allowing for derivations that are not linguistically motivated (as in hierarchical translation). (2.) *Soft syntactic constraints* augment the system with additional source-side syntax features while not modifying the set of string-to-tree translation rules or the baseline feature scores.

We conduct experiments with a state-of-the-art setup on an English→German translation task. Our results suggest that preference grammars for GHKM translation are inferior to the plain target-syntactified model, whereas the enhancement with soft source syntactic constraints provides consistent gains. By employing soft source syntactic constraints with sparse features, we are able to achieve improvements of up to 0.7 points BLEU and 1.0 points TER.

1 Introduction

Previous research in both formally syntax-based (i.e., hierarchical) and linguistically syntax-based statistical machine translation has demonstrated that significant quality gains can be achieved via integration of syntactic information as features in a non-obtrusive manner, rather than as hard constraints.

We implemented two feature-based extensions for a GHKM-style string-to-tree translation system (Galley et al., 2004):

- Preference grammars to soften the hard target-side syntactic constraints that are imposed by the target non-terminal labels.
- Soft source-side syntactic constraints that enhance the string-to-tree translation model with input tree features based on source syntax labels.

The empirical results on an English→German translation task are twofold. Target-side preference grammars do not show an improvement over the string-to-tree baseline with syntactified translation rules. Source-side syntactic constraints, on the other hand, yield consistent moderate gains if applied as supplementary features in the string-to-tree setup.

2 Outline

The paper is structured as follows: First we give an overview of important related publications (Section 3). In Section 4, we review the fundamentals of syntax-based translation in general, and in particular those of GHKM string-to-tree translation.

We present preference grammars for GHKM translation in Section 5. Our technique for applying soft source syntactic constraints in GHKM string-to-tree translation is described in Section 6.

Section 7 contains the empirical part of the paper. We first describe our experimental setup (7.1), followed by a presentation and discussion of the translation results (7.2). We conclude the paper in Section 8.

3 Related Work

Our syntactic translation model conforms to the GHKM syntax approach as proposed by **G**alley, **H**opkins, **K**night, and **M**arcu (Galley et al., 2004) with composed rules as in (Galley et al., 2006) and (DeNeefe et al., 2007). Systems based on this paradigm have recently been among the top-ranked submissions to public evaluation campaigns (Williams et al., 2014; Bojar et al., 2014).

Our soft source syntactic constraints features borrow ideas from Marton and Resnik (2008) who proposed a comparable approach for hierarchical machine translation. The major difference is that the features of Marton and Resnik (2008) are only based on the labels from the input trees as seen in tuning and decoding. They penalize violations of constituent boundaries but do not employ syntactic parse annotation of the source side of the training data. We, in contrast, equip the rules with latent source label properties, allowing for features that can check for conformance of input tree labels and source labels that have been seen in training.

Other groups have applied similar techniques to a string-to-dependency system (Huang et al., 2013) and—like in our work—a GHKM string-to-tree system (Zhang et al., 2011). Both Huang et al. (2013) and Zhang et al. (2011) store source labels as additional information with the rules. They however investigate somewhat different feature functions than we do.

Marton and Resnik (2008) evaluated their method on the NIST Chinese→English and Arabic→English tasks. Huang et al. (2013) and Zhang et al. (2011) present results on the NIST Chinese→English task. We focus our attention on a very different task: English→German.

4 Syntax-based Translation

In syntax-based translation, a probabilistic synchronous context-free grammar (SCFG) is induced from bilingual training corpora. The parallel training data is word-aligned and annotated with syntactic parses on either target side (string-to-tree), source side (tree-to-string), or both (tree-to-tree). A syntactic rule extraction procedure extracts rules which are consistent with the word-alignment and comply with certain syntactic validity constraints.

Extracted rules are of the form $A, B \rightarrow \langle \alpha, \beta, ^{\sim} \rangle$. The right-hand side of the rule $\langle \alpha, \beta \rangle$ is a bilingual phrase pair that may contain non-terminal symbols, i.e. $\alpha \in (V_F \cup N_F)^+$ and $\beta \in (V_E \cup N_E)^+$, where V_F and V_E denote the source and target terminal vocabulary, and N_F and N_E denote the source and target non-terminal vocabulary, respectively. The non-terminals on the source side and on the target side of rules are linked in a one-to-one correspondence. The $^{\sim}$ relation defines this one-to-one correspondence. The left-hand side of the rule is a pair of source and target non-terminals, $A \in N_F$ and $B \in N_E$.

Decoding is typically carried out with a parsing-based algorithm, in our case a customized version of CYK+ (Chappelier and Rajman, 1998). The parsing algorithm is extended to handle translation candidates and to incorporate language model scores via cube pruning (Chiang, 2007).

4.1 GHKM String-to-Tree Translation

In GHKM string-to-tree translation (Galley et al., 2004; Galley et al., 2006; DeNeefe et al., 2007), rules are extracted from training instances which consist of a source sentence, a target sentence along with its constituent parse tree, and a word alignment matrix. This tuple is interpreted as a directed graph (the *alignment graph*), with edges pointing away from the root of the tree, and word alignment links being edges as well. A set of nodes (the *frontier set*) is determined that contains only nodes with non-overlapping closure of their spans.[1] By computing *frontier graph fragments*—fragments of the alignment graph such that their root and all sinks are in the frontier set—the GHKM extractor is able to induce a minimal set of rules which explain the training instance. The internal tree structure can be discarded to obtain flat SCFG rules. Minimal rules can be assembled to build larger *composed rules*.

Non-terminals on target sides of string-to-tree rules are syntactified. The target non-terminal vocabulary of the SCFG contains the set of labels of the frontier nodes, which is in turn a subset of (or equal to) the set of constituent labels in the parse tree. The target non-terminal vocabulary furthermore contains an initial non-terminal symbol Q. Source sides of the rules are not decorated with syntactic annotation. The source non-terminal vocabulary contains a single generic non-terminal symbol X.

In addition to the extracted grammar, the translation system makes use of a special *glue grammar* with an *initial rule*, *glue rules*, a *final rule*, and *top rules*. The glue rules provide a fall back method to just monotonically concatenate partial derivations during decoding. As we add tokens which

[1]The *span* of a node in the alignment graph is defined as the set of source-side words that are reachable from this node. The *closure* of a span is the smallest interval of source sentence positions that covers the span.

mark the sentence start ("<s>") and the sentence end ("</s>"), the rules in the glue grammar are of the following form:

Initial rule:

$$X, Q \rightarrow \langle \text{<s>}\, X^{\sim 0}, \text{<s>}\, Q^{\sim 0} \rangle$$

Glue rules:

$$X, Q \rightarrow \langle X^{\sim 0} X^{\sim 1}, Q^{\sim 0} B^{\sim 1} \rangle$$

for all $B \in N_E$

Final rule:

$$X, Q \rightarrow \langle X^{\sim 0}\ \text{</s>}, Q^{\sim 0}\ \text{</s>} \rangle$$

Top rules:

$$X, Q \rightarrow \langle \text{<s>}\, X^{\sim 0}\ \text{</s>}, \text{<s>}\, B^{\sim 0}\ \text{</s>} \rangle$$

for all $B \in N_E$

5 Preference Grammars

Preference grammars store a set of *implicit* label vectors as additional information with each SCFG rule, along with their relative frequencies given the rule. Venugopal et al. (2009) have introduced this technique for hierarchical phrase-based translation. The implicit label set refines the label set of the underlying synchronous context-free grammar.

We apply this idea to GHKM translation by not decorating the target-side non-terminals of the extracted GHKM rules with syntactic labels, but with a single generic label. The (explicit) target non-terminal vocabulary N_E thus also contains only the generic non-terminal symbol X, just like the source non-terminal vocabulary N_F. The extraction method remains syntax-directed and is still guided by the syntactic annotation over the target side of the data, but the syntactic labels are stripped off from the SCFG rules. Rules which differ only with respect to their non-terminal labels are collapsed to a single entry in the rule table, and their rule counts are pooled. However, the syntactic label vectors that have been seen with this rule during extraction are stored as implicit label vectors of the rule.

5.1 Feature Computation

Two features are added to the log-linear model combination in order to rate the syntactic well-formedness of derivations. The first feature is similar to the one suggested by Venugopal et al. (2009) and computes a score based on the relative frequencies of implicit label vectors of those rules which are involved in the derivation. The second feature is a simple binary feature which supplements the first one by penalizing a rule application if none of the implicit label vectors match.

We will now formally specify the first feature.[2] We give a recursive definition of the feature score $h_{\text{syn}}(d)$ for a derivation d.

Let r be the top rule in derivation d, with n right-hand side non-terminals. Let d_j denote the sub-derivation of d at the j-th right-hand side non-terminal of r, $1 \leq j \leq n$. $h_{\text{syn}}(d)$ is recursively defined as

$$h_{\text{syn}}(d) = \hat{t}_{\text{syn}}(d) + \sum_{j=1}^{n} h_{\text{syn}}(d_j)\,. \qquad (1)$$

In this equation, $\hat{t}_{\text{syn}}(d)$ is a simple auxiliary function:

$$\hat{t}_{\text{syn}}(d) = \begin{cases} \log t_{\text{syn}}(d) & \text{if } t_{\text{syn}}(d) \neq 0 \\ 0 & \text{otherwise} \end{cases} \qquad (2)$$

Denoting with S the implicit label set of the preference grammar, we define $t_{\text{syn}}(d)$ as a function that assesses the degree of agreement of the preferences of the current rule with the sub-derivations:

$$t_{\text{syn}}(d) = \sum_{s \in S^{n+1}} \left(p(s|r) \cdot \prod_{k=2}^{n+1} \hat{t}_h(s[k]|d_{k-1}) \right) \qquad (3)$$

We use the notation $[\cdot]$ to address the elements of a vector. The first element of an $n+1$-dimensional vector s of implicit labels is an implicit label binding of the left-hand side non-terminal of the rule r. $p(s|r)$ is the preference distribution of the rule.

Here, $\hat{t}_h(Y|d)$ is another auxiliary function that renormalizes the values of $t_h(Y|d)$:

$$\hat{t}_h(Y|d) = \frac{t_h(Y|d)}{\sum_{Y' \in S} t_h(Y'|d)} \qquad (4)$$

It provides us with a probability that the derivation d has the implicit label $Y \in S$ as its root. Finally, the function $t_h(Y|d)$ is defined as

$$t_h(Y|d) = \sum_{s \in S^{n+1}: s[1]=Y} \left(p(s|r) \cdot \prod_{k=2}^{n+1} p_h(s[k]|d_{k-1}) \right)\,. \qquad (5)$$

Note that the denominator in Equation (4) thus equals $t_{\text{syn}}(d)$.

[2]Our notational conventions roughly follow the ones by Stein et al. (2010).

This concludes the formal specification of the first features. The second feature $h_{\text{auxSyn}}(d)$ penalizes rule applications in cases where $t_{\text{syn}}(d)$ evaluates to 0:

$$h_{\text{auxSyn}}(d) = \begin{cases} 0 & \text{if } t_{\text{syn}}(d) \neq 0 \\ 1 & \text{otherwise} \end{cases} \tag{6}$$

Its intuition is that rule applications that do not contribute to $h_{\text{syn}}(d)$ should be punished. Derivations with $t_{\text{syn}}(d) = 0$ could alternatively be dropped completely, but our approach is to avoid hard constraints. We will later demonstrate empirically that discarding such derivations harms translation quality.

6 Soft Source Syntactic Constraints

Similar to the implicit target-side label vectors which we store in preference grammars, we can likewise memorize sets of source-side syntactic label vectors with GHKM rules. In contrast to preference grammars, the rule inventory of the string-to-tree system remains untouched. The target non-terminals of the SCFG stay syntactified, and the source non-terminal vocabulary is not extended beyond the single generic non-terminal.

Source-side syntactic labels are an additional latent property of the rules. We obtain this property by parsing the source side of the training data and collecting the source labels that cover the source-side span of non-terminals during GHKM rule extraction. As the source-side span is frequently not covered by a constituent in the syntactic parse tree, we employ the composite symbols as suggested by Zollmann and Venugopal (2006) for the SAMT system.[3] In cases where a span is still not covered by a symbol, we nevertheless memorize a source-side syntactic label vector but indicate the failure for the uncovered non-terminal with a special label. The set of source label vectors that are seen with a rule during extraction is stored with it in the rule table as an additional property. This information can be used to implement feature-based soft source syntactic constraints.

Table 1 shows an example of a set of source label vectors stored with a grammar rule. The first element of each vector is an implicit source-syntactic label for the left-hand side non-terminal of the rule, the remaining elements are implicit source-syntactic labels for the right-hand side source non-terminals.

[3]Specifically, we apply `relax-parse --SAMT 2` as implemented in the Moses toolkit (Koehn et al., 2007).

source label vector	frequency
(*IN+NP*,*NN*,*NN*)	7
(*IN+NP*,*NNP*,*NNP*)	3
(*IN++NP*,*NNS*,*NNS*)	2
(*IN+NP*,*NP*,*NP*)	2
(*PP//SBAR*,*NP*,*NP*)	1

Table 1: The set of source label vectors (along with their frequencies in the training data) for the rule $X, PP\text{-}MO \rightarrow \langle \text{between } X^{\sim 1} \text{ and } X^{\sim 0}, \text{zwischen } NN^{\sim 0} \text{ und } NN^{\sim 1} \rangle$. The overall rule frequency is 15.

The basic idea for soft source syntactic constraints features is to also parse the input data in a preprocessing step and try to match input labels and source label vectors that are associated with SCFG rules.

6.1 Feature Computation

Upon application of an SCFG rule, each of the non-terminals of the rule covers a distinct span of the input sentence. An *input label* from the input parse may be available for this span. We say that a *non-terminal has a match* in a given source label vector of the rule if its label in the vector is the same as a corresponding input label over the span.

We define three simple features to score matches and mismatches of the impicit source syntactic labels with the labels from the input data:

- A binary feature that fires if a rule is applied which possesses a source syntactic label vector that fully matches the input labels. This feature rewards exact source label matches of complete rules, i.e., the existance of a vector in which all non-terminals of the rule have matches.
- A binary feature that fires if a rule is applied which does not possess any source syntactic label vector with a match of the label for the left-hand side non-terminal. This feature penalizes left-hand side mismatches.
- A count feature that for each rule application adds a cost equal to the number of right-hand side non-terminals that do not have a match with a corresponding input label in any of the source syntactic label vectors. This feature penalizes right-hand side mismatches.

The second and third feature are less strict than the first one and give the system a more detailed clue about the magnitude of mismatch.

6.2 Sparse Features

We can optionally add a larger number of sparse features that depend on the identity of the source-side syntactic label:

- Sparse features which fire if a specific input label is matched. We say that *the input label is matched* in case the corresponding non-terminal that covers the span has a match in any of the source syntactic label vectors of the applied rule. We distinguish input label matches via left-hand side and via right-hand side non-terminals.

- Sparse features which fire if the span of a specific input label is covered by a non-terminal of an applied rule, but the input label is not matched.

The first set of sparse features rewards matches, the second set of sparse features penalizes mismatches.

All sparse features have individual scaling factors in the log-linear model combination. We however implemented a means of restricting the number of sparse features by providing a *core set* of source labels. If such a core set is specified, then only those sparse features are active that depend on the identity of labels within this set. All sparse features for source labels outside of the core set are inactive.

7 Experiments

We empirically evaluate the effectiveness of preference grammars and soft source syntactic constraints for GHKM translation on the English$\rightarrow$German language pair using the standard newstest sets of the Workshop on Statistical Machine Translation (WMT) for testing.[4] The experiments are conducted with the open-source *Moses* implementations of GHKM rule extraction (Williams and Koehn, 2012) and decoding with CYK+ parsing and cube pruning (Hoang et al., 2009).

[4]`http://www.statmt.org/wmt14/translation-task.html`

7.1 Experimental Setup

We work with an English–German parallel training corpus of around 4.5 M sentence pairs (after corpus cleaning). The parallel data originates from three different sources which have been eligible for the constrained track of the ACL 2014 Ninth Workshop on Statistical Machine Translation shared translation task: Europarl (Koehn, 2005), News Commentary, and the Common Crawl corpus as provided on the WMT website. Word alignments are created by aligning the data in both directions with MGIZA++ (Gao and Vogel, 2008) and symmetrizing the two trained alignments (Och and Ney, 2003; Koehn et al., 2003). The German target side training data is parsed with BitPar (Schmid, 2004). We remove grammatical case and function information from the annotation obtained with BitPar and apply right binarization of the German parse trees prior to rule extraction (Wang et al., 2007; Wang et al., 2010; Nadejde et al., 2013). For the soft source syntactic constraints, we parse the English source side of the parallel data with the English Berkeley Parser (Petrov et al., 2006) and produce composite SAMT-style labels as discussed in Section 6.

When extracting syntactic rules, we impose several restrictions for composed rules, in particular a maximum number of 100 tree nodes per rule, a maximum depth of seven, and a maximum size of seven. We discard rules with non-terminals on their right-hand side if they are singletons in the training data.

For efficiency reasons, we also enforce a limit on the number of label vectors that are stored as additional properties. Label vectors are only stored if they occur at least as often as the 50th most frequent label vector of the given rule. This limit is applied separately for both source-side label vectors (which are used by the soft syntactic contraints) and target-side label vectors (which are used by the preference grammar).

Only the 200 best translation options per distinct rule source side with respect to the weighted rule-level model scores are loaded by the decoder. Search is carried out with a maximum chart span of 25, a rule limit of 500, a stack limit of 200, and a *k*-best limit of 1000 for cube pruning.

A standard set of models is used in the baseline, comprising rule translation probabilities and lexical translation probabilities in both directions, word penalty and rule penalty, an *n*-gram language

system	dev		newstest2013		newstest2014	
	BLEU	TER	BLEU	TER	BLEU	TER
GHKM string-to-tree baseline	34.7	47.3	20.0	63.3	19.4	65.6
+ soft source syntactic constraints	35.1	47.0	20.3	62.7	19.7	64.9
+ sparse features	35.8	46.5	20.3	62.8	19.6	65.1
+ sparse features (core = non-composite)	35.4	46.8	20.2	62.9	19.6	65.1
+ sparse features (core = dev-min-occ100)	35.6	46.7	20.2	62.9	19.6	65.2
+ sparse features (core = dev-min-occ1000)	35.4	46.9	20.3	62.8	19.6	65.2
+ hard source syntactic constraints	34.6	47.4	19.9	63.4	19.4	65.6
string-to-string (GHKM syntax-directed rule extraction)	33.8	48.0	19.3	63.8	18.7	66.2
+ preference grammar	33.9	47.7	19.3	63.7	18.8	66.0
+ soft source syntactic constraints	34.6	47.0	19.8	62.9	19.5	65.2
+ drop derivations with $t_{\text{syn}}(d) = 0$	34.0	47.5	19.7	63.0	18.8	65.8

Table 2: English→German experimental results (truecase). BLEU scores are given in percentage. A selection of 2000 sentences from the newstest2008-2012 sets is used as development set.

model, a rule rareness penalty, and the monolingual PCFG probability of the tree fragment from which the rule was extracted (Williams et al., 2014). Rule translation probabilities are smoothed via Good-Turing smoothing.

The language model (LM) is a large interpolated 5-gram LM with modified Kneser-Ney smoothing (Kneser and Ney, 1995; Chen and Goodman, 1998). The target side of the parallel corpus and the monolingual German News Crawl corpora are employed as training data. We use the SRILM toolkit (Stolcke, 2002) to train the LM and rely on KenLM (Heafield, 2011) for language model scoring during decoding.

Model weights are optimized to maximize BLEU (Papineni et al., 2002) with batch MIRA (Cherry and Foster, 2012) on 1000-best lists. We selected 2000 sentences from the newstest2008-2012 sets as a development set. The selected sentences obtained high sentence-level BLEU scores when being translated with a baseline phrase-based system, and do each contain less than 30 words for more rapid tuning. newstest2013 and newstest2014 are used as unseen test sets. Translation quality is measured in truecase with BLEU and TER (Snover et al., 2006).[5]

7.2 Translation Results

The results of the empirical evaluation are given in Table 2. Our GHKM string-to-tree system attains state-of-the-art performance on newstest2013 and newstest2014.

[5]TER scores are computed with `tercom` version 0.7.25 and parameters `-N -s`.

7.2.1 Soft Source Syntactic Constraints

Adding the three dense soft source syntactic constraints features from Section 6.1 improves the baseline scores by 0.3 points BLEU and 0.6 points TER on newstest2013 and by 0.3 points BLEU and 0.7 points TER on newstest2014.

Somewhat surprisingly, the sparse features from Section 6.2 do not boost translation quality further on any of the two test sets. We observe a considerable improvement on the development set, but it does not carry over to the test sets. We attributed this to an overfitting effect. Our source-side soft syntactic label set of composite SAMT-style labels comprises 8504 different labels that appear on the source-side of the parallel training data. Four times the amount of sparse features are possible (left-hand side/right-hand side matches and mismatches for each label), though not all of them fire on the development set. 3989 sparse weights are tuned to non-zero values in the experiment. Due to the sparse nature of the features, overfitting cannot be ruled out.

We attempted to take measures in order to avoid overfitting by specifying a core set of source labels and deactivating all sparse features for source labels outside of the core set (cf. Section 6.2). First we specified the core label set as all non-composite labels. Non-composite labels are the plain constituent labels as given by the syntactic parser. Complex SAMT-style labels are not included. The size of this set is 71 (non-composite labels that have been observed during rule extraction). Translation performance on the development set drops in the *sparse features (core = non-*

system (tuned on newstest2012)	**newstest2012**		**newstest2013**		**newstest2014**	
	BLEU	TER	BLEU	TER	BLEU	TER
GHKM string-to-tree baseline	17.9	65.7	19.9	63.2	19.4	65.3
+ soft source syntactic constraints	18.2	65.3	20.3	62.6	19.7	64.7
+ sparse features	18.6	64.9	20.4	62.5	19.8	64.7
+ sparse features (core = non-composite)	18.4	65.1	20.3	62.7	19.8	64.7
+ sparse features (core = dev-min-occ100)	18.4	64.8	20.6	62.2	19.9	64.4

Table 3: English→German experimental results (truecase). BLEU scores are given in percentage. newstest2012 is used as development set.

composite) setup, but performance does not increase on the test sets.

Next we specified the core label set in another way: We counted how often each source label occurs in the input data on the development set. We then applied a minimum occurrence count threshold and added labels to the core set if they did not appear more rarely than the threshold. We tried values of 100 and 1000 for the minimum occurrence, resulting in 277 and 37 labels being in the core label set, respectively. Neither the *sparse features (core = dev-min-occ100)* experiment nor the *sparse features (core = dev-min-occ1000)* experiment yields better translation quality than what we see in the setup without sparse features.

We eventually conjectured that the choice of our development set might be a reason for the ineffectiveness of the sparse features, as on a fine-grained level it could possibly be too different from the test sets with respect to its syntactic properties. We therefore repeated some of the experiments with scaling factors optimized on newstest2012 (Table 3). The *sparse features (core = dev-min-occ100)* setup indeed performs better when tuned on newstest2012, with improvements of 0.7 points BLEU and 1.0 points TER on newstest2013 and of 0.5 points BLEU and 0.9 points TER on newstest2014 over the baseline tuned on the same set.

Finally, we were interested in demonstrating that soft source syntactic constraints are superior to hard source syntactic constraints. We built a setup that forces the decoder to match source-side syntactic label vectors in the rules with input labels.[6] Hard source syntactic constraints are indeed worse than soft source syntactic constraints (by 0.4 BLEU on newstest2013 and 0.3 BLEU on newstest2014). The setup with hard source syntactic constraints performs almost exactly at the level of the baseline.

[6]Glue rules are an exception. They do not need to match the input labels.

7.2.2 Preference Grammar

In the series of experiments with a preference grammar, we first evaluated a setup with the underlying SCFG of the preference grammar system, but without preference grammar. We denote this setup as *string-to-string (GHKM syntax-directed rule extraction)* in Table 2. The extraction method for this string-to-string system is GHKM syntax-directed with right-binarized syntactic target-side parses from BitPar, as in the string-to-tree setup. The constituent labels from the syntactic parses are however not used to decorate non-terminals. The grammar contains rules with a single generic non-terminal instead of syntactic ones. The *string-to-string (GHKM syntax-directed rule extraction)* setup is on newstest2013 0.7 BLEU (0.5 TER) worse and on newstest2014 0.7 BLEU (0.6 TER) worse than the standard GHKM string-to-tree baseline.

We then activated the preference grammar as described in Section 5. GHKM translation with a preference grammar instead of a syntactified target non-terminal vocabulary in the SCFG is considerably worse than the standard GHKM string-to-tree baseline and barely improves over the string-to-string setup.

We added soft source syntactic constraints on top of the preference grammar system, thus combining the two techniques. Soft source syntactic constraints give a nice gain over the preference grammar system, but the best setup without a preference grammar is not outperformed. In another experiment, we investigated the effect of dropping derivations with $t_{\text{syn}}(d) = 0$ (cf. Section 5.1). Note that the second feature $h_{\text{auxSyn}}(d)$ is not useful in this setup, as the system is forced to discard all derivations that would be penalized by that feature. We deactivated $h_{\text{auxSyn}}(d)$ for the experiment. The hard decision of dropping derivations with $t_{\text{syn}}(d) = 0$ leads to a performance loss of

0.1 BLEU on newstest2013 and a more severe deterioration of 0.7 BLEU on newstest2014.

8 Conclusions

We investigated two soft syntactic extensions for GHKM translation: Target-side preference grammars and soft source syntactic constraints.

Soft source syntactic constraints proved to be suitable for advancing the translation quality over a strong string-to-tree baseline. Sparse features are beneficial beyond just three dense features, but they require the utilization of an appropriate development set. We also showed that the soft integration of source syntactic constraints is crucial: Hard constraints do not yield gains over the baseline.

Preference grammars did not perform well in our experiments, suggesting that translation models with syntactic target non-terminal vocabularies are a better choice when building string-to-tree systems.

Acknowledgements

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreements n^{o} 287658 (EU-BRIDGE) and n^{o} 288487 (MosesCore).

References

Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 Workshop on Statistical Machine Translation. In *Proc. of the Workshop on Statistical Machine Translation (WMT)*, pages 12–58, Baltimore, MD, USA, June.

Jean-Cédric Chappelier and Martin Rajman. 1998. A Generalized CYK Algorithm for Parsing Stochastic CFG. In *Proc. of the First Workshop on Tabulation in Parsing and Deduction*, pages 133–137, Paris, France, April.

Stanley F. Chen and Joshua Goodman. 1998. An Empirical Study of Smoothing Techniques for Language Modeling. Technical Report TR-10-98, Computer Science Group, Harvard University, Cambridge, MA, USA, August.

Colin Cherry and George Foster. 2012. Batch Tuning Strategies for Statistical Machine Translation. In *Proc. of the Human Language Technology Conf. / North American Chapter of the Assoc. for Computational Linguistics (HLT-NAACL)*, pages 427–436, Montréal, Canada, June.

David Chiang. 2007. Hierarchical Phrase-Based Translation. *Computational Linguistics*, 33(2):201–228, June.

Steve DeNeefe, Kevin Knight, Wei Wang, and Daniel Marcu. 2007. What Can Syntax-Based MT Learn from Phrase-Based MT? In *Proc. of the 2007 Joint Conf. on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 755–763, Prague, Czech Republic, June.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *Proc. of the Human Language Technology Conf. / North American Chapter of the Assoc. for Computational Linguistics (HLT-NAACL)*, pages 273–280, Boston, MA, USA, May.

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 *Proc. of the 21st Int. Conf. on Computational Linguistics and 44th Annual Meeting of the Assoc. for Computational Linguistics*, pages 961–968, Sydney, Australia, July.

Qin Gao and Stephan Vogel. 2008. Parallel Implementations of Word Alignment Tool. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, SETQA-NLP '08, pages 49–57, Columbus, OH, USA, June.

Kenneth Heafield. 2011. KenLM: Faster and Smaller Language Model Queries. In *Proc. of the Workshop on Statistical Machine Translation (WMT)*, pages 187–197, Edinburgh, Scotland, UK, July.

Hieu Hoang, Philipp Koehn, and Adam Lopez. 2009. A Unified Framework for Phrase-Based, Hierarchical, and Syntax-Based Statistical Machine Translation. In *Proc. of the Int. Workshop on Spoken Language Translation (IWSLT)*, pages 152–159, Tokyo, Japan, December.

Zhongqiang Huang, Jacob Devlin, and Rabih Zbib. 2013. Factored Soft Source Syntactic Constraints for Hierarchical Machine Translation. In *Proc. of the Conf. on Empirical Methods for Natural Language Processing (EMNLP)*, pages 556–566, Seattle, WA, USA, October.

Reinhard Kneser and Hermann Ney. 1995. Improved Backing-Off for M-gram Language Modeling. In *Proceedings of the Int. Conf. on Acoustics, Speech, and Signal Processing*, volume 1, pages 181–184, Detroit, MI, USA, May.

Philipp Koehn, Franz Joseph Och, and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In *Proc. of the Human Language Technology Conf. / North American Chapter of the Assoc. for Computational Linguistics (HLT-NAACL)*, pages 127–133, Edmonton, Canada, May/June.

P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proc. of the Annual Meeting of the Assoc. for Computational Linguistics (ACL)*, pages 177–180, Prague, Czech Republic, June.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proc. of the MT Summit X*, Phuket, Thailand, September.

Yuval Marton and Philip Resnik. 2008. Soft Syntactic Constraints for Hierarchical Phrased-Based Translation. In *Proc. of the Annual Meeting of the Assoc. for Computational Linguistics (ACL)*, pages 1003–1011, Columbus, OH, USA, June.

Maria Nadejde, Philip Williams, and Philipp Koehn. 2013. Edinburgh's Syntax-Based Machine Translation Systems. In *Proc. of the Workshop on Statistical Machine Translation (WMT)*, pages 170–176, Sofia, Bulgaria, August.

Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51, March.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proc. of the Annual Meeting of the Assoc. for Computational Linguistics (ACL)*, pages 311–318, Philadelphia, PA, USA, July.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning Accurate, Compact, and Interpretable Tree Annotation. In *Proc. of the 21st Int. Conf. on Computational Linguistics and 44th Annual Meeting of the Assoc. for Computational Linguistics*, pages 433–440, Sydney, Australia, July.

Helmut Schmid. 2004. Efficient Parsing of Highly Ambiguous Context-Free Grammars with Bit Vectors. In *Proc. of the Int. Conf. on Computational Linguistics (COLING)*, Geneva, Switzerland, August.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proc. of the Conf. of the Assoc. for Machine Translation in the Americas (AMTA)*, pages 223–231, Cambridge, MA, USA, August.

Daniel Stein, Stephan Peitz, David Vilar, and Hermann Ney. 2010. A Cocktail of Deep Syntactic Features for Hierarchical Machine Translation. In *Proc. of the Conf. of the Assoc. for Machine Translation in the Americas (AMTA)*, Denver, CO, USA, October/November.

Andreas Stolcke. 2002. SRILM – an Extensible Language Modeling Toolkit. In *Proc. of the Int. Conf. on Spoken Language Processing (ICSLP)*, volume 3, Denver, CO, USA, September.

Ashish Venugopal, Andreas Zollmann, Noah A. Smith, and Stephan Vogel. 2009. Preference Grammars: Softening Syntactic Constraints to Improve Statistical Machine Translation. In *Proc. of the Human Language Technology Conf. / North American Chapter of the Assoc. for Computational Linguistics (HLT-NAACL)*, pages 236–244, Boulder, CO, USA, June.

Wei Wang, Kevin Knight, and Daniel Marcu. 2007. Binarizing Syntax Trees to Improve Syntax-Based Machine Translation Accuracy. In *Proc. of the 2007 Joint Conf. on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 746–754, Prague, Czech Republic, June.

Wei Wang, Jonathan May, Kevin Knight, and Daniel Marcu. 2010. Re-structuring, Re-labeling, and Re-aligning for Syntax-based Machine Translation. *Computational Linguistics*, 36(2):247–277, June.

Philip Williams and Philipp Koehn. 2012. GHKM Rule Extraction and Scope-3 Parsing in Moses. In *Proc. of the Workshop on Statistical Machine Translation (WMT)*, pages 388–394, Montréal, Canada, June.

Philip Williams, Rico Sennrich, Maria Nadejde, Matthias Huck, Eva Hasler, and Philipp Koehn. 2014. Edinburgh's Syntax-Based Systems at WMT 2014. In *Proc. of the Workshop on Statistical Machine Translation (WMT)*, pages 207–214, Baltimore, MD, USA, June.

Jiajun Zhang, Feifei Zhai, and Chengqing Zong. 2011. Augmenting String-to-Tree Translation Models with Fuzzy Use of Source-side Syntax. In *Proc. of the Conf. on Empirical Methods for Natural Language Processing (EMNLP)*, pages 204–215, Edinburgh, Scotland, UK, July.

Andreas Zollmann and Ashish Venugopal. 2006. Syntax Augmented Machine Translation via Chart Parsing. In *Proc. of the Workshop on Statistical Machine Translation (WMT)*, pages 138–141, New York City, NY, USA, June.