Natural Language Generation with Tree Conditional Random Fields

Wei Lu[1], **Hwee Tou Ng**[1,2], **Wee Sun Lee**[1,2]
[1]Singapore-MIT Alliance
[2]Department of Computer Science
National University of Singapore
luwei@nus.edu.sg
{nght,leews}@comp.nus.edu.sg

Abstract

This paper presents an effective method for generating natural language sentences from their underlying meaning representations. The method is built on top of a *hybrid tree* representation that jointly encodes both the meaning representation as well as the natural language in a tree structure. By using a tree conditional random field on top of the hybrid tree representation, we are able to explicitly model phrase-level dependencies amongst neighboring natural language phrases and meaning representation components in a simple and natural way. We show that the additional dependencies captured by the tree conditional random field allows it to perform better than directly inverting a previously developed hybrid tree semantic parser. Furthermore, we demonstrate that the model performs better than a previous state-of-the-art natural language generation model. Experiments are performed on two benchmark corpora with standard automatic evaluation metrics.

1 Introduction

One of the ultimate goals in the field of natural language processing (NLP) is to enable computers to converse with humans through human languages. To achieve this goal, two important issues need to be studied. First, it is important for computers to capture the meaning of a natural language sentence in a meaning representation. Second, computers should be able to produce a human-understandable natural language sentence from its meaning representation. These two tasks are referred to as semantic parsing and natural language generation (NLG), respectively.

In this paper, we use corpus-based statistical methods for constructing a natural language generation system. Given a set of pairs, where each pair consists of a natural language (NL) sentence and its formal meaning representation (MR), a learning method induces an algorithm that can be used for performing language generation from other previously unseen meaning representations.

A crucial question in any natural language processing system is the representation used. Meaning representations can be in the form of a tree structure. In Lu et al. (2008), we introduced a *hybrid tree* framework together with a probabilistic generative model to tackle semantic parsing, where tree structured meaning representations are used. The hybrid tree gives a natural joint tree representation of a natural language sentence and its meaning representation.

A joint generative model for natural language and its meaning representation, such as that used in Lu et al. (2008) has several advantages over various previous approaches designed for semantic parsing. First, unlike most previous approaches, the generative approach models a simultaneous generation process for both NL and MR. One elegant property of such a joint generative model is that it allows the modeling of both semantic parsing and natural language generation within the same process. Second, the generative process proceeds as a recursive top-down Markov process in a way that takes advantage of the tree structure of the MR. The hybrid tree generative model proposed in Lu et al. (2008) was shown to give state-of-the-art accuracy in semantic parsing on benchmark corpora.

While semantic parsing with hybrid trees has been studied in Lu et al. (2008), its inverse task – NLG with hybrid trees – has not yet been explored. We believe that the properties that make the hybrid trees effective for semantic parsing also make them effective for NLG. In this paper, we develop systems for the generation task by building

Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, pages 400–409,
Singapore, 6-7 August 2009.

on top of the generative model introduced in Lu et al. (2008) (referred to as the LNLZ08 system).

We first present a baseline model by directly "inverting" the LNLZ08 system, where an NL sentence is generated word by word. We call this model the *direct inversion model*. This model is unable to model some long range global dependencies over the entire NL sentence to be generated. To tackle several weaknesses exhibited by the baseline model, we next introduce an alternative, novel model that performs generation at the phrase level. Motivated by conditional random fields (CRF) (Lafferty et al., 2001), a different parameterization of the conditional probability of the hybrid tree that enables the model to encode some longer range dependencies amongst phrases and MRs is used. This novel model is referred to as the *tree* CRF*-based model*.

Evaluation results for both models are presented, through which we demonstrate that the tree CRF-based model performs better than the direct inversion model. We also compare the tree CRF-based model against the previous state-of-the-art model of Wong and Mooney (2007). Furthermore, we evaluate our model on a dataset annotated with several natural languages other than English (Japanese, Spanish, and Turkish). Evaluation results show that our proposed tree CRF-based model outperforms the previous model.

2 Related Work

There have been substantial earlier research efforts on investigating methods for transforming MR to their corresponding NL sentences. Most of the recent systems tackled the problem through the architecture of chart generation introduced by Kay (1996). Examples of such systems include the chart generator for Head-Driven Phrase Structure Grammar (HPSG) (Carroll et al., 1999; Carroll and Oepen, 2005; Nakanishi et al., 2005), and more recently for Combinatory Categorial Grammar (CCG) (White and Baldridge, 2003; White, 2004). However, most of these systems only focused on surface realization (inflection and ordering of NL words) and ignored lexical selection (learning the mappings from MR domain concepts to NL words).

The recent work by Wong and Mooney (2007) explored methods for generation by inverting a system originally designed for semantic parsing. They introduced a system named WASP^{-1} that employed techniques from statistical machine translation using Synchronous Context-Free Grammar (SCFG) (Aho and Ullman, 1972). The system took in a linearized MR tree as input, and translated it into a natural language sentence as output. Unlike most previous systems, their system integrated both lexical selection and surface realization in a single framework. The performance of the system was enhanced by incorporating models borrowed from PHARAOH (Koehn, 2004). Experiments show that this new hybrid system named WASP^{-1}++ gives state-of-the-art accuracies and outperforms the direct translation model obtained from PHARAOH, when evaluated on two corpora. We will compare our system's performance against that of WASP^{-1}++ in Section 5.

3 The Hybrid Tree Framework and the LNLZ08 System

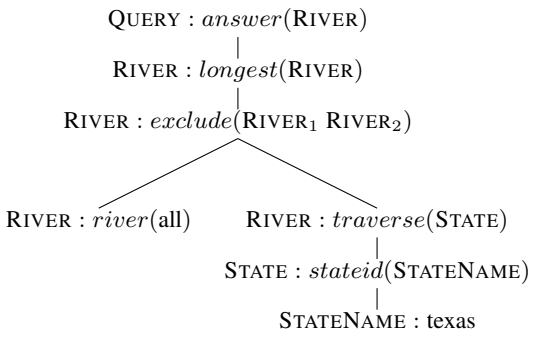

Figure 1: An example MR paired with its NL sentence.

Following most previous works in this area (Kate et al., 2005; Ge and Mooney, 2005; Kate and Mooney, 2006; Wong and Mooney, 2006; Lu et al., 2008), we consider MRs in the form of tree structures. An example MR and its corresponding natural language sentence are shown in Figure 1. The MR is a tree consisting of nodes called MR productions. For example, the node "QUERY : $answer$(RIVER)" is one MR production. Each MR production consists of a semantic category ("QUERY"), a function symbol ("$answer$") which can be optionally omitted, as well as an argument list which possibly contains

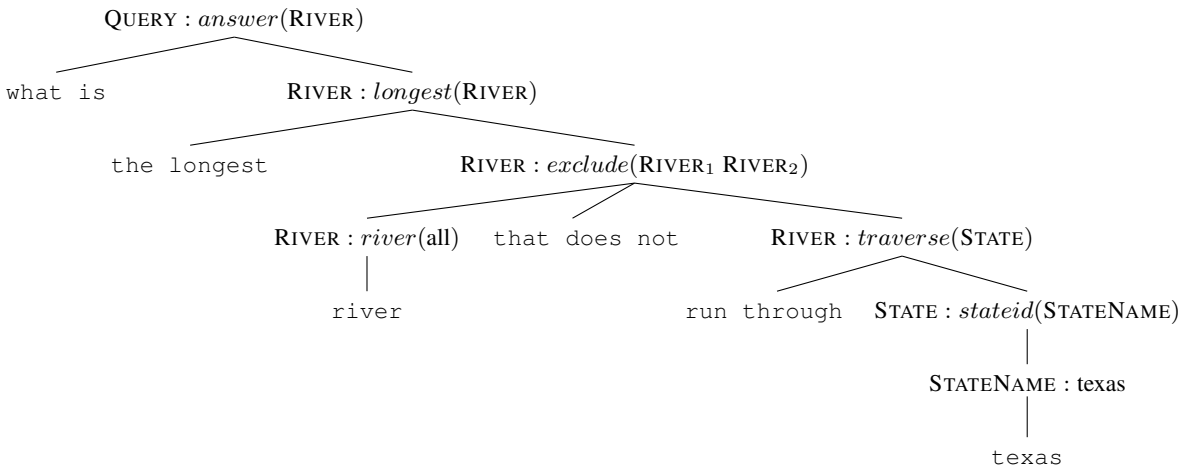

Figure 2: One possible hybrid tree $\mathcal{T}_1$

child semantic categories ("RIVER").

Now we give a brief overview of the hybrid tree framework and the LNLZ08 system that was presented in Lu et al. (2008). The training corpus required by the LNLZ08 system contains example pairs $\mathbf{d}^{(i)} = (\widehat{\mathbf{m}}^{(i)}, \widehat{\mathbf{w}}^{(i)})$ for $i = 1 \ldots N$, where each $\widehat{\mathbf{m}}^{(i)}$ is an MR, and each $\widehat{\mathbf{w}}^{(i)}$ is an NL sentence. The system makes the assumption that the entire training corpus is generated from an underlying generative model, which is specified by the parameter set Ω.

The parameter set Ω includes the following: the MR model parameter $\rho(m_j|m_i, \arg_k)$ which models the generation of an MR production m_j from its parent MR production m_i as its k-th child, the emission parameter $\theta(t|m_i, \Lambda)$ that is responsible for generation of an NL word or a semantic category t from the MR production m_i (the parent of t) under the context Λ (such as the token to the left of the current token), and the pattern parameter $\phi(r|m_i)$, which models the selection of a *hybrid pattern* r that defines globally how the NL words and semantic categories are interleaved given a parent MR production m_i. All these parameters are estimated from the corpus during the training phase. The list of possible hybrid patterns is given in Table 1 (at most two child semantic categories are allowed – MR productions with more child semantic categories are transformed into those with two).

In the table, m refers to the MR production, the symbol **w** denotes an NL word sequence and is optional if it appears inside []. The symbol $\mathcal{Y}$ and $\mathcal{Z}$ refer to the first and second semantic category under the MR production m respectively.

# RHS	Hybrid Pattern	# Patterns
0	$m \rightarrow$ **w**	1
1	$m \rightarrow$ [**w**]$\mathcal{Y}$[**w**]	4
2	$m \rightarrow$ [**w**]$\mathcal{Y}$[**w**]$\mathcal{Z}$[**w**]	8
	$m \rightarrow$ [**w**]$\mathcal{Z}$[**w**]$\mathcal{Y}$[**w**]	8

Table 1: The list of possible hybrid patterns, [] denotes optional

The generative process recursively creates MR productions as well as NL words at each generation step in a top-down manner. This process results in a *hybrid tree* for each MR-NL pair. The list of children under each MR production in the hybrid tree forms a *hybrid sequence*. One example hybrid tree for the MR-NL pair given in Figure 1 is shown in Figure 2. In this hybrid tree $\mathcal{T}_1$, the list of children under the production RIVER : $longest$(RIVER) forms the hybrid sequence "`the longest` RIVER : $exclude$(RIVER$_1$ RIVER$_2$)". The yield of the hybrid tree is exactly the NL sentence. The MR can also be recovered from the hybrid tree by recording all the internal nodes of the tree, subject to the reordering operation required by the hybrid pattern.

To illustrate, consider the generation of the hybrid tree $\mathcal{T}_1$ shown in Figure 2. The model first generates an MR production from its parent MR production (empty as the MR production is the root in the MR). Next, it selects a hybrid pattern $m \rightarrow \mathbf{w}\mathcal{Y}$ from the predefined list of hybrid patterns, which puts a constraint on the set of all allowable hybrid sequences that can be generated: the hybrid sequence must be an NL word sequence

followed by a semantic category. Finally, actual NL words and semantic categories are generated from the parent MR production. Now the generation for one level is complete, and the above process repeats at the newly generated MR productions, until the complete NL sentence and MR are both generated.

Mathematically, the above generative process yields the following formula that models the joint probability for the MR-NL pair, assuming the context Λ for the emission parameter is the preceding word or semantic category (*i.e.*, the bigram model is assumed, as discussed in Lu et al. (2008)):

$$\begin{aligned} & p\big(\mathcal{T}_1(\widehat{\mathbf{w}}, \widehat{\mathbf{m}})\big) \\ &= \rho(\textsc{Query} : answer(\textsc{River}) | -, \arg_1) \\ &\times \phi(m \rightarrow \mathbf{w}\mathcal{Y} | \textsc{Query} : answer(\textsc{River})) \\ &\times \theta(\texttt{what} | \textsc{Query} : answer(\textsc{River}), \texttt{BEGIN}) \\ &\times \theta(\texttt{is} | \textsc{Query} : answer(\textsc{River}), \texttt{what}) \\ &\times \theta(\textsc{River} | \textsc{Query} : answer(\textsc{River}), \texttt{is}) \\ &\times \theta(\texttt{END} | \textsc{Query} : answer(\textsc{River}), \textsc{River}) \\ &\times \rho(\textsc{River} : longest(\textsc{River}) | \\ &\textsc{Query} : answer(\textsc{River}), \arg_1) \times \ldots \end{aligned} \quad (1)$$

where $\mathcal{T}_1(\widehat{\mathbf{w}}, \widehat{\mathbf{m}})$ denotes the hybrid tree $\mathcal{T}_1$ which contains the NL sentence $\widehat{\mathbf{w}}$ and MR $\widehat{\mathbf{m}}$.

For each MR-NL pair in the training set, there can be potentially many possible hybrid trees associated with the pair. However, the correct hybrid tree is completely unknown during training. The correct hybrid tree is therefore treated as a hidden variable. An efficient inside-outside style algorithm (Baker, 1979) coupled with further dynamic programming techniques is used for efficient parameter estimation.

During the testing phase, the system makes use of the learned model parameters to determine the most probable hybrid tree given a new natural language sentence. The MR contained in that hybrid tree is the output of the system. Dynamic programming techniques similar to those of training are also employed for efficient decoding.

The generative model used in the LNLZ08 system has a natural symmetry, allowing for easy transformation from NL to MR, as well as from MR to NL. This provides the starting point for our work in "inverting" the LNLZ08 system to generate natural language sentences from the underlying meaning representations.

4 Generation with Hybrid Trees

The task of generating NL sentences from MRs can be defined as follows. Given a training corpus consisting of MRs paired with their NL sentences, one needs to develop algorithms that learn how to effectively "paraphrase" MRs with natural language sentences. During testing, the system should be able to output the most probable NL "paraphrase" for a given new MR.

The LNLZ08 system models $p(\mathcal{T}(\widehat{\mathbf{w}}, \widehat{\mathbf{m}}))$, the joint generative process for the hybrid tree containing both NL and MR. This term can be rewritten in the following way:

$$p(\mathcal{T}(\widehat{\mathbf{w}}, \widehat{\mathbf{m}})) = p(\widehat{\mathbf{m}}) \times p\left(\mathcal{T}(\widehat{\mathbf{w}}, \widehat{\mathbf{m}}) | \widehat{\mathbf{m}}\right) \quad (2)$$

In other words, we reach an alternative view of the joint generative process as follows. We choose to generate the complete MR $\widehat{\mathbf{m}}$ first. Given $\widehat{\mathbf{m}}$, we generate hybrid sequences below each of its MR production, which gives us a complete hybrid tree $\mathcal{T}(\widehat{\mathbf{w}}, \widehat{\mathbf{m}})$. The NL sentence $\widehat{\mathbf{w}}$ can be constructed from this hybrid tree exactly.

We define an operation $yield(\mathcal{T})$ which returns the NL sentence as the yield of the hybrid tree $\mathcal{T}$. Given an MR $\widehat{\mathbf{m}}$, we find the most probable NL sentence $\widehat{\mathbf{w}}^*$ as follows:

$$\widehat{\mathbf{w}}^* = yield\Big(\operatorname*{argmax}_{\mathcal{T}} p(\mathcal{T} | \widehat{\mathbf{m}})\Big) \quad (3)$$

In other words, we first find the most probable hybrid tree $\mathcal{T}$ that contains the provided MR $\widehat{\mathbf{m}}$. Next we return the yield of $\mathcal{T}$ as the most probable NL sentence.

Different assumptions can be made in the process of finding the most probable hybrid tree. We first describe a simple model which is a direct inversion of the LNLZ08 system. This model, as a baseline model, generates a complete NL sentence word by word. Next, a more sophisticated model that exploits NL phrase-level dependencies is built that tackles some weaknesses of the simple baseline model.

4.1 Direct Inversion Model

Assume that a pre-order traversal of the MR $\widehat{\mathbf{m}}$ gives us the list of MR productions $m_1, m_2, \ldots, m_S$, where S is the number of MR productions in $\widehat{\mathbf{m}}$. Based on the independence assumption made by the LNLZ08 system, each MR production independently generates a hybrid

sequence. Denote the hybrid sequence generated under the MR production m_s as h_s, for $s = 1, \ldots, S$. We call the list of hybrid sequences $\mathbf{h} = \langle h_1, h_2, \ldots, h_S \rangle$ a *hybrid sequence list* associated with this particular MR. Thus, our goal is to find the optimal hybrid sequence list $\mathbf{h}^*$ for the given MR $\widehat{\mathbf{m}}$, which is formulated as follows:

$$\mathbf{h}^* = \langle h_1^*, \ldots, h_S^* \rangle = \underset{h_1,\ldots,h_S}{\operatorname{argmax}} \prod_{s=1}^{S} p(h_s|m_s) \quad (4)$$

The optimal hybrid sequence list defines the optimal hybrid tree whose yield gives the optimal NL sentence.

Due to the strong independence assumption introduced by the LNLZ08 system, the hybrid tree generation process is in fact highly decomposable. Optimization of the hybrid sequence list $\langle h_1, \ldots, h_S \rangle$ can be performed individually since they are independent of one another. Thus, mathematically, for $s = 1, \ldots, S$, we have:

$$h_s^* = \underset{h_s}{\operatorname{argmax}}\, p(h_s|m_s) \quad (5)$$

The LNLZ08 system presented three models for the task of transforming NL to MR. In this inverse task, for generation of a hybrid sequence, we choose to use the bigram model (model II). We choose this model mainly due to its stronger ability in modeling dependencies between adjacent NL words, which we believe to be quite important in this NL generation task. With the bigram model assumption, the optimal hybrid sequence that can be generated from each MR production is defined as follows:

$$\begin{aligned} h_s^* &= \underset{h_s}{\operatorname{argmax}}\, p(h_s|m_s) \\ &= \underset{h_s}{\operatorname{argmax}} \left\{ \phi(r|m_s) \times \prod_{j=1}^{|h_s|+1} \theta(t_j|m_s, t_{j-1}) \right\} \end{aligned} \quad (6)$$

where t_i is either an NL word or a semantic category with $t_0 \equiv$ BEGIN and $t_{|h_s|+1} \equiv$ END, and r is the hybrid pattern that matches the hybrid sequence h_s, which is equivalent to $t_1, \ldots, t_{|h_s|}$.

Equivalently, we can view the problem in the log-space:

$$\begin{aligned} h_s^* = \underset{h_s}{\operatorname{argmin}} \Bigg\{ & -\log \phi(r|m_s) \\ & + \sum_{j=1}^{|h_s|+1} -\log \theta(t_j|m_s, t_{j-1}) \Bigg\} \end{aligned} \quad (7)$$

Note the term $-\log \phi(r|m_s)$ is a constant for a particular MR production m_s and a particular hybrid pattern r. This search problem can be equivalently cast as the shortest path problem which can be solved efficiently with Dijkstra's algorithm (Cormen et al., 2001). We define a set of states. Each state represents a single NL word or a semantic category, including the special symbols BEGIN and END. A directed path between two different states t_u and t_v is associated with a distance measure $-\log \theta(t_v|m_s, t_u)$, which is non-negative. The task now is to find the shortest path between BEGIN and END[1]. The sequence of words appearing in this path is simply the most probable hybrid sequence under this MR production m_s. We build this model by directly inverting the LNLZ08 system, and this model is therefore referred to as the *direct inversion model*.

A major weakness of this baseline model is that it encodes strong independence assumptions during the hybrid tree generation process. Though shown to be effective in the task of transforming NL to MR, such independence assumptions may introduce difficulties in this NLG task. For example, consider the MR shown in Figure 1. The generation steps of the hybrid sequences from the two adjacent MR productions QUERY : $answer$(RIVER) and RIVER : $longest$(RIVER) are completely independent of each other. This may harm the fluency of the generated NL sentence, especially when a transition from one hybrid sequence to another is required. In fact, due to such an independence assumption, the model always generates the same hybrid sequence from the same MR production, regardless of its context such as parent or child MR productions. Such a limitation points to the importance of better utilizing the tree structure of the MR for this NLG task. Furthermore, due to the bigram assumption, the model is unable to capture longer range dependencies amongst the words or semantic categories in each hybrid sequence.

To tackle the above issues, we explore ways of relaxing various assumptions, which leads to an

[1] In addition, we should make sure that the generated hybrid sequence $t_0 \ldots t_{|h_s|+1}$ is a valid hybrid sequence that comply with the hybrid pattern r. For example, the MR production STATE : loc_1(RIVER) can generate the following hybrid sequence "BEGIN have RIVER END" but not this hybrid sequence "BEGIN have END". This can be achieved by finding the shortest path from BEGIN to RIVER, which then gets concatenated to the shortest path from RIVER to END.

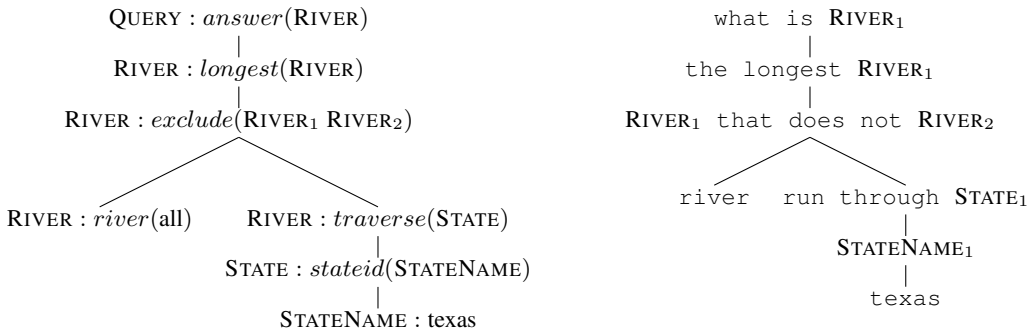

Figure 3: An MR (left) and its associated hybrid sequences (right)

alternative model as discussed next.

4.2 Tree CRF-Based Model

Based on the belief that using known phrases usually leads to better fluency in the NLG task (Wong and Mooney, 2007), we explore methods for generating an NL sentence at phrase level rather than at word level. This is done by generating hybrid sequences as complete objects, rather than sequentially one word or semantic category at a time, from MR productions.

We assume that each MR production can generate a complete hybrid sequence below it from a finite set of possible hybrid sequences. Each such hybrid sequence is called a *candidate hybrid sequence* associated with that particular MR production. Given a set of candidate hybrid sequences associated with each MR production, the generation task is to find the optimal hybrid sequence list $\mathbf{h}^*$ for a given MR $\widehat{\mathbf{m}}$:

$$\mathbf{h}^* = \underset{\mathbf{h}}{\operatorname{argmax}}\, p(\mathbf{h}|\widehat{\mathbf{m}}) \tag{8}$$

Figure 3 shows a complete MR, as well as a possible tree that contains hybrid sequences associated with the MR productions. For example, in the figure the MR production RIVER : $traverse$(STATE) is associated with the hybrid sequence `run through` STATE$_1$. Each MR production can be associated with potentially many different hybrid sequences. The task is to determine the most probable list of hybrid sequences as the ones appearing on the right of Figure 3, one for each MR production.

To make better use of the tree structure of MR, we take the approach of modeling the conditional distribution using a log-linear model. Following the conditional random fields (CRF) framework (Lafferty et al., 2001), we can define the probability of the hybrid sequence list given the complete MR $\widehat{\mathbf{m}}$, as follows:

$$p(\mathbf{h}|\widehat{\mathbf{m}}) = \frac{1}{Z(\widehat{\mathbf{m}})} \exp\left(\sum_{i \in V} \sum_{k} \mu_k g_k(h_i, \widehat{\mathbf{m}}, i) + \sum_{(i,j) \in E} \sum_{k} \lambda_k f_k(h_i, h_j, \widehat{\mathbf{m}}, i, j) \right) \tag{9}$$

where V is the set of all the vertices in the tree, and E is the set of the edges in the tree, consisting of parent-child pairs. The function $Z(\widehat{\mathbf{m}})$ is the normalization function. Note that the dependencies among the features here form a tree, unlike the sequence models used in Lafferty et al. (2001). The function $f_k(h_i, h_j, \widehat{\mathbf{m}}, i, j)$ is a feature function of the entire MR tree $\widehat{\mathbf{m}}$ and the hybrid sequences at vertex i and j. These features are usually referred to as the edge features in the CRF framework. The function $g_k(h_i, \widehat{\mathbf{m}}, i)$ is a feature function of the hybrid sequence at vertex i and the entire MR tree. These features are usually referred to as the vertex features. The parameters λ_k and μ_k are learned from the training data.

In this task, we are given only MR-NL pairs and do not have the hybrid tree corresponding to each MR as training data. Now we describe how the set of candidate hybrid sequences for each MR production is obtained as well as how the training data for this model is constructed. After the joint generative model is learned as done in Lu et al. (2008), we first use a Viterbi algorithm to find the optimal hybrid tree for each MR-NL pair in the training set. From each optimal hybrid tree, we extract the hybrid sequence h_i below each MR production m_i. Using this process on the training MR-NL pairs, we can obtain a set of candidate

hybrid sequences that can be associated with each MR production. The optimal hybrid tree generated by the Viterbi algorithm in this way is considered the "correct" hybrid tree for the MR-NL pair and is used as training data. While this does not provide hand-labeled training data, we believe the hybrid trees generated this way form a high quality training set as both the MR and NL are available when Viterbi decoding is performed, guaranteeing that the generated hybrid tree has the correct yield.

There exist several advantages of such a model over the simple generative model. First, this model allows features that specifically model the dependencies between neighboring hybrid sequences in the tree to be used. In addition, the model can efficiently capture long range dependencies between MR productions and hybrid sequences since each hybrid sequence is allowed to depend on the entire MR tree.

For features, we employ four types of simple features, as presented below. Note that the first three types of features are vertex features, and the last are edge features. Examples are given based on Figure 3. All the features are indicator functions, *i.e.*, a feature takes value 1 if a certain combination is present, and 0 otherwise. The last three features explicitly encode information from the tree structure of MR.

Hybrid sequence features : one hybrid sequence together with the associated MR production. For example:

$$g_1 : \langle \texttt{run through } \textsc{State}_1, \textsc{River} : traverse(\textsc{State}) \rangle ;$$

Two-level hybrid sequence features : one hybrid sequence, its associated MR production, and the parent MR production. For example:

$$g_2 : \langle \texttt{run through } \textsc{State}_1, \textsc{River} : traverse(\textsc{State}), \textsc{River} : exclude(\textsc{River}_1, \textsc{River}_2) \rangle ;$$

Three-level hybrid sequence features : one hybrid sequence, its associated MR production, the parent MR production, and the grandparent MR production. For example:

$$g_3 : \langle \texttt{run through } \textsc{State}_1, \textsc{River} : traverse(\textsc{State}), \textsc{River} : exclude(\textsc{River}_1, \textsc{River}_2), \textsc{River} : longest(\textsc{River}) \rangle ;$$

Adjacent hybrid sequence features : two adjacent hybrid sequences, together with their associated MR productions. For example:

$$f_1 : \langle \texttt{run through } \textsc{State}_1, \textsc{River}_1 \texttt{ that does not } \textsc{River}_2, \textsc{River} : traverse(\textsc{State}), \textsc{River} : exclude(\textsc{River}_1, \textsc{River}_2) \rangle .$$

For training, we use the feature forest model (Miyao and Tsujii, 2008), which was originally designed as an efficient algorithm for solving maximum entropy models for data with complex structures. The model enables efficient training over packed trees that potentially represent exponential number of trees. The tree conditional random fields model can be effectively represented using the feature forest model. The model has also been successfully applied to the HPSG parsing task.

To train the model, we run the Viterbi algorithm on the trained LNLZ08 model and perform convex optimization using the feature forest model. The LNLZ08 model is trained using an EM algorithm with time complexity $O(MN^3D)$ per EM iteration, where M and N are respectively the maximum number of MR productions and NL words for each MR-NL pair, and D is the number of training instances. The time complexity of the Viterbi algorithm is also $O(MN^3D)$. For training the feature forest, we use the Amis toolkit (Miyao and Tsujii, 2002) which utilizes the GIS algorithm. The time complexity for each iteration of the GIS algorithm is $O(MK^2D)$, where K is the maximum number of candidate hybrid sequences associated with each MR production. Finally, the time complexity for generating a natural language sentence from a particular MR is $O(MK^2)$.

5 Experiments

In this section, we present the results of our systems when evaluated on two standard benchmark corpora. The first corpus is GEOQUERY, which contains Prolog-based MRs that can be used to query a US geographic database (Kate et al., 2005). Our task for this domain is to generate NL sentences from the formal queries. The second corpus is ROBOCUP. This domain contains MRs which are instructions written in a formal language called CLANG. Our task for this domain is to generate NL sentences from the coaching advice written in CLANG.

	GEOQUERY (880)		ROBOCUP (300)	
	BLEU	NIST	BLEU	NIST
Direct inversion model	0.3973	5.5466	0.5468	6.6738
Tree CRF-based model	**0.5733**	**6.7459**	**0.6220**	**6.9845**

Table 2: Results of automatic evaluation of both models (**bold** type indicates the best performing system).

	GEOQUERY (880)		ROBOCUP (300)	
	BLEU	NIST	BLEU	NIST
WASP^{-1}++	0.5370	6.4808	0.6022	6.8976
Tree CRF-based model	**0.5733**	**6.7459**	**0.6220**	**6.9845**

Table 3: Results of automatic evaluation of our tree CRF-based model and WASP^{-1}++.

	English		*Japanese*		*Spanish*		*Turkish*	
	BLEU	NIST	BLEU	NIST	BLEU	NIST	BLEU	NIST
WASP^{-1}++	0.6035	5.7133	0.6585	4.6648	0.6175	5.7293	0.4824	4.3283
Tree CRF-based model	**0.6265**	**5.8907**	**0.6788**	**4.8486**	**0.6382**	**5.8488**	**0.5096**	**4.5033**

Table 4: Results on the GEOQUERY-250 corpus with 4 natural languages.

The GEOQUERY domain contains 880 instances, while the ROBOCUP domain contains 300 instances. The average NL sentence length for the two corpora are 7.57 and 22.52 respectively. Following the evaluation methodology of Wong and Mooney (2007), we performed 4 runs of the standard 10-fold cross validation and report the averaged performance in this section using the standard automatic evaluation metric BLEU (Papineni et al., 2002) and NIST (Doddington, 2002)[2]. The BLEU and NIST scores of the WASP^{-1}++ system reported in this section are obtained from the published paper of Wong and Mooney (2007). Note that to make our experimental results directly comparable to Wong and Mooney (2007), we used the identical training and test data splits for the 4 runs of 10-fold cross validation used by Wong and Mooney (2007) on both corpora.

Our system has the advantage of always producing an NL sentence given any input MR, even if there exist unseen MR productions in the input MR. We can achieve this by simply skipping those unseen MR productions during the generation process. However, in order to make a fair comparison against WASP^{-1}++, which can only generate NL sentences for 97% of the input MRs, we also do not generate any NL sentence in the case of observing an unseen MR production. All the evaluations discussed in this section follow this evaluation methodology, but we notice that empirically our system is able to achieve higher BLEU/NIST scores if we allow generation for those MRs that include unseen MR productions.

[2]We used the official evaluation script (version 11b) provided by http://www.nist.gov/.

5.1 Comparison between the two models

We compare the performance of our two models in Table 2. From the table, we observe that the tree CRF-based model outperforms the direct inversion model on both domains. This validates our earlier belief that some long range dependencies are important for the generation task. In addition, while the direct inversion model performs reasonably well on the ROBOCUP domain, it performs substantially worse on the GEOQUERY domain where the sentence length is shorter. We note that the evaluation metrics are strongly correlated with the cumulative matching n-grams between the output and the reference sentence (n ranges from 1 to 4 for BLEU, and 1 to 5 for NIST). The direct inversion model fails to capture the transitional behavior from one phrase to another, which makes it more vulnerable to n-gram mismatch, especially when evaluated on the GEOQUERY corpus where phrase-to-phrase transitions are more frequent. On the other hand, the tree CRF-based model does not suffer from this problem, mainly due to its ability to model such dependencies between neighboring phrases. Sample outputs from the two models are shown in Figure 4.

Reference:	`what is the largest state bordering texas`
Direct inversion model:	`what the largest states border texas`
Tree CRF-based model:	`what is the largest state that borders texas`
Reference:	`if DR2C7 is true then players 2 , 3 , 7 and 8 should pass to player 4`
Direct inversion model:	`if DR2C7 , then players 2 , 3 7 and 8 should ball to player 4`
Tree CRF-based model:	`if the condition DR2C7 is true then players 2 , 3 , 7 and 8 should pass to player 4`

Figure 4: Sample outputs from the two models, for GEOQUERY domain (top) and ROBOCUP domain (bottom) respectively.

5.2 Comparison with previous model

We also compare the performance of our tree CRF-based model against the previous state-of-the-art system WASP^{-1}++ in Table 3. Our tree CRF-based model achieves better performance on both corpora. We are unable to carry out statistical significance tests since the detailed BLEU and NIST scores of the cross validation runs of WASP^{-1}++ as reported in the published paper of Wong and Mooney (2007) are not available.

The results confirm our earlier discussions: the dependencies between the generated NL words are important and need to be properly modeled. The WASP^{-1}++ system uses a log-linear model which incorporates two major techniques to attempt to model such dependencies. First, a back-off language model is used to capture dependencies at adjacent word level. Second, a technique that merges smaller translation rules into a single rigid rule is used to capture dependencies at phrase level (Wong, 2007). In contrast, the proposed tree CRF-based model is able to explicitly and flexibly exploit phrase-level features that model dependencies between adjacent phrases. In fact, with the hybrid tree framework, the better treatment of the tree structure of MR enables us to model some crucial dependencies between the complete MR tree and generated NL phrases. We believe that this property plays an important role in improving the quality of the generated sentences in terms of fluency, which is assessed by the evaluation metrics.

Furthermore, WASP^{-1}++ employs minimum error rate training (Och, 2003) to directly optimize the evaluation metrics. We have not done so but still obtain better performance. In future, we plan to explore ways to directly optimize the evaluation metrics in our system.

5.3 Experiments on different languages

Following the work of Wong and Mooney (2007), we also evaluated our system's performance on a subset of the GEOQUERY corpus with 250 instances, where sentences of 4 natural languages (English, Japanese, Spanish, and Turkish) are available. The evaluation results are shown in Table 4. Our tree CRF-based model achieves better performance on this task compared to WASP^{-1}++. We are again unable to conduct statistical significance tests for the same reason reported earlier.

6 Conclusions

In this paper, we presented two novel models for the task of generating natural language sentences from given meaning representations, under a hybrid tree framework. We first built a simple direct inversion model as a baseline. Next, to address the limitations associated with the direct inversion model, a tree CRF-based model was introduced. We evaluated both models on standard benchmark corpora. Evaluation results show that the tree CRF-based model performs better than the direct inversion model, and that the tree CRF-based model also outperforms WASP^{-1}++, which was a previous state-of-the-art system reported in the literature.

Acknowledgments

The authors would like to thank Seung-Hoon Na for his suggestions on the presentation of this paper, Yuk Wah Wong for answering various questions related to the WASP^{-1}++ system, and the anonymous reviewers for their thoughtful comments on this work.

References

Alfred V. Aho and Jeffrey D. Ullman. 1972. *The Theory of Parsing, Translation and Compiling*. Prentice-Hall, Englewood Clis, NJ.

James K. Baker. 1979. Trainable grammars for speech recognition. In *Proceedings of the Spring Conference of the Acoustical Society of America*, pages 547–550, Boston, MA, June.

John Carroll and Stephan Oepen. 2005. High efficiency realization for a wide-coverage unification grammar. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing (IJCNLP 2005)*, pages 165–176.

John Carroll, Ann Copestake, Dan Flickinger, and Victor Poznanski. 1999. An efficient chart generator for (semi-) lexicalist grammars. In *Proceedings of the 7th European Workshop on Natural Language Generation (EWNLG 1999)*, pages 86–95.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2001. *Introduction to Algorithms (Second Edition)*. MIT Press.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the 2nd International Conference on Human Language Technology Research (HLT 2002)*, pages 138–145.

Ruifang Ge and Raymond J. Mooney. 2005. A statistical semantic parser that integrates syntax and semantics. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL 2005)*, pages 9–16.

Rohit J. Kate and Raymond J. Mooney. 2006. Using string-kernels for learning semantic parsers. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL 2006)*, pages 913–920.

Rohit J. Kate, Yuk Wah Wong, and Raymond J. Mooney. 2005. Learning to transform natural to formal languages. In *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI 2005)*, pages 1062–1068.

Martin Kay. 1996. Chart generation. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL 1996)*, pages 200–204.

Philipp Koehn. 2004. Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *Proceedings of the 6th Conference of the Association for Machine Translation in the Americas (AMTA 2004)*, pages 115–124.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning (ICML 2001)*, pages 282–289.

Wei Lu, Hwee Tou Ng, Wee Sun Lee, and Luke S. Zettlemoyer. 2008. A generative model for parsing natural language to meaning representations. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP 2008)*, pages 783–792.

Yusuke Miyao and Jun'ichi Tsujii. 2002. Maximum entropy estimation for feature forests. In *Proceedings of the 2nd International Conference on Human Language Technology Research (HLT 2002)*, pages 292–297.

Yusuke Miyao and Jun'ichi Tsujii. 2008. Feature forest models for probabilistic HPSG parsing. *Computational Linguistics*, 34(1):35–80.

Hiroko Nakanishi, Yusuke Miyao, and Jun'ichi Tsujii. 2005. Probabilistic models for disambiguation of an HPSG-based chart generator. In *Proceedings of the 9th International Workshop on Parsing Technologies (IWPT 2005)*, volume 5, pages 93–102.

Franz J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL 2003)*, pages 160–167.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pages 311–318.

Michael White and Jason Baldridge. 2003. Adapting chart realization to CCG. In *Proceedings of the 9th European Workshop on Natural Language Generation (EWNLG 2003)*, pages 119–126.

Michael White. 2004. Reining in CCG chart realization. In *Proceeding of the 3rd International Conference on Natural Language Generation (INLG 2004)*, pages 182–191.

Yuk Wah Wong and Raymond J. Mooney. 2006. Learning for semantic parsing with statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL 2006)*, pages 439–446.

Yuk Wah Wong and Raymond J. Mooney. 2007. Generation by inverting a semantic parser that uses statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (NAACL/HLT 2007)*, pages 172–179.

Yuk Wah Wong. 2007. *Learning for Semantic Parsing and Natural Language Generation Using Statistical Machine Translation Techniques*. Ph.D. thesis, The University of Texas at Austin.