

Generation in Machine Translation from Deep Syntactic Trees

Keith Hall

Center for Language and Speech Processing
Johns Hopkins University
Baltimore, MD 21218
keith_hall@jhu.edu

Petr Němec

Institute of Formal and Applied Linguistics
Charles University
Prague, Czech Republic
nemec@ufal.mff.cuni.cz

Abstract

In this paper we explore a generative model for recovering surface syntax and strings from deep-syntactic tree structures. Deep analysis has been proposed for a number of language and speech processing tasks, such as machine translation and paraphrasing of speech transcripts. In an effort to validate one such formalism of deep syntax, the Praguian Tectogrammatical Representation (TR), we present a model of synthesis for English which generates surface-syntactic trees as well as strings. We propose a generative model for function word insertion (prepositions, definite/indefinite articles, etc.) and subphrase reordering. We show by way of empirical results that this model is effective in constructing acceptable English sentences given impoverished trees.

1 Introduction

Syntactic models for language are being reintroduced into language and speech processing systems thanks to the success of sophisticated statistical models of parsing (Charniak and Johnson, 2005; Collins, 2003). Representing deep syntactic relationships is an open area of research; examples of such models are exhibited in a variety of grammatical formalisms, such as Lexical Functional Grammars (Bresnan and Kaplan, 1982), Head-driven Phrase Structure Grammars (Pollard and Sag, 1994)

and the Tectogrammatical Representation (TR) of the Functional Generative Description (Sgall et al., 1986). In this paper we do not attempt to analyze the differences of these formalisms; instead, we show how one particular formalism is sufficient for automatic analysis and synthesis. Specifically, in this paper we provide evidence that TR is sufficient for synthesis in English.

Augmenting models of machine translation (MT) with syntactic features is one of the main fronts of the MT research community. The Hiero model has been the most successful to date by incorporating syntactic structure amounting to simple tree structures (Chiang, 2005). Synchronous parsing models have been explored with moderate success (Wu, 1997; Quirk et al., 2005). An extension to this work is the exploration of deeper syntactic models, such as TR. However, a better understanding of the synthesis of surface structure from the deep syntax is necessary.

This paper presents a generative model for surface syntax and strings of English given tectogrammatical trees. Sentence generation begins by inserting auxiliary words associated with autosemantic nodes; these include prepositions, subordinating conjunctions, modal verbs, and articles. Following this, the linear order of nodes is modeled by a similar generative process. These two models are combined in order to synthesize a sentence.

The Amalgam system provides a similar model for generation from a *logical form* (Corston-Oliver et al., 2002). The primary difference between our approach and that of the Amalgam system is that we focus on an impoverished deep structure (akin to

logical form); we restrict the deep analysis to contain only the features which transfer directly across languages; specifically, those that transfer directly in our Czech-English machine translation system. Amalgam targets different issues. For example, Amalgam’s generation of prepositions and subordinating conjunctions is severely restricted as most of these are considered part of the logical form.

The work of Langkilde-Geary (2002) on the Halogen system is similar to the work we present here. The differences that distinguish their work from ours stem from the type of deep representation from which strings are generated. Although their syntactic and semantic representations appear similar to the Tectogrammatical Representation, more explicit information is preserved in their representation. For example, the Halogen representation includes markings for determiners, voice, subject position, and dative position which simplifies the generation process. We believe their *minimally specified* results are based on input which most closely resembles the input from which we generate in our experiments.

Amalgam’s reordering model is similar to the one presented here; their model reorders constituents in a similar way that we reorder subtrees. Both the model of Amalgam and that presented here differ considerably from the n -gram models of Langkilde and Knight (1998), the TAG models of Bangalore and Rambow (2000), and the *stochastic generation from semantic representation* approach of Soricut and Marcu (2006). In our work, we order the local-subtrees¹ of an augmented deep-structure tree based on the syntactic features of the nodes in the tree. By factoring these decisions to be independent for each local-subtree, the set of strings we consider is only constrained by the projective structure of the input tree and the local permutation limit described below.

In the following sections we first provide a brief description of the Tectogrammatical Representation as used in our work. Both manually annotated and synthetic TR trees are utilized in our experiments; we present a description of each type of tree as well as the motivation for using it. We then describe the generative statistical process used to model the synthesis of analytical (surface-syntactic) trees based

¹A local subtree consists of a parent node (governor) and it’s immediate children.

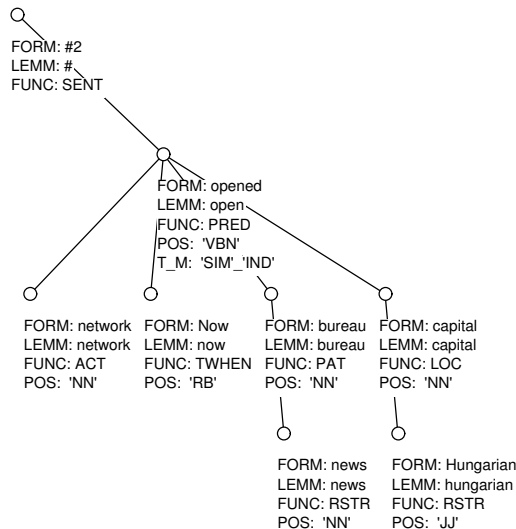


Figure 1: Example of a manually annotated, Synthetic TR tree (see Section 2.2).

Reference: **Now the network has opened a news bureau in the Hungarian capital**

Each sentence has an artificial root node labeled #. Verbs contain their tense and mood (labeled T_M).

on the TR trees. Details of the model’s features are presented in the following section. Finally we present empirical results for experiments using both the manually annotated and automatically generated data.

2 Tectogrammatical (Deep) Syntax

The Tectogrammatical Representation (TR) comes out of the Praguian linguistic theory known as the Functional Generative Description of language (Sgall et al., 1986). TR attempts to capture deep syntactic relationships based on the valency of predicates (i.e., function-argument structure) and modification of participants (i.e., nouns used as actors, patients, etc.). A key feature of TR is that dependency relationships are represented only for autosemantic words (content words), meaning that synsemantic words (syntactic function words) are encoded as features of the grammatical relationships rather than the actual words. Abstracting away from specific syntactic lexical items allows for the representation to be less language-specific making the representation attractive as a medium for machine translation and summarization.

Figure 1 shows an example TR tree, the nodes of

which represent the autosemantic words of the sentence. Each node is labeled with a morphologically reduced word-form called the lemma and a functor that describes the deep syntactic relationship to its governor (function-argument form). Additionally, the nodes are labeled with grammatemes that capture morphological and semantic information associated with the autosemantic words. For example, English verb forms are represented by the infinitive form as the lemma and the grammatemes encode the tense, aspect, and mood of the verb. For a detailed description of the TR annotation scheme see Böhmová et al. (2002). In Figure 1 we show only those features that are present in the TR structures used throughout this paper.

Both the synsemantic nodes and the left-to-right surface order² in the TR trees is under-specified. In the context of machine translation, we assume the TR word order carries no information with the exception of a single situation: the order of coordinated phrases is preserved in one of our models.

2.1 Analytic Representation

While it is not part of the formal TR description, the authors of the TR annotation scheme have found it useful to define an intermediate representation between the sentence and the TR tree (Böhmová et al., 2002). The analytical representation (AR) is a surface-syntactic dependency tree that encodes syntactic relationships between words (i.e., object, subject, attribute, etc.). Unlike the TR layer, the analytical layer contains all words of the sentence and their relative ordering is identical to the surface order.

2.2 Manually Annotated TR

In order to evaluate the efficacy of the generation model, we construct a dataset from both manually annotated data and automatically generated data. The information contained in the originally manually annotated TR all but specifies the surface form. We have modified the annotated data by removing all features except those that could be directly transferred across languages. Specifically, we preserve the following features: lemma, functor, verbal gram-

²In a TR tree, a subtree is always between the nodes to the left and right of its governor. More specifically, all TR trees are projective. For this reason, the relative ordering of subtrees imposes an absolute ordering for the tree.

matemes, and part-of-speech tags. The **lemma** is the morphologically reduced form of the word; for verbs this is the infinitive form and for nouns this is the singular form. The **functor** is the deep-syntactic function of the node; for example, the deep functor indicates whether a node is a predicate, an actor, or a patient. Modifiers can be labeled as locative, temporal, benefactive, etc. Additionally we include a **verbal grammateme** which encodes tense and mood as well as a Penn Treebank style part-of-speech tag.

3 Generative Process

In this section we describe the generative process that inserts the synsemantic auxiliary words, reorders the trees, and produces a sentence. Our evaluation will be on English data, so we describe the models and the model features in the context of English. While the model is language independent, the specific features and the size of the necessary conditioning contexts is a function of the language.

Given a TR tree T , we wish to predict the correct auxiliary nodes A and an ordering of the words associated with $\{T \cup A\}$, defined by the function $f(\{T \cup A\})$. The functions f determine the surface word order of the words associated with nodes of the auxiliary-inserted TR tree: $N = \{T \cup A\}$. The node features that we use from the nodes in the TR and AR trees are: the word lemma, the part-of-speech (POS) tag, and the functor.³ The objective of our model is:

$$\begin{aligned} & \arg \max_{A, f} P(A, f|T) \\ & = \arg \max_{A, f} P(f|A, T)P(A|T) \end{aligned} \quad (1)$$

$$\approx \arg \max_f P(f|T, \arg \max_A P(A|T)) \quad (2)$$

In Equation 2 we approximate the full model with a greedy procedure. First, we predict the most likely A according to the model $P(A|T)$. Given A , we compute the best ordering of the nodes of the tree, including those introduced in A .

There is an efficient dynamic-programming solution to the objective function in Equation 1; how-

³The type of functor used (deep syntactic or surface-syntactic) depends on the tree to which we are applying the model. One form of the reordering model operates on AR trees and therefore uses surface syntactic functors. The other model is based on TR trees and uses deep-syntactic functors.

ever, in this work we experiment with the greedy approximation.

3.1 Insertion Model

The specific English auxiliary nodes which are not present in TR include articles, prepositions, subordinating conjunctions, and modal verbs.⁴ For each node in the TR tree, the generative process predicts which synsemantic word, if any, should be inserted as a dependent of the current node. We make the assumption that these decisions are determined independently.

Let $T = \{w_1, \dots, w_i, \dots, w_k\}$ be the nodes of the TR tree. For each node w_i , we define the associated node a_i to be the auxiliary node that should be inserted as a dependent of w_i . Given a tree T , we wish to find the set of auxiliary nodes $A = \{a_1, \dots, a_k\}$ that should be inserted⁵:

$$P(A|T) = \prod_i P(a_i|a_1, \dots, a_{i-1}, T) \quad (3)$$

$$\approx \prod_i P(a_i|T) \quad (4)$$

$$\approx \prod_i P(a_i|w_i, w_{g(i)}) \quad (5)$$

Equation 3 is simply a factorization of the original model, Equation 4 shows the independence assumption, and in Equation 5 we make an additional conditional independence assumption that in order to predict auxiliary a_i , we need only know the associated node w_i and its governor $w_{g(i)}$.⁶

We further divide the model into three components: one that models articles, such as the English articles *the* and *a*; one that models prepositions and subordinating conjunctions; and one that models modal verbs. The first two models are of the form described by Equation 5. The modal verb insertion model is a deterministic mapping based on

⁴The function of synsemantic nodes are encoded by functors. For example, the prepositions *to*, *at*, *in*, *by*, and *on* may be used to indicate time or location. An autosemantic modifier will be labeled as temporal or locative, but the particular preposition is not specified.

⁵Note that we include the auxiliary node labeled NOAUX to be inserted, which in fact means a node is not inserted.

⁶In the case of nodes whose governor is a coordinating conjunction, the governor information comes from the governor of the coordination node.

grammatemes expressing the verb modality of the main verb. Additionally, each model is independent of the other and therefore up to two insertions per TR node are possible (an article and another syntactic modifier). In a variant of our model, we perform a small set of deterministic transformations in cases where the classifier is relatively uncertain about the predicted insertion node (i.e., the entropy of the conditional distribution is high).

We note here that unlike the Amalgam system (Corston-Oliver et al., 2002), we do not address features which are determined (or almost completely determined) by the underlying deep-structure. For example, the task of inserting prepositions is non-trivial given we only know a node’s functor (e.g., the node’s valency role).

3.2 Analytical Representation Tree Generation

We have experimented with two paradigms for synthesizing sentences from TR trees. The first technique involves first generating AR trees (surface syntax). In this model, we predict the node insertions, transform the functors from TR to AR functions (deep valency relationship to surface-syntactic relationships), and then reorder the nodes. In the second framework, we reorder the nodes directly in the TR trees with inserted auxiliary nodes.

3.3 Surface-order Model

The node ordering model is used to determine a projection of the tree to a string. We assume the ordering of the nodes in the input TR trees is arbitrary, the reordering model proposed here is based only on the dependency structure and the node’s attributes (words, POS tags, etc.). In a variant of the reordering model, we assume the deep order of coordinating conjunctions to be the surface order.

Algorithm 1 presents the bottom-up node reordering algorithm. In the first part of the algorithm, we determine the relative ordering of child nodes. We maximize the likelihood of a particular order via the precedence operator \prec . If node $c_i \prec c_{i+1}$, then the subtree of the word associated with c_i immediately precedes the subtree of the word associated with c_{i+1} in the projected sentence.

In the second half of the algorithm (starting at line 13), we predict the position of the governor within the previously ordered child nodes. Recall

Algorithm 1 Subtree Reordering Algorithm

```
procedure REORDER( $T, A, O$ ) ▷ Result in  $O$ 
   $N \leftarrow \text{bottomUp}(T \cup A)$ ;    $O \leftarrow \{\}$ 
  for  $g \in N$  do
    bestScore  $\leftarrow 0$ ;    $o_g \leftarrow \{\}$ 
5:   for  $C \leftarrow$  permutation of  $g$ 's children do
     for  $i \leftarrow 1 \dots |C|$  do
        $s \leftarrow s * P(c_i \prec c_{i+1} | c_i, c_{i+1}, g)$ 
     end for
10:    if  $s > \text{bestScore}$  then
      bestScore  $\leftarrow s$ ;    $o_g \leftarrow C$ 
    end if
  end for
  bestScore  $\leftarrow 0$ ;    $m \leftarrow 0$ 
  for  $i \leftarrow 1 \dots |\text{bestOrder}|$  do
15:    $s \leftarrow P(c_i \prec g \prec c_{i+1} | c_i, c_{i+1}, g)$ 
     if  $s > \text{bestScore}$  then
        $s \leftarrow \text{bestScore}$ ;    $m \leftarrow i$ 
     end if
  end for
20:   Insert governor  $c_g$  after  $m^{\text{th}}$  child in  $o_g$ 
    $O \leftarrow O \cup o_g$ 
end for
end procedure
```

that this is a dependency structure; knowing the governor does not tell us where it lies on the surface with respect to its children. The model is similar to the general reordering model, except we consider an absolute ordering of three nodes (left child, governor, right child). Finally, we can reconstruct the total ordering from the subtree ordering defined in $O = \{o_1, \dots, o_n\}$.

The procedure described here is greedy; first we choose the best child ordering and then we choose the location of the governor. We do this to minimize the computational complexity of the algorithm. The current algorithm's runtime complexity is $O(n!)$, but the complexity of the alternative algorithm for which we consider triples of child nodes is $O(n!(n-1)!)$. The actual complexity is determined by the maximum number of child nodes $k = |C|$ and is $O(\frac{n}{k}k!)$.

3.4 Morphological Generation

In order to produce true English sentences, we convert the lemma and POS tag to a word form. We use John Carroll's morphg tool⁷ to generate English word forms given lemma/POS tag pairs. This is not perfect, but it performs an adequate job at recovering English inflected forms. In the complete-system evaluation, we report scores based on gener-

⁷Available on the web at:

<http://www.informatics.susx.ac.uk/research/nlp/carroll/morph.html>.

ated morphological forms.

3.5 Insertion Features

Features for the insertion model come from the current node being examined and the node's governor. When the governor is a coordinating conjunction, we use features from the governor of the conjunction node. The features used are the lemma, POS tag, and functor for the current node, and the lemma, POS tag, and functor of the governor.

$$\prod_i P(a_i | w_i, w_g) \quad (6)$$
$$= \prod_i P(a_i | l_i, t_i, f_i, l_g, t_g, f_g)$$

The left-hand side of Equation 6 is repeated from Equation 5 above. Equation 6 shows the expanded model for auxiliary insertion where l_i is the lemma, t_i is the POS tag, and f_i is the functor of node w_i .

3.6 Reordering Features

Our reordering model for English is based primarily on non-lexical features. We use the POS tag and functor from each node as features. The two distributions in our reordering model (used in Algorithm 1) are:

$$P(c_i \prec c_{i+1} | c_i, c_{i+1}, g) \quad (7)$$

$$= (c_i \prec c_{i+1} | f_i, t_i, f_{i+1}, t_{i+1}, f_g, t_g)$$

$$P(c_i \prec g \prec c_{i+1} | c_i, c_{i+1}, g) \quad (8)$$

$$= P(c_i \prec g \prec c_{i+1} | f_i, t_i, f_{i+1}, t_{i+1}, t_g, f_g)$$

In both Equation 7 and Equation 8, only the functor and POS tag of each node is used.

4 Empirical Evaluation

We have experimented with the above models on both manually annotated TR trees and synthetic trees (i.e., automatically generated trees). The data comes from the PCEDT 1.0 corpus⁸, a version of the Penn WSJ Treebank that has been translated to Czech and automatically transformed to TR in both English and Czech. The English TR was automatically generated from the Penn Treebank's manually annotated surface syntax trees (English phrase-structure trees). Additionally, a small set of 497 sentences were manually annotated at the TR level: 248

⁸LDC catalog number: LDC2004T25.

Model	Manual Data				Synthetic Data			
	Ins. Rules		No Rules		Ins. Rules		No Rules	
Model	Articles	Prep & SC	Articles	Prep & SC	Articles	Prep & SC	Articles	Prep & SC
Baseline	N/A	N/A	77.93	76.78	N/A	N/A	78.00	78.40
w/o g. functor	87.29	89.65	86.25	89.31	88.07	91.83	87.34	91.06
w/o g. lemma	86.77	89.48	85.68	89.02	87.53	90.95	86.55	91.16
w/o g. POS	87.29	89.45	86.10	89.14	87.68	91.86	86.89	92.07
w/o functor	86.10	85.02	84.86	84.56	86.01	85.60	84.79	85.65
w/o lemma	81.34	89.02	80.88	88.91	81.28	91.03	81.42	91.33
w/o POS	84.81	88.01	84.01	87.29	85.53	91.08	84.69	90.98
All Features	87.49	89.68	86.45	89.28	87.87	91.83	87.24	92.02

Table 1: Classification accuracy for insertion models on development data from PCEDT 1.0. Article accuracy is computed over the set of nouns. Preposition and subordinating conjunction accuracy (P & SC) is computed over the set of nodes that appear on the surface (excluding hidden nodes in the TR – these will not exist in automatically generated data). Models are shown for all features minus the specified feature. Features with the prefix “g.” indicate governor features, otherwise the features are from the node’s attributes. The Baseline model is one which never inserts any nodes (i.e., the model which inserts the most probable value – NOAUX).

for development and 249 for evaluation; results are presented for these two datasets.

All models were trained on the PCEDT 1.0 data set, approximately 49,000 sentences, of which 4,200 were randomly selected as held-out training data, the remainder was used for training. We estimate the model distributions with a smoothed maximum likelihood estimator, using Jelinek-Mercer EM smoothing (i.e., linearly interpolated backoff distributions). Lower order distributions used for smoothing are estimated by deleting the rightmost conditioning variable (as presented in the above models).

Similar experiments were performed at the 2002 Johns Hopkins summer workshop. The results reported here are substantially better than those reported in the workshop report (Hajič et al., 2002); however, the details of the workshop experiments are not clear enough to ensure the experimental conditions are identical.

4.1 Insertion Results

For each of the two insertion models (the article model and the preposition and subordinating conjunction model), there is a finite set of values for the dependent variable a_i . For example, the articles are the complete set of English articles as collected from the Penn Treebank training data (these have manual POS tag annotations). We add a dummy value to this set which indicates no article should be inserted.⁹ The preposition and auxiliary model

assumes the set of possible modifiers to be all those seen in the training data that were removed when modifying the manual TR trees.

The classification accuracy is the percentage of nodes for which we predicted the correct auxiliary from the set of candidate nodes for the auxiliary type. Articles are only predicted and evaluated for nouns (determined by the POS tag). Prepositions and subordinating conjunctions are predicted and evaluated for all nodes that appear on the surface. We do not report results for the modal verb insertion as it is primarily determined by the features of the verb being modified (accuracy is approximately 100%). We have experimented with different features sets and found that the model described in Equation 6 performs best when all features are used.

In a variant of the insertion model, when the classifier prediction is of low certainty (probability less than .5) we defer to a small set of deterministic rules. For infinitives, we insert “to”; for origin nouns, we insert “from”, for actors we insert “of”, and we attach “by” to actors of passive verbs. In the article insertion model, we do not insert anything if there is another determiner (e.g., “none” or “any”) or personal pronoun; we insert “the” if the word appeared within the previous four sentences or if there is a *suggestive* adjective attached to the noun.¹⁰

Table 1 shows that the classifiers perform better on automatically generated data (Synthetic Data), but also perform well on the manually annotated

⁹In the classifier evaluation we consider the article a and an to be equivalent.

¹⁰Any adjective that is always followed by the definite article in the training data.

Model	Manual Data				Synthetic Data			
	Coord. Rules		No Rules		Coord. Rules		No Rules	
	All	Interior	All	Interior	All	Interior	All	Interior
Baseline	N/A	N/A	68.43	21.67	N/A	N/A	69.00	21.42
w/o g. functor	94.51	86.44	92.42	81.27	94.90	87.25	93.37	83.42
w/o g. tag	93.43	83.75	90.89	77.50	93.82	84.56	91.64	79.12
w/o c. functors	91.38	78.70	89.71	74.57	91.91	79.79	90.41	76.04
w/o c. tags	88.85	72.44	82.29	57.36	88.91	72.29	83.04	57.60
All Features	94.43	86.24	92.01	80.26	95.21	88.04	93.37	83.42

Table 2: Reordering accuracy for TR trees on development data from PCEDT 1.0. We include performance on the interior nodes (excluding leaf nodes) for the Manual data to show a more detailed analysis of the performance. “g.” are the governor features and “c.” are the child features. The baseline model sorts subtrees of each node randomly.

data. Prediction of articles is primarily dependent on the lemma and the tag of the node. The lemma and tag of the governing node and the node’s functor is important to a lesser degree. In predicting the prepositions and subordinating conjunctions, the node’s functor is the most critical factor.

% Errors	Reference→Hypothesis
41	the → NULL
19	a/an → NULL
16	NULL → the
11	a/an → the
11	the → a/an
2	NULL → a/an

Table 3: Article classifier errors on development data.

Manual		Synthetic	
Det.	P & SC	Det.	P & SC
85.53	89.18	85.31	91.54

Table 4: Accuracy of best models on the evaluation data.

Table 3 presents a confusion set from the best article classifier on the development data. Our model is relatively conservative, incurring 60% of the error by choosing to insert nothing when it should have inserted an article. The model requires more informed features as we are currently being overly conservative.

In Table 4 we report the overall accuracy on evaluation data using the model that performed best on the development data. The results are consistent with the results for the development data; however, the article model performs slightly worse on the evaluation set.

4.2 Reordering Results

Evaluation of the final sentence ordering was based on predicting the correct words in the correct po-

sitions. We use the reordering metric described in Hajič et al. (2002) which computes the percentage of nodes for which all children are correctly ordered (i.e., no credit for partially correct orderings).

Table 2 shows the reordering accuracy for the full model and variants where a particular feature type is removed. These results are for ordering the correct auxiliary-inserted TR trees (using deep-syntactic functors and the correctly inserted auxiliaries). In the model variant that preserves the deep order of coordinating conjunctions, we see a significant increase in performance. The child node tags are critical for the reordering model, followed by the child functors.

4.3 Combined System Results

Model	Manual	Synthetic
TR w/ Rules	.4614	.4777
TR w/o Rules	.4532	.4657
AR	.2337	.2451

Table 5: BLEU scores for complete generation system for TR trees (with and without rules applied) and the AR trees.

In order to evaluate the combined system, we used the multiple-translation dataset in the PCEDT corpus. This data contains four retranslations from Czech to English of each of the original English sentences in the development and evaluation datasets. In Table 5 we report the BLEU scores on development data for our TR generation model (including the morphological generation module) and the AR generation model. Results for the system that uses AR trees as an intermediate stage are very poor; this is likely due to the noise introduced when generating AR trees. Additionally, the results for the TR model with the additional rules are consistent with the pre-

vious results; the rules provide only a marginal improvement. Finally, we have run the complete system on the evaluation data and achieved a BLEU score of **.4633** on the manual data and **.4750** on the synthetic data. These can be interpreted as the upper-bound for Czech-English translation systems based on TR tree transduction.

5 Conclusion

We have provided a model for sentence synthesis from Tectogrammatical Representation trees. We provide a number of models based on relatively simple, local features that can be extracted from impoverished TR trees. We believe that further improvements will be made by allowing for more flexible use of the features. The current model uses simple linear interpolation smoothing which limits the types of model features used (forcing an explicit factorization). The advantage of simple models of the type presented in this paper is that they are robust to errors in the TR trees – which are expected when the TR trees are generated automatically (e.g., in a machine translation system).

Acknowledgments

This work was partially supported by U.S. NSF grants IIS-9982329 and OISE-0530118; by the project of the Czech Ministry of Education #LC536; by the Information Society Project No. 1ET201120505 of the Grant Agency of the Academy of Sciences of the Czech Republic; and Grant No. 352/2006 of the Grant Agency of Charles University.

References

- Srinivas Bangalore and Owen Rambow. 2000. Exploiting a probabilistic hierarchical model for generation. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING 2000)*, Saarbrücken, Germany.
- Alena Böhmová, Jan Hajič, Eva Hajičová, and Barbora Vidová Hladká. 2002. The prague dependency treebank: Three-level annotation scenario. In Anne Abeille, editor, *In Treebanks: Building and Using Syntactically Annotated Corpora*. Dordrecht, Kluwer Academic Publishers, The Netherlands.
- Joan Bresnan and Ronald M. Kaplan. 1982. Lexical-functional grammar: A formal system for grammatical representation. In *The Mental Representation of Grammatical Relations*. MIT Press.
- Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 263–270, Ann Arbor, MI.
- Michael Collins. 2003. Head-driven statistical models for natural language processing. *Computational Linguistics*, 29(4):589–637.
- Simon Corston-Oliver, Michael Gamon, Eric Ringger, and Robert Moore. 2002. An overview of Amalgam: A machine-learned generation module. In *Proceedings of the International Natural Language Generation Conference*, pages 33–40, New York, USA.
- Jan Hajič, Martin Čmejrek, Bonnie Dorr, Yuan Ding, Jason Eisner, Dan Gildea, Terry Koo, Kristen Parton, Dragomir Radev, and Owen Rambow. 2002. Natural language generation in the context of machine translation. Technical report, Center for Language and Speech Processing, Johns Hopkins University, Baltimore. Summer Workshop Final Report.
- Irene Langkilde and Kevin Knight. 1998. The practical value of n-grams in generation. In *Proceedings of the International Natural Language Generation Workshop*.
- Irene Langkilde-Geary. 2002. An empirical verification of coverage and correctness for a general-purpose sentence generator. In *Proceedings of the International Natural Language Generation Conference*.
- Carl Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. University of Chicago Press.
- Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal SMT. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 271–279, Ann Arbor, Michigan, June. Association for Computational Linguistics.
- Petr Sgall, Eva Hajičová, and Jarmila Panevová. 1986. *The Meaning of the Sentence in Its Semantic and Pragmatic Aspects*. Kluwer Academic, Boston.
- Radu Soricut and Daniel Marcu. 2006. Stochastic language generation using WIDL-expressions and its application in machine translation and summarization. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics*.
- DeKai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–404.