F-STRUCTURE TRANSFER-BASED STATISTICAL MACHINE TRANSLATION

Yvette Graham Josef van Genabith Anton Bryl
Dublin City University

Proceedings of the LFG09 Conference

Miriam Butt and Tracy Holloway King (Editors)

2009

CSLI Publications

http://csli-publications.stanford.edu/

Abstract

In this paper, we describe a statistical deep syntactic transfer decoder that is trained fully automatically on parsed bilingual corpora. Deep syntactic transfer rules are induced automatically from the f-structures of a LFG parsed bitext corpus by automatically aligning local f-structures, and inducing all rules consistent with the node alignment. The transfer decoder outputs the n-best TL f-structures given a SL f-structure as input by applying large numbers of transfer rules and searching for the best output using a log-linear model to combine feature scores. The decoder includes a fully integrated dependency-based tri-gram language model. We include an experimental evaluation of the decoder using different parsing disambiguation resources for the German data to provide a comparison of how the system performs with different German training and test parses.

1 Introduction

In this paper, we describe a statistical deep syntactic transfer decoder used as the transfer component of a Transfer-Based Machine Translation (TBMT) system to transfer source language (SL) deep structures to the target language (TL). Deep syntactic transfer rules are induced automatically from the functional structures of a Lexical Functional Grammar (LFG) (Kaplan and Bresnan, 1982; Bresnan, 2001; Dalrymple, 2001) parsed bitext corpus. Firstly, local f-structures are automatically aligned, before all rules consistent with the node alignment are induced automatically. The transfer decoder applies large numbers of transfer rules to the input SL f-structure and searches for the best TL output f-structure using a log-linear model to combine feature scores.

The paper is structured as follows: in Section 1, we give our motivation for using deep syntax in MT, Section 2 describes the architecture of deep syntactic Transfer-Based MT, Section 3 describes the main focus of this paper, statistical transfer between source and target deep syntactic structures, in Section 4, we give an experimental evaluation of the transfer decoder in the context of a hybrid system that uses LFG functional structures (f-structures) as the intermediate representation for transfer, training and testing the system using two different disambiguation models for the German data for German to English translation, and Section 5 gives our plans for future work.

2 Motivation

In TBMT, among the different types of intermediate structures used for transfer are deep syntactic structures. For example, Bojar and Hajič (2008) use the Functional Generative Description (FGD) (Sgall et al., 1986) Tectogrammatical Layer (T-layer), labeled ordered dependency trees, while Riezler and Maxwell (2006) use the LFG f-structure, an attribute-value structure encoding of bilexical labeled dependencies.

Deep syntactic structures are more language independent than other representations used for MT such as surface form strings and phrase-structure trees, and therefore should provide a better means of forming generalizations about how to translate from one language to another. For example, automatic translation between very distant language pairs can require complex re-ordering of words between source and target. For many languages incorrect word order in TL output results in one of two problems; the output is either (i) ungrammatical or (ii) grammatical with incorrect meaning. Since the permitted word order of a sentence in many languages is strongly influenced by the dependency relations between the words of the sentence, explicitely including these relations in the translation model should help produce correct TL word order, especially when translating between very distant language pairs. In addition, using a language specific generator designed to generate from structures in which these relations between words are explicitely represented could also help to produce better quality output with respect to word order.

As well as dependency relations, many theories of deep syntax also include morphological analysis, so that words in the surface form are represented in the deep syntactic structure in lemma form with a set of features encoding grammatical information, like case, person, number, tense, etc. Explicitly representing this grammatical information may be important for translation from morphologically poor languages into morphologically richer ones. For example, when translating from English into German *the red wine* has at least three possible translations: *der rote Wein*, *den roten Wein* and *dem roten Wein*. In this example, the value of the feature *case* in the TL needs to be known in order to choose the correct morphological inflection of the determiner *der* and adjective *rot*. If the case of the noun in the English phrase is established this information should help select the best phrase in German. Including this grammatical information present in the source and target deep syntactic structure should therefore help produce the correct morphology in the TL.

3 Deep Syntactic Transfer-Based MT

Deep Syntactic Transfer-Based MT is composed of three parts; (i) parsing to deep syntactic structure, (ii) transfer from SL deep structure to TL deep structure and (iii) generation of TL sentence (Figure 1). Each stage in the three stage pipeline architecture could be carried out using fully automatically learned (statistical) resources, hand-crafted resources or a hybrid of statistical and hand-crafted resources. For example, for parsing Riezler and Maxwell (2006) use hand-crafted grammars in addition to automatically learned disambiguation models. The parsing step in their system is therefore a hybrid of hand-crafted and statistical methods. For transfer, they use mostly automatically induced transfer rules as well as some hand-crafted rules. In addition, they carry out hand-

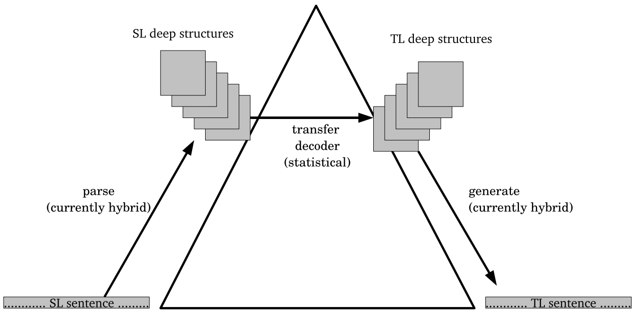


Figure 1: Deep Syntax Transfer-Based MT Pipeline Architecture

selected corrections of the word alignment prior to rule induction.[1] They also use a statistical search and statistical model to transfer SL structures to TL structures. The transfer component of their system therefore is also a hybrid. Finally, for generation, they used a hand-crafted generation grammar and a statistical model, including a TL model, for example, to select the best output. Thus the generation step in their system is also a hybrid of hand-crafted and automatically learned resources.

The focus of our work is to investigate methods of automatically learning how to translate from training data. The transfer step in our system is trained fully automatically without any hand-crafted rules or human-selected corrections to any part of the rules or word-alignment.[2] Our system uses hand-crafted resources for parsing and generation (Kaplan et al., 2004; Riezler et al., 2002). The bitext training data is automatically parsed (Kaplan et al., 2002) and the same type of grammar is used for generation. The transfer stage of our system is fully statistical, but the experimental evaluation in this paper is evaluating the decoder in the context of a hybrid system, using hand-crafted resources for parsing and generation.[3] Figure 1 shows the Transfer-Based MT system pipeline with each stage labeled either statistical or hybrid for our system.

4 Statistical Transfer

4.1 Transfer Rule Induction

To induce transfer rules automatically from the parsed corpus, we use the RIA rule induction tool (Graham and van Genabith, 2009). Figure 2 shows some

[1] Through personal communication with John Maxwell.

[2] Note that results for our system should not be compared with results reported in Riezler and Maxwell (2006) since our transfer component is statistical while that of Riezler and Maxwell (2006) is a hybrid.

[3] There are parsing and generation resources available for LFG that are trained fully automatically (Cahill et al., 2004; Cahill and van Genabith, 2006). We plan to use these resources with our statistical transfer decoder to compare with the current hybrid system in the near future.

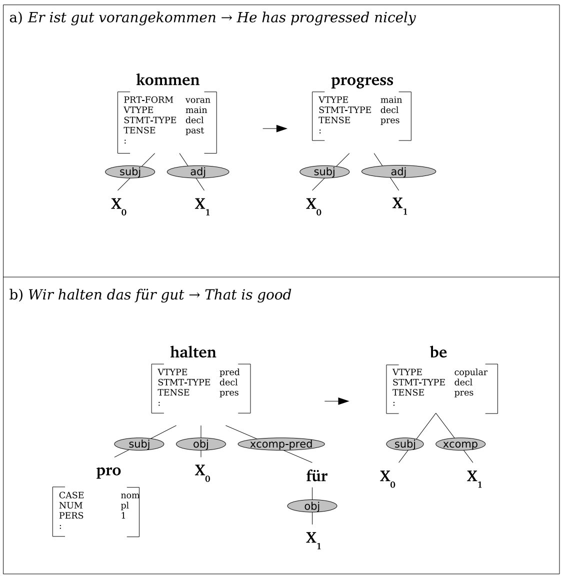


Figure 2: Example Transfer Rules

example transfer rules produced by the tool. The transfer rule induction algorithm takes as input (i) a dependency structure pair and (ii) a one-to-one set of alignments between nodes of the dependency structure pair.

4.1.1 Local F-Structure Alignment

Prior to rule induction a set of one-to-one correspondences between the local f-structures of each pair of parsed sentences in the bilingual corpus must be established. For automatic alignment of local f-structures we take the parsed bilingual corpus and extract the predicate values from each pair of f-structures to reconstruct a lemmatized version of the bitext. Figure 3 shows an example of a bitext corpus that is first parsed, then reconstructed from the f-structure representation. The order of the predicates in the reconstructed version of the bitext (Figure 3(c)) is determined by the location of the local f-structure within the overall f-structure. The predicate values are ordered via a depth-first traversal of the underlying dependency graph encoded in the f-structure. For example, the order of the predicates in the reconstructed corpus (Figure 3(c)) of the German f-structure in Figure 3(b) is *ähneln und bill bob* since *ähneln* is the predicate of the main f-structure with daughter *und* that in turn has daughters *bill* and *bob*. In order for the depth-first traversal not to loop if the f-structure contains instances of reentrancy or argument sharing we temporarily ignore these dependencies when reconstructing the corpus from the f-structures. The reconstructed bitext is then input to Giza++ (Och et al., 1999) and automatic word alignment is run in both language directions. The output is then input to Moses to compute the symmetrization of the bidirectional alignment. We currently use the intersection in order to get a reliable set of one-to-one correspondences between words.

The aligned parsed bitext is used as input to the rule induction step. We use the RIA open source rule induction tool (Graham and van Genabith, 2009) to induce transfer rules. For each input f-structure pair and its node alignment, RIA induces all transfer rules consistent with the node alignment. The following section provides the definition for consistent transfer rules.

4.1.2 Consistent Transfer Rules

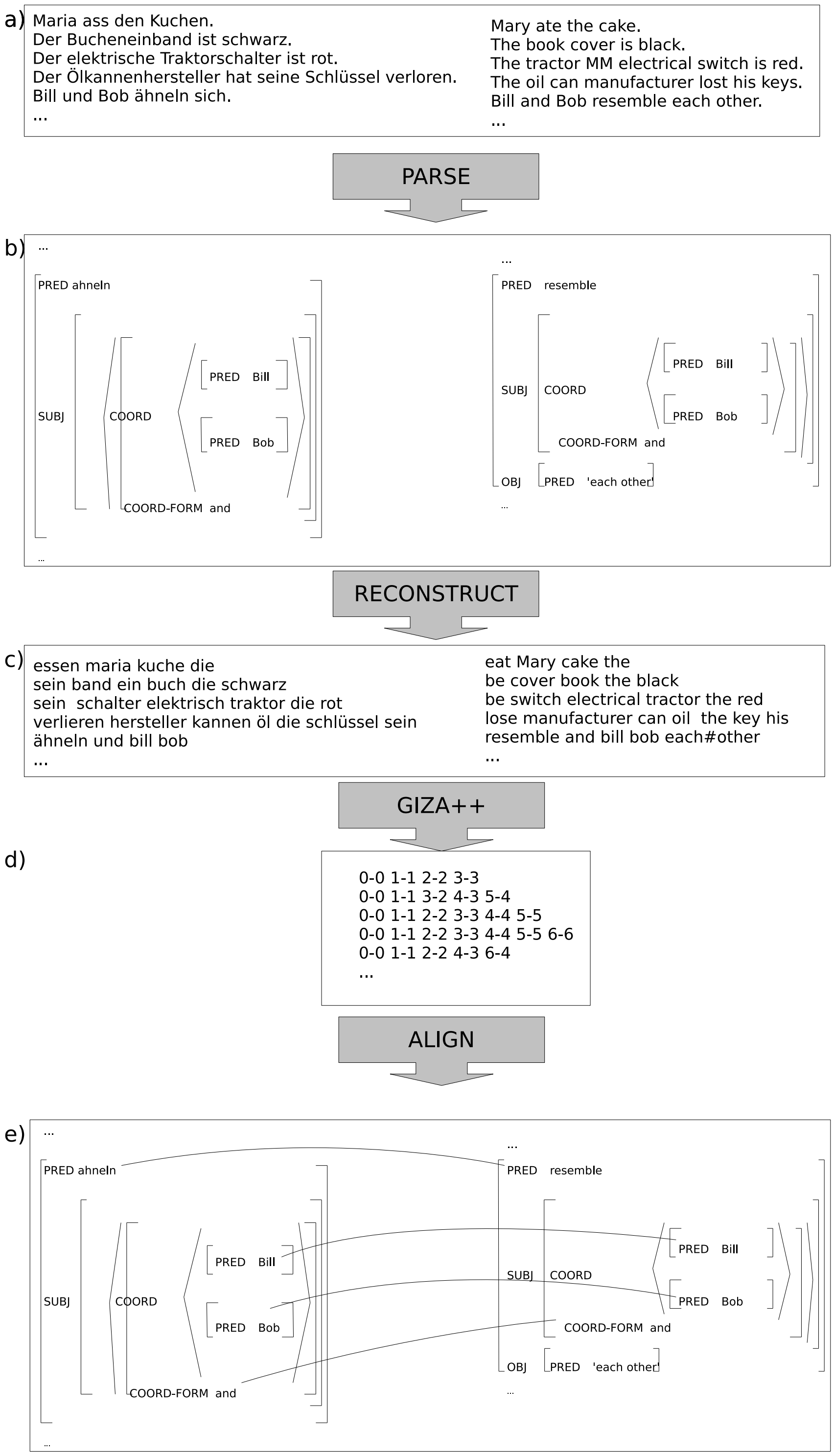
As in Phrase-Based Statistical Machine Translation (PB-SMT), where a word alignment for each example sentence pair is first established before phrases consistent with that word alignment are extracted (Och et al., 1999; Koehn et al., 2003), we induce transfer rules that are consistent with the node alignment. We define a consistent transfer rule using a simplification of the actual training dependency structures and temporarily consider them as acyclic graph structures by ignoring edges that cause cycles in the graph or edges that share an end node with another edge. Definition 1 applied to a (simplified) dependency structure pair yields a set of rules containing no variables by constraining rule induction using both the alignments between nodes and the position of the nodes within the two structures:

Definition 1.
Given a one-to-one set of alignments A between nodes in dependency pair (F, E), $(\overline{f}, \overline{e})$ is a rule consisting of nodes (N_f, N_e), rooted at (r_f, r_e), with descendents (D_f, D_e) of r_f and r_e in F and E respectively, if

$$\begin{array}{ll} & N_f = r_f \cup D_f \\ \bigwedge & N_e = r_e \cup D_e \\ \bigwedge & \forall f_i \in N_f : (f_i, e_j) \in A \rightarrow e_j \in N_e \\ \bigwedge & \forall e_j \in N_e : (f_i, e_j) \in A \rightarrow f_i \in N_f \\ \bigwedge & \exists e_j \in N_e : (r_f, e_j) \in A \\ \bigwedge & \exists f_i \in N_f : (f_i, r_e) \in A \end{array}$$

Definition 2.
For any rule $(\overline{f}, \overline{e})$ in dependency pair (F, E) rooted at (r_f, r_e) consisting of nodes N_f and N_e, where $(\overline{s}, \overline{t})$ is also a rule in (F, E) rooted at (r_s, r_t) consisting of nodes N_s and N_t where $r_s \neq r_f$, $r_t \neq r_e$, iff $r_s \in N_f$ and $r_t \in N_e$, there is a rule $(\overline{a}, \overline{b})$ rooted at (r_f, r_e) with nodes r_s and r_t replaced by variable x_k, where k is an index unique to the transfer rule, consisting of nodes:

$$\begin{array}{l} N_a : N_f \backslash N_s \cup x_k \\ N_b : N_e \backslash N_t \cup x_k \end{array}$$



Maria ass den Kuchen.
Der Bucheneinband ist schwarz.
Der elektrische Traktorschalter ist rot.
Der Ölkannenhersteller hat seine Schlüssel verloren.
Bill und Bob ähneln sich.
...

Mary ate the cake.
The book cover is black.
The tractor MM electrical switch is red.
The oil can manufacturer lost his keys.
Bill and Bob resemble each other.
...

essen maria kuche die
sein band ein buch die schwarz
sein schalter elektrisch traktor die rot
verlieren hersteller kannen öl die schlüssel sein
ähneln und bill bob
...

eat Mary cake the
be cover book the black
be switch electrical tractor the red
lose manufacturer can oil the key his
resemble and bill bob each#other
...

0-0 1-1 2-2 3-3
0-0 1-1 3-2 4-3 5-4
0-0 1-1 2-2 3-3 4-4 5-5
0-0 1-1 2-2 3-3 4-4 5-5 6-6
0-0 1-1 2-2 4-3 6-4
...

Figure 3: Alignment of Local F-structures

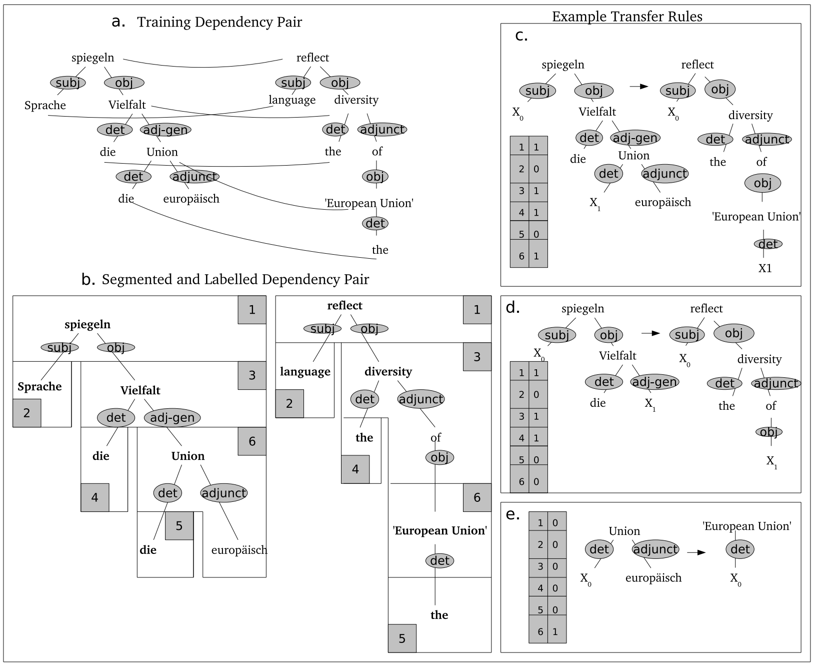


Figure 4: Consistent Transfer Rules

Definition 2 allows the introduction of variables into transfer rules. Any rule that contains another rule nested within it can be used to form a new rule by replacing the nested rule with a single variable in its LHS and RHS. To help visualize what is considered a consistent transfer rule, Figure 4(b) shows the example dependency structure in Figure 4(a) divided into parts by a number of boxes with corresponding parts of the dependency structure pair labeled with the numbers 1-6. Each consistent transfer rule can be realised by assigning a binary value to each pair of boxes, so that boxes assigned 1 are included in the rule and boxes assigned 0 are left out. Combinations of binary values for nodes are constrained and this can be visualized by only allowing adjoining boxes in Figure 4(b) to be labeled 1 for any rule. Figures 4(c), 4(d) and 4(e) show example consistent rules with the binary value combinations that encode them.

4.2 Translation Model

As in PB-SMT, a Transfer-Based SMT translation model can be defined as a combination of several feature functions combined using a log-linear model:

$$p(e|f) = exp \sum_{i=1}^{n} \lambda_i h_i(e,f)$$

4.2.1 Transfer Rule Probabilities

In PB-SMT the translation of an input sentence into an output sentence is modeled by breaking down the translation of the sentence into the translation of a set of phrases. Similarly, for Transfer-Based SMT, the transfer of the SL structure **f** into a TL structure **e** can be broken down into the transfer of a set of rules $\{\bar{f}, \bar{e}\}$:

$$p(\bar{f}_1^I|\bar{e}_1^I) = \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i)$$

We compute all rules from the training corpus and estimate the translation probability distribution by relative frequency of the rules:

$$\phi(\bar{f}, \bar{e}) = \frac{count(\bar{e}, \bar{f})}{\sum_{\bar{f}_i} count(\bar{e}, \bar{f}_i)}$$

This is carried out in both the source-to-target and target-to-source direction and each model is used as a feature.

4.2.2 Lexical Weighting

We adapt a standard lexical-weighting method used in PB-SMT to hierarchical deep syntactic structure. In PB-SMT, lexical weighting is used as a back-off since it provides richer statistics and more reliable probability estimates. Adapting this feature to deep syntax is straightforward. In PB-SMT the lexical translation probability of a phrase pair is calculated based on the alignment between the words in the phrase pair. For deep syntax, we simply calculate the same probability via the alignment of lexical items in the LHS and RHS of a transfer rule. The lexical translation probability of a RHS, $\bar{e}$, given the LHS, $\bar{f}$, is estimated as follows:

$$lex(\bar{e}|\bar{f}, a) = \prod_{i=1}^{length(\bar{e})} \frac{1}{|\{j|(i,j) \in a\}|} \sum_{\forall(i,j)\in a} w(e_i|f_j)$$

We use lexical weighting in both language directions.

4.2.3 A Dependency-Based Language Model

The overall system employs a language model at two different stages; a trigram dependency-based language model is used as a feature in the log-linear model by the transfer decoder and a standard trigram language model is used after generation to select the single best TL output. Riezler and Maxwell (2006) used a dependency-based language model in their system, but this was only done after decoding by calculating dependency-based language model scores on the n-best output of the decoder.[4] We take an approach that is more in keeping with SMT and use language modeling during decoding. This section describes how

[4] Through personal communication with John Maxwell.

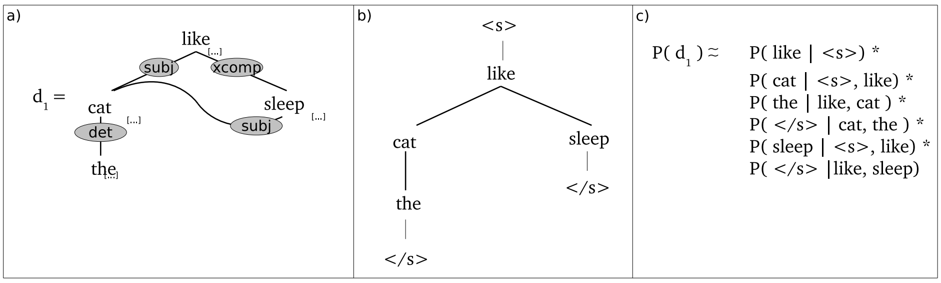


Figure 5: Dependency-Based Language Model Example for F-structure of *The cat likes to sleep*

we have fully integrated a dependency-based language model into the transfer decoder.

Since our statistical search produces dependency structures where words are organized in a graph as opposed to a standard language model that deals with linear sequences of words, we estimate the probability of a dependency structure using the preceding context of each word within the dependency graph. In a standard trigram language model, the probability of the i^{th} word in the context of its preceding i-1 words is approximated by the probability of observing it preceeded by its two preceding words:

$$P(w_1, ..., w_m) \approx \prod_{i=1}^{m} P(w_i|w_{i-2}, w_{i-1})$$

The dependency-based language model approximates the probability of each word in the structure as the probability of observing it preceeded by its parent and grandparent words:

$$P(w_1(..., w_m)) \approx \prod_{i=1}^{m} P(w_i|parent(parent(w_i)), parent(w_i))$$

If all dependency relations between local f-structures that cause either argument sharing or reentrancy are ignored, the underlying pred-only structure is an acyclic graph. We ignore such dependency relations when extracting the dependency-based language model so that each node in the structure can be assumed to have at most a single parent node. Figure 5(a) shows an example f-structure for the English sentence *The cat likes to sleep*. Figure 5(b) shows the simplified graph that used for language modeling where the reentrancy involving *sleep* and *cat* is ignored. As in standard language modeling, where the start of a sentence is represented by the special symbol $<s>$, we add a root node to the structure with this symbol. We also add the end symbol to the leaf nodes $</s>$. Figure 5(c) shows the probability approximation of the f-structure shown in Figures 5(a) and (b).

4.2.4 Other Features

Other features included in the log-linear model for ranking TL hypothesis structures include:

- Word Penalty
- Phrase Penalty
- Fragmented Structure Penalty
- Fragmented Rule Penalty
- Grammatical Mismatch Penalty

The word penalty and phrase penalty are taken almost directly from PB-SMT. The word penalty is used to counterbalance the dependency-based language model's bias for shorter TL structures and the phrase penalty is used to counterbalance the bias of transfer rule probabilities toward smaller rules. All other things being equal, it is better to transfer the structure using large transfer rules, as the chunk of structure that forms the RHS was already observed together in the corpus and therefore can be assumed to cause no problems with regard to creating unusual TL word combinations, which can happen when combining smaller rules. In addition, as the system can produce structures that are missing dependency relations between two nodes in the TL structure, the fragmented structure penalty is used to allow the model to bias towards more complete structures. A fragmented rule penalty is also used to disprefer rules that were induced from training data that had received a fragment parse from the parser. These rules tend to lead to bad TL structures that cause problems for the generator. It would be possible to completely filter out such rules to ensure they were never used, but in theory it is better to leave them in and allow the system to bias against their use as it is still possible in some cases that a fragmented rule leads to the best solution for a given input, for example when no non-fragmented rule is available to translate the word. Finally, the grammatical mismatch penalty is used to penalize rules by the amount of mismatching grammatical information in the LHS of the rule and the SL structure. All else being equal, rules that have a small amount of LHS grammatical information matching that of the SL structure are dispreferred.

4.3 Decoding

4.3.1 Top-down Transfer Rule Application

Decoding takes a single SL structure as input and involves a statistical search for the n-best TL structures. TL solutions are created via a top-down application of transfer rules to the SL structure beginning at the root (or main) f-structure. When the LHS of a rule unifies with the SL structure, the RHS produces a

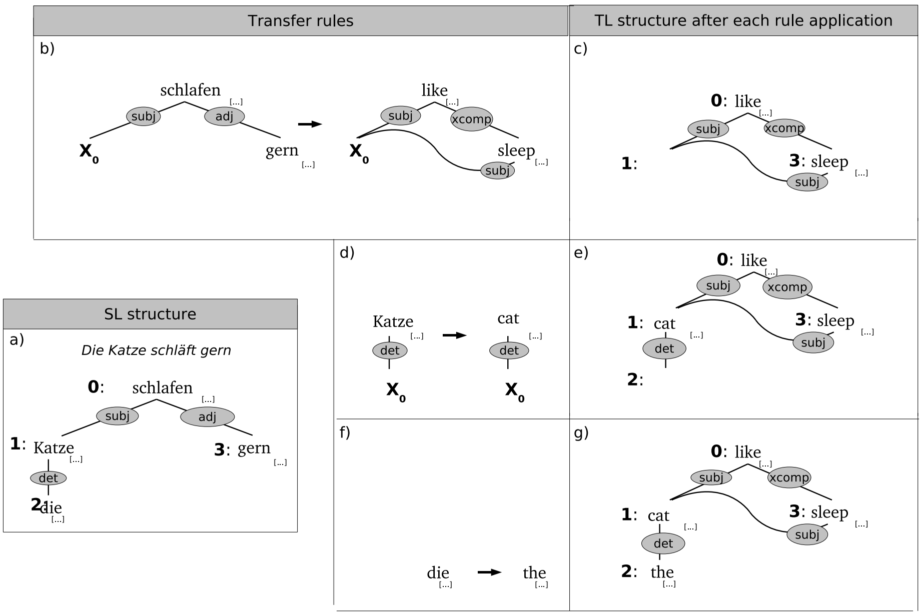


Figure 6: Example top-down application of transfer rules

portion of the TL structure. Figure 6 shows an example application of three rules to the dependency structure for the German sentence *Die Katze schläft gern* ‘The cat likes to sleep’ shown in Figure 6(a). Figure 6(b) shows the first transfer rule to be applied to the root node of the SL structure to produce the TL structure portion shown in Figure 6(c). Transfer rule variables map arguments in the SL structure to the desired position when creating a TL solution. For example, variable X_0 in Figure 6(b) maps the *subject* of *schlafen* to the *subject* of *like* in the TL structure labeled with id number *1* shown in Figure 6(c). Next *Katze* in the SL structure is translated (Figures 6(d) and 6(e)), before finally *die* is translated (Figures 6(f) and 6(g)).

4.3.2 Beam Search

As with all SMT systems, the number of possible output translations given a single SL input is too large to exhaustively rank each possible output. We therefore employ a standard search algorithm, beam search, to produce the n-best TL solutions.

Partial translations (or translation hypotheses) are constructed by applying transfer rules to the SL structure. While TL translations are constructed, beam search manages the large search space by ranking translation hypotheses and pruning the search by dropping lower scoring hypotheses. A number of stacks are used to organize translation hypotheses into groups of comparable hypotheses, according to the portion of SL structure that has already been translated to produce each hypothesis, i.e. hypothesis stack *N* stores TL translation hypotheses with *N* nodes covered in the SL structure. For example, Figure 7(a) shows the hypothesis stacks for decoding the f-structure of *Die Katze schläft gern* containing 4 nodes and therefore requiring stacks 1-4 for decoding, each stack storing translation hypotheses for solutions covering one to four nodes, respectively.

Transfer rules are indexed by root node so that they can be retrieved quickly to translate SL structure nodes. For example, in Figure 7(a) the rules rooted at node *Katze* are stored together. Since rules are applied top-down to the SL structure (see Section 4.3.1) rules beginning at the root node of the SL structure (or main SL f-structure) are first used to construct hypotheses. For example, in Figure 7(b) the rule that translates the root node of the SL structure *schlafen* as *doze* is first used to construct a hypothesis and since it covers one SL node it is stored in hypothesis stack 1. Figure 7(c) shows the next three hypotheses that are constructed: *snooze*, *sleep* and *like sleep*. Hypotheses are ordered within each stack according to their score, high-to-low from bottom-to-top. We currently use histogram pruning. When a stack becomes full, lower scoring solutions are pruned by being popped off the top of the stack.

For efficiency, each partial translation is only stored once in memory even though it may be part of several different future hypotheses. For example, hypothesis stack 2 in Figure 7(d) contains four translations constructed by expanding hypothesis *doze* by four different rules, each translating the word *Katze* into a different TL word. These new hypotheses are represented by a reference to the most recently applied transfer rule (rules translating *Katze*) and a reference back to the previous hypothesis. Figure 6 shows an example of decoding. Figure 7(e) shows an example of how per single completed translation, the structure for *the lion likes to doze*, is represented in the hypothesis stacks and Figure 7(f) shows all hypotheses are represented when the decoder has completed translating a single SL input structure. The n-best translated structures can be retrieved from the final stack.

4.3.3 Efficient Dependency-Based Language Modeling

An important feature in an SMT decoder is the language model and integrating one can be a more challenging task than other features since the language model score of a translation hypothesis cannot be calculated by simply combining the language model scores of the phrases (or rules) that it is composed of.

Although the search space is limited by beam search, during decoding large numbers of TL hypothesis structures need to be ranked. At each expansion of a translation hypothesis (via joining of an existing hypothesis with a new rule) a language model score for the newly created hypothesis needs to be calculated. Since this is carried out very many times per single decoding run, it is vital that

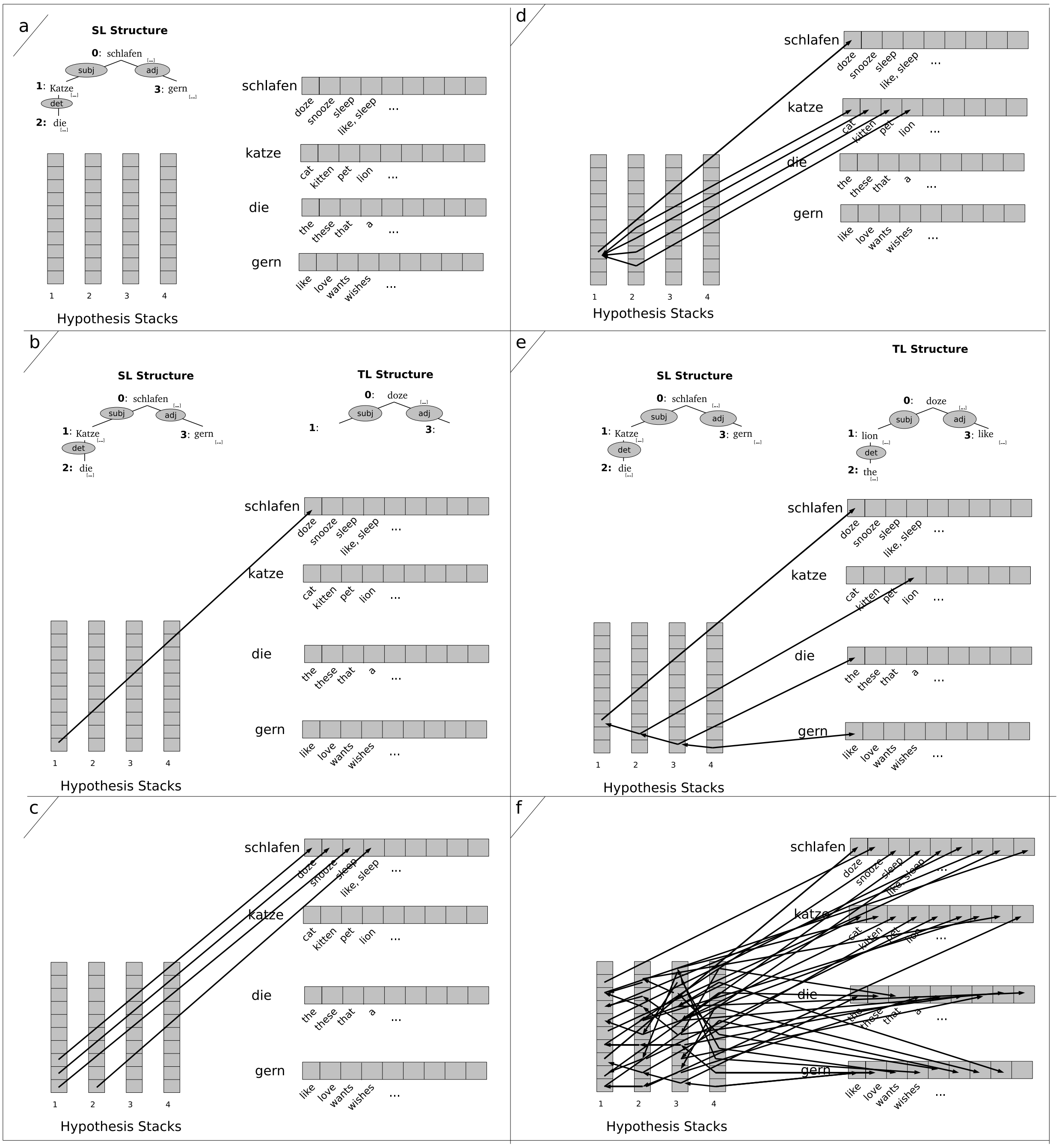


Figure 7: Beam Search Decoding

the method of calculating this score is highly efficient.

In our system, we pre-compute a dependency-based language model score for each transfer rule prior to beam search. This score is calculated only once for each rule even though a single rule may be part of several translation hypotheses. Then during decoding, when a translation hypothesis is expanded by adding a new rule, the new hypothesis score can be calculated quickly by combining the score of the old hypothesis, the rule score and a score calculated based on the probabilities of trigrams where the old hypothesis and rule join together. The probability of a TL hypothesis, h_n, that was produced by combining hypothesis h_{n-1} and rule r can be calculated as follows:

$$hyp\_score(h_n) = hyp\_score(h_{n-1}) * join\_score(h_{n-1}, r) * rule\_score(r)$$

Since $hyp\_score(h_{n-1})$ and $rule\_score(r)$ are already computed, only $join\_score(h_{n-1}, r)$ needs to be computed when $hyp\_score(h_n)$ is computed.

Figure 8 shows how the language model scores are efficiently calculated when decoding the f-structure for the German sentence *Die Werbung spiegelt die Vielfalt der britischen Universität wider* ‘The advertisement reflects the diversity of the British university’. We begin with the German f-structure graph shown in Figure 8(a) with nodes labeled by id numbers. Figure 8(b) shows the initial empty translation hypothesis that has probability 1.

Figures 8(c), 8(f) and 8(i) show example transfer rules that can be applied to the German f-structure. Dependency-based language model scores are pre-computed for each rule by identifying all trigrams within the RHS structure and calculating the product of their individual probability estimations retrieved from the language model; we will call this the rule_score (see Figure 8(d) for *Rule A*, Figure 8(g) for *Rule B* and Figure 8(j) for *Rule C*). In addition, for each rule, n-grams located at the RHS root node and frontier nodes are recorded. For example, *Rule B* in Figure 8(g) has a single root node bigram *advertisement the* located at node *2* while *Rule A* in Figure 8(d) has two frontier bigrams $< s >$, *reflect* and *diversity, of* located at nodes *2* and *6*, respectively. This information is used to calculate the language model score of joining a rule and a hypothesis.

Figure 8(e) shows the translation hypothesis established by applying *Rule A* to the German structure. The language model score for the structure is established by combining the score of the previous hypothesis (since this is the first rule for this hypothesis, the previous hypothesis is the empty hypothesis and is therefore 1), the join score (since we are joining the rule with the empty hypothesis this score is also 1) and the rule score (see Figure 8(d)).

Figure 8(h) shows the translation hypothesis created by expanding $Hypothesis_1$ by $Rule_B$. Since this expansion involved adding a rule at node *2* in the TL structure, the joining trigrams are derived by creating lists of words via all possible combinations of the frontier bigrams belonging to $Hypothesis_1$ labeled *2* and the root bigrams of $Rule_B$, also labeled *2* (see root n-grams in Figure 8(g)). For this example, this results in a single word sequence $<s>$*re-*

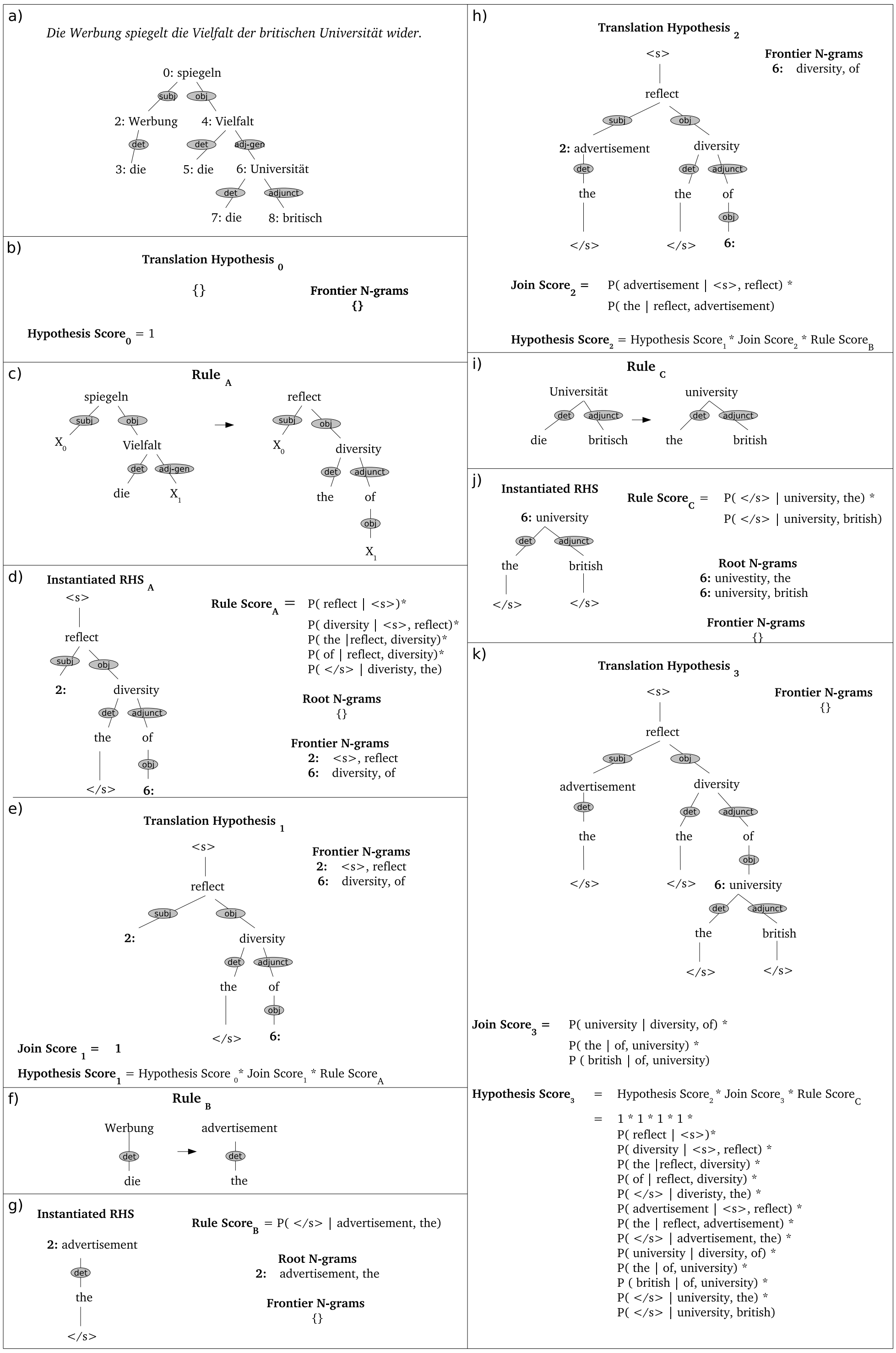


Figure 8: Efficient Dependency-based Language Modeling

flect advertisement the which forms two trigrams $<s>$*-reflect-advertisement* and *reflect-advertisement-the*. The score for $Hypothesis_2$ is then calculated by combining the hypothesis score for $Hypothesis_1$, the join score and the pre-computed rule score for *Rule B*.

5 Experimental Evaluation

In our experimental evaluation of the system, we investigate the effects of the disambiguation model used to select the best parse. Riezler and Maxwell (2006) used an English disambiguation model for parsing both the German and English data when translating from German to English. If a single disambiguation model is used for both languages, the f-structures of a given pair of training sentences are likely to be quite similar, and this may help the rule induction process. However, another approach is to use language-specific disambiguation models for parsing. In this case, it is more likely that the *actual* best f-structure for each sentence of the training data is selected. Although a more authentic German parse may help the overall MT system, at the same time this is likely to increase the dissimilarity between the parses of the German-English sentences pairs, which may increase the difficulty of transfer.

We conduct an empirical investigation into which approach achieves better machine translation output for our system, by training and testing the system using (i) an English disambiguation model (Kaplan et al., 2004; Riezler et al., 2002) to select the best parse for both German and English sentences, and compare with results when (ii) a German disambiguation model (Forst, 2007) is used for selecting the best German parse and an English disambiguation model (Kaplan et al., 2004; Riezler et al., 2002) is used to select the best parse for the English sentences.

5.1 Training

The system was trained separately for each configuration. Training data for both configurations used data restricted by sentence length of 5-15 words from the Europarl (Koehn et al., 2005) and Newswire parallel corpora, which resulted in approximately 360,000 German-English sentence pairs, and a held-out development set of 500 sentences pairs. Both sides of the training corpus were parsed with the XLE parse engine (Kaplan et al., 2002). For Configuration 1, an English disambiguation model (Kaplan et al., 2004; Riezler et al., 2002) was used when parsing both the German and English data. For Configuration 2, a German disambiguation model (Forst, 2007) was used when parsing the German data and the English disambiguation model (Kaplan et al., 2004; Riezler et al., 2002) for the English data. The single best parse for each sentence, according to the appropriate disambiguation model, was used for training for both configurations.

For node alignment, Giza++ (Och et al., 1999) was run in both language

Config.	BLEU	NIST	Coverage	Connected TL structure
1	0.1121	3.5685	92.2%	33.4%
2	0.0730	2.6643	91.8%	47.2%

Table 1: Machine Translation System Results for Configuration 1: English disambiguation model for both German and English data, and Configuration 2: German disambiguation model for German data and English disambiguation model for English data

directions and the intersection was obtained using Moses (Koehn et al., 2007). We used both a dependency-based language model from the parsed TL side of the Europarl corpus and a conventional language model using lower-cased TL sentences, both trained on approximately 1,250,000 sentences. The SRILM toolkit (Stolcke, 2002) was used for both language models. Minimum Error Rate Training (Och, 2003) was carried out using ZMERT (Zaidan, 2009) to train weights for each configuration on 500 randomly selected held-out development set sentences optimizing for Bleu.

5.2 Testing

The system was tested in a single language direction, German to English on 500 randomly selected held-out test set German sentences and the single best TL translation produced by the system was evaluated using automatic metrics with a single reference translation.

For each configuration, the German sentences were parsed with the same parsing engine and grammar as was used for training, and the single best f-structure according to the disambiguation model was selected as input to the decoder.

TL decoder output structures can be fragmented, and we automatically repair them if necessary. Automatic repair involves adding edges (in the form of FIRST/REST equations with nodes ordered via the position of their translations in the SL structure) to any TL structure that does not already form a single connected graph. For each test sentence, the 100-best TL decoder output structures were repaired automatically, before being input to the generator and a maximum of 50,000 sentences were generated per test sentence. A standard language model was used to select the final TL output.

5.3 Results

The Bleu (Papineni et al., 2002) and NIST (Doddington, 2002) scores for both system configurations on the test set are shown in Table 1. According to the automatic metrics, Configuration 1 achieves a Bleu score of 0.1121 outperforming Configuration 2, which achieves 0.073, almost 4 Bleu points lower than Configuration 1. Configuration 1 also has higher system coverage, i.e. it was able to produce at least some output for 92.2% of the test set, while Configuration 2 achieves 91.8% coverage. The number of TL structures output from the decoder that already formed a single connected graph and therefore did not require any repair was, however, higher for Configuration 2 (47.2%) than Configuration 1 (33.4%).

5.4 Discussion

The results obtained in the experimental evaluation are contrary to our initial expectations. With the current system translating from English to German, using the English disambiguation model for both languages outperforms automatic evaluation results when the system is run on parse data disambiguated by language specific models. We had expected that the more authentic parses for the German data should lead to an overall increase in translation results, even if the difficulty of transfer is increased slightly by the slight increase in non-isomorphism across the f-structure representations for the parsed sentence pairs in the training data. The transfer rule induction algorithm is designed to induce rules that capture non-isomorphism, and therefore increasing non-isomorphism should not effect the system to this degree. One suspected cause of the problems for Configuration 2 may lie in the grammar used with this disambiguation model. The number of features in the grammar is higher than that of the German grammar used with the English disambiguation model of Configuration 1. When the data is parsed this leads to the German f-structures of Configuration 2 containing far more atomic features than those of Configuration 1. In fact, for the German development set parses, the ratio of number features in the f-structures for Configuration 1 compared to Configuration 2 is approximately 1:4. We suspect that due to the higher number of features of Configuration 2, transfer rules do not generalize as well to unseen data. The SL atomic features are used in our system to guide the selection of transfer rules. The smaller set of features of Configuration 1 may be a better guide for transfer than the larger set of Configuration 2.

6 Future Work

The size of the training corpus used in the evaluation is small compared to corpora usually used for training SMT systems. We would like to perform further extrinsic evaluation of the two disambiguation models when the system is trained on a larger corpus not restricted by sentence length. This would provide each configuration with richer statistical estimates and higher coverage of transfer rules on unseen SL structures.

7 Conclusion

We presented a SMT transfer decoder that uses deep syntactic structures, as the intermediate representation for transfer that applies state-of-the-art methods of PB-SMT to deep syntactic transfer. In the experimental evaluation the decoder achieves better results using an English disambiguation model for parsing German data, than when a German disambiguation model is used.

Acknowledgements

This work was partly funded by a Science Foundation Ireland Ph.D. scholarship P07077-60101.

References

Ondřej Bojar and Jan Hajič. 2008. Phrase-Based and Deep Syntactic English-to-Czech Statistical Machine Translation. In *Proceedings of the third Workshop on Statistical Machine Translation*, Columbus, Ohio, June 2008.

Joan Bresnan. 2001. *Lexical-Functional Syntax.*, Blackwell Oxford, 2001.

Miriam Butt, Helge Dyvik, Tracy H. King, Hiroshi Masuichi and Christian Rohrer. 2002. The Parallel Grammar Project. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING'02), Workshop on Grammar Engineering and Evaluation*, pages 1-7. Tapei, ROC.

Aoife Cahill, Michael Burke, Ruth O'Donovan, Josef van Genabith and Andy Way. 2004. Long-Distance Dependency Resolution in Automatically Acquired Wide-Coverage PCFG-based LFG approximations. In *Proceedings of the 42nd ACL*.

Aoife Cahill and Josef van Genabith. 2006. Robust PCFG-Based Generation using Automatically Acquired LFG Approximations. In *Proceedings of COLING-ACL 2006*, pages 1033-1040, Sydney, Australia.

Mary Dalrymple. *Lexical Functional Grammar*, Academic Press, San Diego, CA; London. 2001.

George Doddington. 2002. Automatic Evaluation of Machine Translation Quality Using N-Gram Co-Occurrence Statistics. In *Proceedings of HLT 2002*.

Martin Forst. 2007. PhD thesis. Disambiguation for a Linguistically Precise German Parser. University of Stuttgart.

Yvette Graham, Deirdre Hogan and Josef van Genabith. 2007. Automatic Evaluation of Generation and Parsing for Machine Translation with Automatically Acquired Transfer Rules. In *Proceedings of the 2007 Workshop on Using Corpora for NLG: Language Generation and Machine Translation*, at MT Summit XI, Copenhagen, September 2007.

Yvette Graham and Josef van Genabith. 2008. Packed Rules for Automatic Transfer Rule Induction. In *Proceedings of the European Association of Machine Translation Conference 2008*, Hamburg, Germany.

Yvette Graham and Josef van Genabith. 2009. An Open Source Rule Induction Tool for Transfer-Based SMT. To appear in *The Prague Bulletin of Mathematical Linguistics*.

Ronald M. Kaplan, Stefan Riezler, Tracy H. King, John T. Maxwell, and Alexander Vasserman. 2004 Speed and Accuracy in Shallow and Deep Stochastic Parsing. In *Proceedings of Human Language Technology Conference/North American Chapter of the Association for Computational Linguistics Meeting*, Boston, MA, May 2-7 2004.

Ronald M. Kaplan, Tracy H. King and John T. Maxwell. 2002. Adapting Existing Grammars: the XLE Experience. In *Proceedings of COLING 2002*, Taipei, Taiwan.

Ronald Kaplan and Joan Bresnan. 1982. Lexical Functional Grammar, a Formal System for Grammatical Represenation. In J. Bresnan, editor, *The Mental Representation of Grammatical Relations*, pages 173-281, MIT Press, Cambridge, MA.

Philipp Koehn, Franz Josef Och and Daniel Marcu. 2003. Statistical Phrase-based Translation. In *Proceedings of the HLT-NAACL 2003*, pages 48-54, Edmonton, May/June 2003.

Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings of MT Summit 2005*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison Burch, Richard Zens, Alexandra Constantin, Marcello Federico, Nicola Bertoldi, Chris Dyer, Evan Herbst, Brooke Cowen, Wade Shen, Christine Moran and Ondřej Bojar. 2007. Moses: Open Source Toolkit for Statistical Machine Translation In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*

John T. Maxwell III and Ronald M. Kaplan. 1991. A Method for Disjunctive Constraint Satisfaction. In *Current Issues in Parsing Technology*, Masaru Tomita editor, pages 173-190, Kluwer Aca demic Publishers.

Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics* (ACL-03), Sapporo, Japan, pages 160-167.

Franz Josef Och, Christoph Tillmann Hermann and Franz Josef Ney. 2000. Improved Alignment Models for Statistical Machine Translation. In *Proceedings of the 1999 Conference on Empirical Methods in Natural Language Processing* (EMNLP'99). College Park, MD, pages 20-28.

Kishore Papineni, Salim Roukos, Todd Ward and Wei-Jing Zhu. 2002. A Method for Automatic Evaluation of Machine Translation. In *Proceedings of ACL 2002*, Philadelphia, pages 311-318.

Stefan Riezler, Tracy H. King, Ronald M. Kaplan, Richard Crouch, John T. Maxwell, and Mark Johnson. 2002. Parsing the Wall Street Journal using Lexical Functional Grammar and Discriminitive Estimation Techniques . (grammar version 2005) In *Proceedings of the 40th Annual Meeting of the Association of Computational Linguistics (ACL)*, Philadelphia, July 2002.

Stefan Riezler and John T. Maxwell III. 2006. Grammatical Machine Translation. In *Proceedings of HLT-ACL*, pages 248-255, New York.

Petr Sgall, Eva Hajičova and Jarmilla Panevova. 1986. *The Meaning of the Sentence and its Semantic and Pragmatic Aspects*. Dordrecht: Reidel and Prague: Academia 1986.

Andreas Stolcke. 2002. SRILM - An Extensible Language Modeling Toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, Denver, Colorado, September 2002.

Omar Zaidan. 2009. Z-MERT: A Fully Configurable Open Source Tool for Minimum Error Rate Training of Machine Translation Systems. In *The Prague Bulletin of Mathematical Linguistics*, No. 91:79:88.