# Discriminative Training of the Hidden Vector State Model for Semantic Parsing

Deyu Zhou and Yulan He

**Abstract**—In this paper, we discuss how discriminative training can be applied to the hidden vector state (HVS) model in different task domains. The HVS model is a discrete hidden Markov model (HMM) in which each HMM state represents the state of a push-down automaton with a finite stack size. In previous applications, maximum-likelihood estimation (MLE) is used to derive the parameters of the HVS model. However, MLE makes a number of assumptions and unfortunately some of these assumptions do not hold. Discriminative training, without making such assumptions, can improve the performance of the HVS model by discriminating the correct hypothesis from the competing hypotheses. Experiments have been conducted in two domains: the travel domain for the semantic parsing task using the DARPA Communicator data and the Air Travel Information Services (ATIS) data and the bioinformatics domain for the information extraction task using the GENIA corpus. The results demonstrate modest improvements of the performance of the HVS model using discriminative training. In the travel domain, discriminative training of the HVS model gives a relative error reduction rate of 31 percent in F-measure when compared with MLE on the DARPA Communicator data and 9 percent on the ATIS data. In the bioinformatics domain, a relative error reduction rate of 4 percent in F-measure is achieved on the GENIA corpus.

**Index Terms**—Semantic parsing, information extraction, hidden vector state model, discriminative training.

## 1 Introduction

Semantic parsing, mapping an input sentence into a structured representation of its meaning, can be applied into several applications such as spoken language understanding, information extraction, etc. An example of semantic parsing for extracting protein-protein interactions (PPI) is given in Fig. 1. The original sentence is mapped to a semantic parse tree from which the PPIs could be easily extracted.

Traditionally, research in the field of semantic parsing can be divided into two categories: rule-based approaches and statistical approaches. Based on handcrafted semantic grammar rules, rule-based approaches fill slots in semantic frames using word pattern and semantic tokens [1], [2]. Such rule-based approaches are typically domain specific and often fragile. Statistical approaches are generally based on stochastic models. Given a model and an observed word sequence $W = (w_1 \cdots w_T)$, semantic parsing can be viewed as a pattern recognition problem and statistical decoding can be used to find the most likely semantic representation. If we assume that the hidden data take the form of a semantic parse tree $C$, then the model should be a push-down automata that can generate the pair $\langle W, C\rangle$ through some canonical sequence of moves $D = (d_1 \cdots d_T)$. That is

$$P(W,C) = \prod_{t=1}^{T} P(d_t | d_{t-1} \cdots d_1). \tag{1}$$

Decision sequences are usually steps in some top-down or bottom-up derivation of trees. For the general case of an unconstrained hierarchical model, $D$ will consist of three types of probabilistic move:

1. popping semantic category labels off the stack,
2. pushing one or more nonterminal semantic category label onto the stack, and
3. generating the next word.

In practice, conditional independence can be used to reduce the number of parameters needed to manageable proportions. As in conventional statistical language modeling, this involves defining an equivalence function $\Phi$, which groups move sequences into equivalent classes. Thus, the final generic parsing model is

$$P(W,C) = \prod_{t=1}^{T} P(d_t | \Phi(d_{t-1} \cdots d_1)). \tag{2}$$

The above is essentially the history-based model [3], [4] where the probability of each parser action is conditioned on the history of previous actions in the parse or some partially built structure.

Traditionally, the parsing models have been trained to have the maximum likelihood (ML) $P(W,C)$. However, the goal of training the parsing models is to find the correct semantic representation $C$, i.e., minimize the recognition error. Although a maximum probability is often correlated with a better recognition performance, the correlation is not perfect and is not proved. We can provide some examples with high likelihood but without a better recognition rate. Furthermore, ML estimation (MLE) makes some assumptions on the model, such as the model correctly represents the underlying stochastic process, the amount of training data is infinite, and the true global maximum of the likelihood can be found. When assumptions made about the model are incorrect and the training data are not sufficient, MLE yields a suboptimal solution.

- *The authors are with the Informatics Research Centre, The University of Reading, 3rd Floor, Philip Lyle Building, whiteknights, Reading, Berkshire, RG6 6BX, UK. E-mail: {sir07dz, y.he}@reading.ac.uk.*

*Manuscript received 17 May 2007; revised 30 Sept. 2007; accepted 30 Apr. 2008; published online 7 May 2008.*
*For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2007-05-0222.*
*Digital Object Identifier no. 10.1109/TKDE.2008.95.*

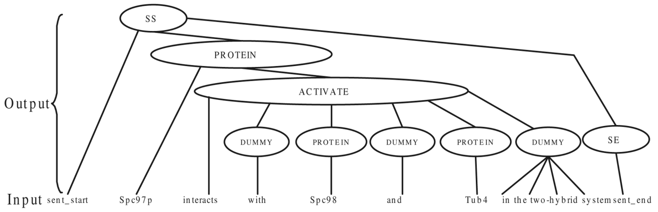


Fig. 1. An example of semantic parsing for PPI extraction.

Previous research has shown that the performance of the hidden Markov model (HMM) trained using MLE can often be improved further using discriminative training. Discriminative training methods based on different criteria such as maximum mutual information (MMI), minimum classification error (MCE), etc, have been tried. In particular, MMI estimation has been studied for speech recognition, and substantial gains in performance have been reported [5], while discriminative training based on MCE [6], [7], [8], [9] has also been applied for speech recognition.

An early example of a purely statistical approach to semantic parsing is the finite state semantic tagger used in AT&T's CHRONUS system [10]. In this system, utterance generation is modeled by an HMM-like process in which the hidden states correspond to semantic concepts and the state outputs correspond to the individual words. This model is sometimes referred to as the flat-concept model to emphasize its inability to represent a hierarchical structure. For the constrained case of the flat concept model, the stack is effectively depth one, and $\langle W, C \rangle$ is built by repeatedly popping one label off the stack, pushing one new label onto the stack and then generating the next word. This kind of model is unable to capture long-distance dependencies. This inability of representing hierarchical structures can be overcome by allowing the state stack to grow without limit and more than one new semantic label to be pushed onto the stack. This is essentially analogous to using stochastic phrase structure rules and extends the class of supported languages from *regular* to *context free*. The Hidden Understanding Model (HUM) model [11], [12], [13], [14] is an early example of such an SCFG model, which uses fully annotated corpora to simplify the parameter estimation problem that is otherwise complex due to the recursive nature of hierarchical parse trees.

A general SCFG model is computationally expensive to train. However, computational tractability issues may be tackled by imposing certain constraints on the SCFG model itself. The hierarchical HMM (HHMM) model [15] constrains the level of hierarchies or the state stack depth to be a bounded depth; it is nevertheless still complex as its state inference takes $O(T^3)$ time, where $T$ is the sequence length. Murphy and Paskin convert an HHMM into a dynamic Bayesian network [16] such that the inference can be done using the junction tree algorithm, which only takes $O(T)$ time empirically, provided that the HHMM hierarchy depth and the number of states at each level of hierarchy are bounded to some relatively small values.

The weakness of nonlexicalized SCFG models such as HUM and HHMM can be avoided by associating a headword to each nonterminal in the parse tree. Examples of lexicalized SCFG models such as the immediate-head parsing model [17] achieved 6 percent reduction in recall error and 5 percent reduction in precision error compared to a general nonlexicalized model when tested on Penn WSJ treebank data [18].

Chelba and Mahajan's structured language model (SLM) [19] does not impose a constraint on the state stack depth, but it does constrain the pushing of at most one new tag (a POStag in this case, not a semantic tag) into the stack. As opposed to the conventional SCFG models where each parser action is only conditioned on the immediately preceding nonterminal tag being expanded, a parser action in the SLM is conditioned on the two previously exposed headwords. However, Chelba and Mahajan's SLM has the limitation that it is not able to capture dependencies between nonheadwords due to its headword percolation rules, for example, *less* and *than* as in *less people join the society this month than last month*, where neither *less* nor *than* are headwords of this phrase.

The flat-concept model is simple and robust to estimate. However, it cannot represent nested structured information. On the other hand, the hierarchical structured models are able to capture long-distance dependencies but require fully annotated treebank data for training, which are difficult to obtain in practice. A hidden vector state (HVS) model [20] has been proposed, which extends the flat-concept HMM model by expanding each state to encode the stack of a push-down automaton. This allows the model to efficiently encode hierarchical context. At the same time, such a model can be trained using only lightly annotated data.

In this paper, we propose a discriminative approach based on parse error measure to train the HVS model. To adjust the HVS model to achieve the minimum parse error, the generalized probabilistic descent (GPD) algorithm [21] was used. Experiments have been conducted in two domains: the travel domain for the semantic parsing task using the DARPA Communicator data and the Air Travel Information Services (ATIS) data and the bioinformatics domain for the information extraction tasks using the GENIA corpus. The results demonstrate modest improvements of the performance of the HVS model using discriminative training. In the travel domain, discriminative training of the HVS model gives a relative error reduction rate of 31 percent in F-measure when compared with MLE on the DARPA Communicator data and 9 percent on the ATIS data. In the bioinformatics domain, a relative error reduction rate of 4 percent in F-measure is achieved on the GENIA corpus.

The rest of the paper is organized as follows: Section 2 surveys related work. In Section 3, we briefly describe the HVS model and how it can be trained in a discriminative way. The experimental setup is discussed in Section 4, and the experimental results are presented in Section 5. Finally, Section 6 concludes the paper and gives future directions.

## 2 Related Work

Discriminative training was initially proposed as an alternative training technique for the speech recognition problem. Historically, the predominant training technique has been MLE. However, it turns out that MLE gives optimal estimates only if the following three conditions are satisfied:

- The model correctly represents the stochastic process.
- An infinite amount of training data are available.
- The true global maximum of the likelihood can be found.

In practice, none of the above conditions is satisfied. This is the motivation for discriminative training. Discriminative training attempts to optimize the correctness of a model by formulating an objective function that in some way penalizes parameter sets that are liable to confuse correct and incorrect answers. Many discriminative training schemes have been proposed based on different objective functions such as MMI, minimum word error (MWE), minimum phone error (MPE), MCE, etc.

From an information-theoretic standpoint, MMI, which is shared between $X$ and $Y$, is the reduction of $X$'s uncertainty due to the knowledge of $Y$. Given the observation $O$, the speech recognizer should choose a word sequence $W$ to make sure that the correct answer has the minimal amount of uncertainty. The IBM speech recognition group was the first to report results with MMI estimation [22]. They obtained an 18 percent lower recognition error rate in a speaker-dependent isolated word recognition system using gradient descent. After that, improvements were reported in [23] using MMI estimation for isolated word recognition. Since gradient descent for MMI estimation does not guarantee convergence and is computationally expensive, an alternative strategy is to use the extended Baum-Welch (EBW) [24]. In [5], a reduction of the string error rate by close to 50 percent was reported using EBW on the TI/NIST connected digit database. Later, lattice-based discriminative training was proposed to optimize the parameters of a continuous-density HMM-based large vocabulary recognition system using MMI criterion [25].

The MCE objective function is designed to directly minimize the errors made by the recognizer on the training set. In [9], experiments were conducted on several key speech recognition tasks, and the MCE method provided a significant reduction of the recognition error rate. In [26], the MWE and MPE objective functions were proposed. The MWE objective function attempts to minimize the number of word-level errors. Instead of maximizing the word accuracy in the MWE approach, the MPE approach maximizes the phone-level accuracy.

All the discriminative methods described above were applied in the speech recognition domain. However, there has been little work in extending them to semantic parsing. To the best of our knowledge, [27] is the only work estimating the probabilities for a neural network statistical parser using the discriminative training criterion. Experiments were conducted to compare the performance of three statistical parsers: one generative, one discriminative, and one generative but using the discriminative training criterion. Results showed that the last parser outperforms the previous two and achieves 90.1 percent in F-measure on the Penn Treebank data.

In this paper, we propose an discriminative training approach based on the minimum parse error for the HVS model. Here, the minimum parse error is similar to MCE and is used to describe the error of semantic parsing on the training set. To adjust the HVS model parameters to achieve the minimum parse error, the GPD algorithm [21] is used to minimize a smoothed function of the parsing error along the steepest direction.

## 3 Methodologies

### 3.1 Hidden Vector State Model

All the parser models described in Section 1 apply constraints in one way or another to the general framework described in (2). In particular, when considering a constrained form of automata where the stack is of finite depth and $\langle W, C\rangle$ is built by repeatedly popping 0 to $n$ labels off the stack, pushing exactly one new label onto the stack, and then generating the next word, it defines the HVS model in which conventional grammar rules are replaced by three probability tables.

Given a word sequence $W$, a concept vector sequence $\mathbf{C}$, and a sequence of stack pop operations $N$, the joint probability of $P(W, \mathbf{C}, N)$ can be decomposed as

$$P(W, \mathbf{C}, N) = \prod_{t=1}^{T} P(n_t|\mathbf{c}_{t-1})\cdot \\ P(c_t[1]|c_t[2\cdots D_t])\cdot P(w_t|\mathbf{c}_t), \tag{3}$$

where $\mathbf{c}_t$, the vector state at word position $t$, is a vector of $D_t$ semantic concept labels (tags), i.e., $\mathbf{c}_t = [c_t[1], c_t[2], ..c_t[D_t]]$, where $c_t[1]$ is the preterminal concept label and $c_t[D_t]$ is the root concept label, $n_t$ is the vector stack shift operation at word position $t$ and take values in the range $0, \dots, D_{t-1}$, and $c_t[1] = c_{w_t}$ is the new preterminal semantic tag assigned to word $w_t$ at word position $t$.

Thus, the HVS model consists of three types of probabilistic move, each move being determined by a discrete probability table:

1. popping semantic labels off the stack—$P(n|\mathbf{c})$,
2. pushing a pre-terminal semantic label onto the stack—$P(c[1]|c[2\cdots D])$, and
3. generating the next word—$P(w|\mathbf{c})$.

This constrained form of automata implements a right-branching parser that has some very convenient properties. It is left to right, and it has complexity $O(TQ^D)$,[1] yet it can still model a hierarchical structure.

### 3.2 Maximum Likelihood Training of the HVS Model

In the HVS-based semantic parser, the purpose of training is to find the HVS parameter set $\lambda = \{\mathbf{C}, N\}$, which will result in the decoder with the lowest possible recognition error rate. This is done by maximizing some objective function $R(\lambda)$. By far, the most commonly used parameter estimation technique is MLE. The objective function typically used in MLE, given the observations $W = \{W_1, W_2, \dots, W_I\}$, is

$$R(\lambda) = f_{ML}(\lambda) = \log\prod_{r=1}^{I} P(W_r, \lambda) = \sum_{r=1}^{I}\log P(W_r, \lambda). \tag{4}$$

MLE attempts to maximize the likelihood of the training data. Thus, we need to compute the $\lambda$ that best explains the data, i.e.,

[1] Here, $T$ is the length of the sequence, $D$ is the maximum stack depth, and $Q$ is the maximum number of concepts (node labels) at each level of the stack.

$$\lambda^* = \arg\max_{\lambda} \sum_{r=1}^{I} \log P(W_r, \lambda). \quad (5)$$

The most obvious quality of MLE is the existence of a reestimation formula $f(\cdot)$ such that if $\hat{\lambda} = f(\lambda)$, then we will have $R(\hat{\lambda}) \geq R(\lambda)$, with equality only when $\lambda$ is a local maximum (or, possibly, a saddle point) of $R(\lambda)$. It can be quickly trained using the globally convergent Baum-Welch algorithm [28].

The reestimation formulas derived are [20]:

$$P^*(n|c') = \frac{\sum_t P(n_t = n, \mathbf{c}_{t-1} = c'|W, \lambda^k)}{\sum_t P(\mathbf{c}_{t-1} = c', W|\lambda^k)}, \quad (6)$$

$$P^*(c[1]|c[2..D]) = \frac{\sum_t P(\mathbf{c}_t, W|\lambda^k)}{\sum_t P(c_t[2..D] = c[2..D]|W, \lambda^k)}, \quad (7)$$

$$P^*(w|\mathbf{c}) = \frac{\sum_t P(\mathbf{c}_t = \mathbf{c}, w_t = w|\lambda^k)}{\sum_t P(\mathbf{c}_t = \mathbf{c}, W|\lambda^k)}. \quad (8)$$

MLE makes a number of assumptions: observation are from a known family of distribution, training data are unlimited, and the global maximum of the likelihood can be found. Unfortunately, in general, none of these assumptions holds. Given that MLE's assumptions are in general not satisfied, it is not guaranteed to produce optimal results. Also, MLE is suboptimal as it only aims to maximize the correct model and ignores the impact of other incorrect competing models. This has led researchers to explore the feasibility and efficiency of using discriminative training.

### 3.3 Discriminative Training of the HVS Model

Normally, MLE is used for generative statistical model training in which only the correct model needs to be updated during training. It is believed that improvement can be achieved by training the generative model based on a discriminative optimization criterion [29] in which the training procedure is designed to maximize the conditional probability of the parses given the sentences in the training corpus. That is, not only the likelihood for the correct model should be increased but also the likelihood for the incorrect models should be decreased.

Given a word sequence $W$, a semantic parser needs to compute the most likely set of embedded concepts $\hat{C}$ by maximizing the following equation:

$$P(C|W) = \frac{P(W|C)P(C)}{P(W)}. \quad (9)$$

For a given $W$, $P(W)$ is a constant, and therefore

$$\hat{C} = \arg\max_{C} P(W|C)P(C), \quad (10)$$

where $P(W|C)$ is called the lexical model, and $P(C)$ is called the semantic model.

Traditionally, semantic parsing models have been trained to have the ML $P(C)$. Although a higher $P(C)$ is often related with a better performance, the correlation is not perfect. Discriminative training based on the minimum parse error is therefore proposed here.

Assuming the most likely semantic parse tree is $\hat{C} = C_j$ and there are altogether $M$ semantic parse hypotheses for a particular sentence $W$, a parse error measure [6], [7], [8] can be defined as

$$d(W) = -\log P(W, C_j) + \log\left[\frac{1}{M-1}\sum_{i,i\neq j} P(W, C_i)^\eta\right]^{\frac{1}{\eta}}, \quad (11)$$

where $\eta$ is a positive number and is used to select competing semantic parses. When $\eta = 1$, the competing semantic parse term is the average of all the competing semantic parse scores. When $\eta \to \infty$, the competing semantic parse term becomes $\max_{i.i\neq j} P(W, C_i)$, which is the score for the top competing semantic parse result. By varying the value of $\eta$, we can take all the competing semantic parses into consideration. $d(W) > 0$ implies a classification error, and $d(W) \leq 0$ implies a correct decision.

The sigmoid function can be used to normalize $d(W)$ in a smooth zero-one range, and the loss function is thus defined as [6]:

$$\ell(W) = \text{sigmoid}(d(W)), \quad (12)$$

where

$$\text{sigmoid}(x) = \frac{1}{1 + e^{-\gamma x}}. \quad (13)$$

Here, $\gamma$ is a constant that controls the slope of the sigmoid function.

For a given training data set consisting of $I$ samples $\{W_1, \ldots, W_I\}$, the empirical probability measure $P_I$ defined on the training data set is a discrete probability measure that assigns equal mass at each sample. The empirical loss, on the other hand, is thus expressed as

$$L_0(\lambda) = \frac{1}{I}\sum_{j=1}^{I}\sum_{i=1}^{M} \ell_i(W_j, \lambda) = \int \ell(W, \lambda) dP_I. \quad (14)$$

The expected loss is defined as

$$L(\lambda) = E_W\{\ell(W, \lambda)\}. \quad (15)$$

It has been shown that the empirical loss defined on the $I$ independent training samples will converge to the expected loss, as the sample size $I$ increases.

The update formula is given by

$$\lambda^{k+1} = \lambda^k - \epsilon^k \nabla \ell(W_i, \lambda^k), \quad (16)$$

where $\epsilon^k$ is the step size.

Using the definition of $\ell(W_i, \lambda^k)$ and after working out the mathematics, the following update formulas can be obtained:

$$(\log P(n|\mathbf{c}'))^* = \log P(n|\mathbf{c}') - \epsilon\gamma\ell(d_i)(1 - \ell(d_i)) \times \left[-I(C_j, n, \mathbf{c}') + \sum_{i,i\neq j} I(C_i, n, \mathbf{c}') \frac{P(W_i, C_i, \lambda)^\eta}{\sum_{i,i\neq j} P(W_i, C_i, \lambda)^\eta}\right], \quad (17)$$

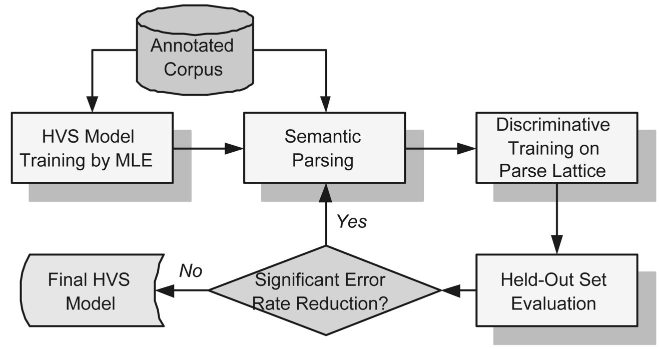

Annotated Corpus
HVS Model Training by MLE
Semantic Parsing
Discriminative Training on Parse Lattice
Held-Out Set Evaluation
Significant Error Rate Reduction?
Yes
No
Final HVS Model

Fig. 2. The discriminative training procedure.

$$\begin{aligned}(\log P(c[1]|c[2..D]))^{*} = {} & \log P(c[1]|c[2..D]) \\ & - \epsilon\gamma\ell(d_i)(1-\ell(d_i)) \times \Bigg[ - I(C_j, c[1], c[2..D]) \\ & + \sum_{i,i\neq j} I(C_i, c[1], c[2..D]) \frac{P(W_i, C_i, \lambda)^{\eta}}{\sum_{i,i\neq j} P(W_i, C_i, \lambda)^{\eta}} \Bigg],\end{aligned} \tag{18}$$

$$\begin{aligned}(\log P(w|\mathbf{c}))^{*} = {} & \log P(w|\mathbf{c}) - \epsilon\gamma\ell(d_i)(1-\ell(d_i)) \\ & \times \Bigg[ -I(C_j, w, \mathbf{c}) + \sum_{i,i\neq j} I(C_i, w, \mathbf{c}) \frac{P(W_i, C_i, \lambda)^{\eta}}{\sum_{i,i\neq j} P(W_i, C_i, \lambda)^{\eta}} \Bigg],\end{aligned} \tag{19}$$

where $I(C_i, n, \mathbf{c}')$ denotes the number of times of the operation of popping up $n$ semantic tags at the current vector state $\mathbf{c}'$ in the $C_i$ parse tree, $I(C_i, c[1], c[2..D])$ denotes the number of times of the operation of pushing the semantic tag $c[1]$ at the current vector state $c[2..D]$ in the $C_i$ parse tree, and $I(C_i, w, \mathbf{c})$ denotes the number of times of emitting the word $w$ at the state $\mathbf{c}$ in the parse tree $C_i$.

A full derivation of the update formulas is given in the Appendix.

### 3.4 Framework of Discriminative Training

Fig. 2 shows the overall discriminative training procedure for the HVS model. The model is originally trained by the MLE criteria. The MLE-trained model is then used to parse the sentences from the lightly annotated training corpus. For each training sentence, the parse results are output as a parse lattice. An example is shown in Fig. 3, where the correct parse path is highlighted with the bold line. For each individual word, the count relating to the correct parse decision is increased, while the count to the incorrect parse decisions is decreased.[2] Thus, the model is trained to separate the correct parse from those incorrect parses. The discriminatively trained model is then used to parse the training sentences again, and the whole training procedure repeats until no significant error reduction is observed in the held-out set.

## 4 Experimental Setup

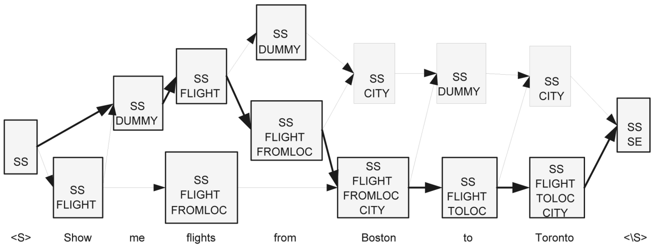


Fig. 3. An example of a semantic parse lattice.

TABLE 1
Statistics of the Three Corpora Used

| Dataset | Training set | Testing set |
|---|---|---|
| DARPA Communicator | 12702 | 1178 |
| ATIS | 4978 | 893 |
| GENIA corpus | 1600 | 250 |

Experiments have been conducted on the three corpora in the two domains: the DARPA Communicator data and the ATIS data in the travel domain and the GENIA corpus in the bioinformatics domain. Table 1 gives the overall statistics of these three corpora. The following describes the experimental setup, as well as the evaluation metric used in the experiments.

### 4.1 DARPA Communicator Data

The DARPA Communicator data [30] are available to the public as open source download. The data contain utterance transcriptions and the semantic parse results from the rule-based Phoenix parser.[3] The DARPA Communicator data were collected in 461 days and consist of 2,211 dialogues or 38,408 utterances in total. From these, 46 days were randomly selected for use as test set data, and the remainder were used for training. After cleaning up the data, the training data consist of 12,702 utterances, while the test set contains 1,178 utterances. Since in our discriminative training framework, the held-out set is needed, we split the test set into two parts with the same size: one is used as the held-out set and the other is used to evaluate the performance of discriminative training.

The abstract annotation used for training and the reference annotation needed for testing were derived by hand correcting the Phoenix parse results. An example of a reference frame is presented as follows:

Show me flights from Boston to New York.
Frame: FLIGHT
Slots: FROMLOC.CITY = Boston
TOLOC.CITY = New York

Performance was then measured in terms of the F-measure on slot/value pairs, which combines the precision and recall with equal weights and is defined as $2 \times \text{Recall} \times \text{Precision}/(\text{Recall} + \text{Precision})$. Recall measures how much relevant information the method has extracted. It is defined as the percentage of correct answers given by the method over the total actual correct answers. Precision measures how much of the information the system extracted is correct. It is defined as the percentage of correct answers given by the method over all the answers extracted by the method.

[2] In our implementation, we use the top $N$ competing parses instead of all the incorrect parses presented in the lattice.

[3] http://communicator.colorado.edu/phoenix.

TABLE 2
Performance Comparison of MLE Training of the HVS Model Using Various Smoothing Techniques

| Smoothing methods | | ATIS | | | DARPA Communicator | | | GENIA | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Stack Shift Operation | Output Probability | Recall | Precision | F-measure | Recall | Precision | F-measure | Recall | Precision | F-measure |
| Linear | Nonlinear | 86.95% | 91.9% | 89.35% | 83.35% | 85.93% | 84.62% | 58.56% | 58.14% | 58.35% |
| | Linear | 86.66% | 91.53% | 89.03% | 83.94% | 85.97% | 84.94% | 57.86% | 57.58% | 57.72% |
| | Witten-Bell | 90.14% | 91.97% | 90.93% | 87.58% | 87.54% | 87.56% | 61.21% | 60.52% | 60.87% |
| NonLinear | NonLinear | 86.24% | 91.22% | 88.66% | 83.35% | 85.77% | 84.54% | 56.49% | 57.81% | 57.14% |
| | Linear | 86.09% | 91.07% | 88.51% | 83.89% | 86.33% | 85.09% | 58.09% | 60.33% | 59.19% |
| | Witten-Bell | 89.61% | 91.82% | 90.7 % | 87.22% | 87.02% | 87.12% | 59.18% | 60.47% | 59.82% |
| Witten-Bell | NonLinear | 87.02% | 92.22% | 89.54% | 82.94% | 86.11% | 84.50% | 61.86% | 57.22% | 59.45% |
| | Linear | 86.91% | 92.14% | 89.45% | 83.26% | 86.40% | 84.80% | 60.75% | 58.27% | 59.48% |
| | Witten-Bell | **90.21%** | **92.04%** | **91.11%** | **87.81%** | **88.13%** | **87.97%** | **61.78%** | **61.16%** | **61.47%** |

## 4.2 ATIS

The ATIS corpus [31] contains air travel information data. The ATIS training set consists of 4,978 utterances selected from the Class-A (context-independent) training data in the ATIS-2 and ATIS-3 corpora, while the ATIS test set contains both the ATIS-3 NOV93 and DEC94 data sets. Abstract semantics for each training utterance were derived semi-automatically from the SQL queries provided in ATIS-3. After the parse results have been generated for the test sets, postprocessing is performed to extract relevant slot/value pairs and convert them into a format compatible with the reference frames.

## 4.3 GENIA Corpus

PPIs referring to the associations of protein molecules are crucial for many biological functions. A major challenge in text mining for biomedicine is automatically extracting PPIs from the vast amount of biomedical literature since most knowledge about them still hides in biomedical publications. We have constructed an information extraction system based on a semantic parser employing the HVS model for PPIs [32].

GENIA [33] is a collection of 2,000 research abstracts selected from the search results of the MEDLINE database using keywords (MESH terms) "*human, blood cells, and transcription factors*." All these abstracts were then split into sentences, and those containing more than two protein names and at least one interaction keyword were kept. Altogether 3,533 sentences were left, and 2,500 sentences were sampled to build our data set.

Abstract annotation were derived manually. An example of such an annotation, together with the PPI information embedded, is presented as follows:

| | |
|---|---|
| Sentences: | CUL-1 was found to interact with SKR-1, SKR-2, SKR-3, SKR-7, SKR-8 and SKR-10 in yeast two-hybrid system. |
| Annotation: | PROTEIN_NAME(INTERACT (PROTEIN_NAME)) |
| PPI : | CUL-1 interact SKR-1<br>CUL-1 interact SKR-2<br>CUL-1 interact SKR-3<br>CUL-1 interact SKR-7<br>CUL-1 interact SKR-8<br>CUL-1 interact SKR-10 |

The evaluation of the experimental results is based on the values of true positive (TP), false positive (FP), and false negative (FN). TP is the number of correctly extracted interactions. $(\mathrm{TP}+\mathrm{FN})$ is the number of all interactions in the test set, and $(\mathrm{TP}+\mathrm{FP})$ is the number of all extracted interactions.

The F-measure is computed using the following formula:

$$\text{F-measure} = \frac{2 \cdot \text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}}, \tag{20}$$

where Recall is defined as $\mathrm{TP}/(\mathrm{TP}+\mathrm{FN})$, and Precision is defined as $\mathrm{TP}/(\mathrm{TP}+\mathrm{FP})$.

# 5 Experimental Results

This section presents the evaluation results in details.

## 5.1 Results Based on MLE Training of the HVS Model

First, experiments were conducted to find the smoothing technique that yielded the best result. For the data in the bioinformatics domain, experiments were further conducted to find the proper size of the training data to generate the best performance since in the GENIA corpus, the size of the training set is not predefined and it is crucial for statistical model training.

### 5.1.1 *Smoothing Techniques*

The performances of the HVS models on the three corpora using different smoothing techniques are listed in Table 2. It can be observed that the best performance was achieved using Witten-Bell for both the stack shift operation and output probabilities.

### 5.1.2 *GENIA Corpus*

To explore the best performance of the HVS model on the GENIA corpus, we need to determine the proper size of the training data. Experiments have been conducted as follows: The corpus was first randomly split into the training set and the test set at the ratio of 9 : 1. The test set consists of 250 sentences, and the remaining 2,250 sentences were used as the training set. The split were conducted 10 times with different training and test data in each round. For each split, 100 sentences were randomly selected from the training set to build an initial HVS model, which was then tested on the test set. Then, another 100 sentences were added from the training set to build a new HVS model, and its performance was analyzed again. This procedure was repeated until all the 2,250 sentences were added into the training set. Fig. 4 illustrates the performance at each stage.

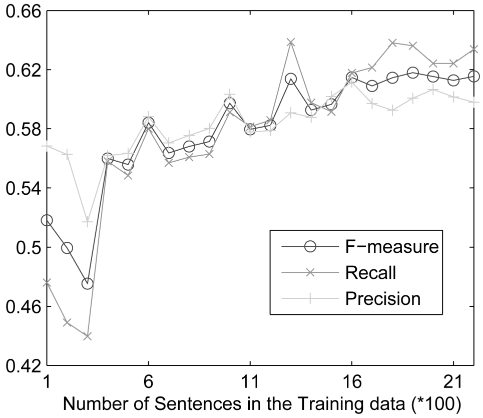


Fig. 4. Performances of the HVS model trained on the increasingly added training data.

It shows that the model performance gradually improves when adding more training data. It saturates when the size of the training data reaches 1,600. At this point, the average F-measure value obtained is 61.47 percent with the balanced recall and precision values. This implies that for the GENIA corpus, 1,600 sentences would be sufficient to train the HVS model. Thus, the training set size of 1,600 is fixed for discriminative training in the subsequent experiments.

## 5.2 Results Based on Discriminative Training of the HVS Model

Experiments have been conducted on the DARPA Communicator data, the ATIS data, and the GENIA corpus by discriminatively training the HVS model based on the update formulas (17), (18), and (19). The following parameters were used: $\gamma = 0.5$, $\eta = 0.1$, and $\epsilon = 0.5$.

### 5.2.1 Optimal Size of Training Data

The size of the training set for discriminative training is highly correlated with the performance of the resulted HVS model. In our experiments, the size of training set is determined by the parameters: $N$, the number of semantic parse hypotheses, and $I$, the number of utterances in the training data. To reveal the relationship between the performance of discriminative training and the size of the training data, experiments were conducted in the following way on the DARPA Communicator data:

- Set $I = 12{,}702$ (the size of the whole training data), 5,000 (almost half of the size of the whole training data), 1,000, 500, 200, 100.
- After fixing the value of $I$, randomly sample $I$ utterances from the whole training set (12,702 utterances) 10 times. At each time $\tau$, a training set $S_\tau$ with a size of $I$ was constructed. We only sampled once for $I = 12{,}702$ and sampled 10 times for all the other values of $I$.
- For each training set $S_\tau$, $\tau = 1, \ldots 10$, discriminative training was conducted when setting $N = 5, 10, 20, 30, 40$.

Table 3 lists the best performance among the experiments for various $I$ and $N$ on the DARPA Communicator data. It can be observed that the best performance of the HVS model using discriminative training is achieved when $I = 1{,}000$ and $N = 30$. It should be noted that a filtering method has been employed to construct the training set by selecting sentences with semantic parse probabilities exceeding a certain threshold. This is to reduce the possible errors introduced to discriminative training since only abstract annotation is provided for each sentence instead of the word-level annotation or the full semantic parse path.

TABLE 3
The Best Performance of 10-Time Sampling Experiments on Various $I$ and $N$ on the DARPA Communicator Data

| $I$<br>$N$ | 100 | 200 | 500 | 1000 | 5000 | 12702 |
|---|---|---|---|---|---|---|
| 5 | 91.39% | 91.45% | 91.1% | 91.15% | 83.70% | 75.60% |
| 10 | 91.01% | 91.29% | 91.37% | 91.31% | 90.71% | 76.28% |
| 20 | 91.13% | 90.87% | 91.38% | 91.55% | 91.58% | 84.62% |
| 30 | 90.99% | 90.9% | 91.3% | **91.68**% | 91.56% | 90.74% |
| 40 | 90.9% | 91.1% | 91.15% | 91.57% | 91.64% | 91.19% |

To examine the performance of the HVS model in each sampling in more detail, Fig. 5 gives the boxplot of the performance of the HVS model in each sampling, showing the variation of the performance of the resulted HVS model as a function of $N$ and $I$ on the DARPA Communicator data. It shows that when $I = 1{,}000$ and $N = 30$, this size of the training set gives the best and balanced performance among all the candidate training sets.

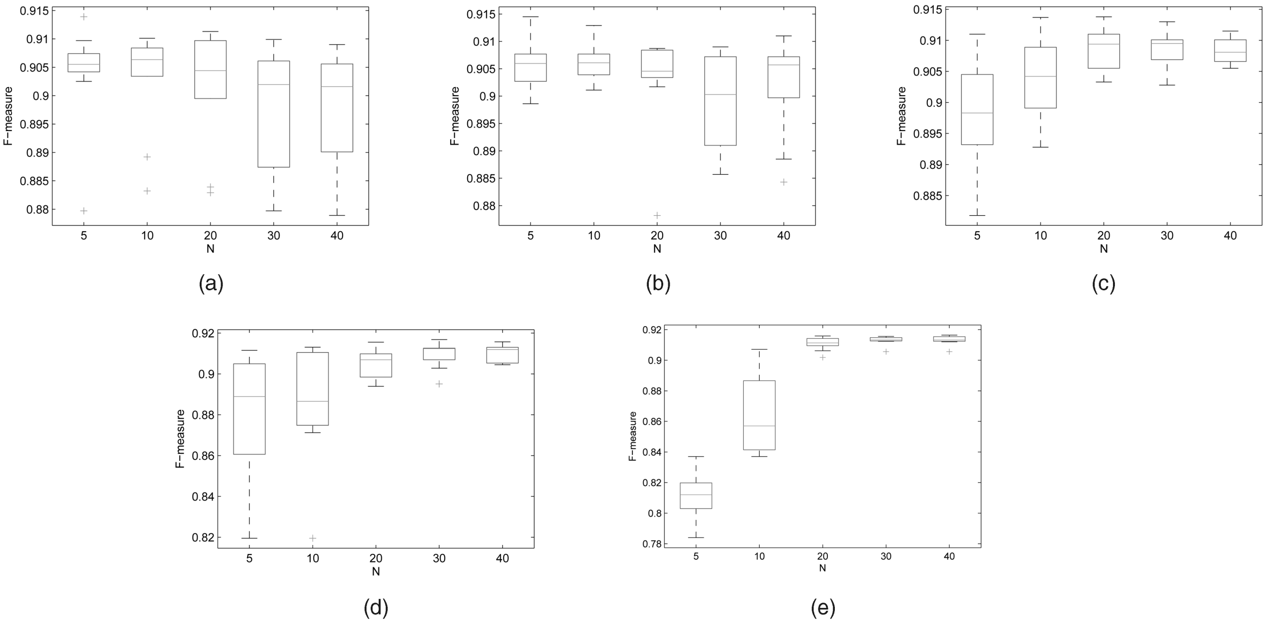
For the ATIS data, experiments were conducted in a similar way. Table 4 lists the best performance among the experiments for various $I$ and $N$ on the ATIS data. It can be observed that the best performance using discriminative training is achieved when $I = 100$ and $N = 5$ or 10.

For the GENIA corpus, the whole training data (1,600 utterances) were split into eight nonoverlapping sets with each set consisting of 200 sentences. The training set size of 200 was chosen empirically, as the performance of the HVS model would degrade if the training set size is set to 100 or 500.

To explore the convergence rate of discriminative training, we further analyzed the experiments on the DARPA Communicator data. Fig. 6 gives the histogram of the iteration numbers on the experiments we have conducted to find the optimal size of the training data. As described above, overall experiments were conducted $4 \times 5 \times 10 + 5 \times 2 + 5 = 215$ times. In Fig. 6, we can see that almost half of the experiments converged before iteration 3. This shows the fast convergence rate of the discriminative training method.

## 5.3 Comparison of Discriminative Training with MLE

The results using MLE and discriminative training are listed in Table 5. For the DARPA Communicator data, $N$ and $I$ were set to 30 and 1,000, respectively, and the discriminatively trained HVS model outperforms the ML-trained HVS model by 4.2 percent, while on the ATIS data, when $N = 5$ and $I = 100$, discriminative training achieves an F-measure of 91.87 percent. For the more complex task on the GENIA corpus, discriminative training improves on the MLE by 2.5 percent, where $N$ and $I$ are set to 5 and 200, respectively.



(a)

(b)

(c)

(d)

(e)

Fig. 5. Boxplot of the performance on the DARPA Communicator data using discriminative training with different $N$ and $I$. (a) $I = 100$. (b) $I = 200$. (c) $I = 500$. (d) $I = 1,000$. (e) $I = 5,000$.

Fig. 7 shows the performance of the HVS model versus the training iterations on the three corpora. It can be observed that discriminative training can quickly achieve the best performance on the HVS model. The best performance on the DARPA Communicator data, the ATIS data, and the GENIA corpus is achieved at iterations 3, 4, and 1, respectively.

## 5.4 Semantic Parsing Based on the Results of the Speech Recognizer Output

The aforementioned experiments conducted in the travel domain used the reference transcriptions derived from the speech utterances as inputs to the semantic parser. That is, it was assumed that the speech recognizer gives 0 percent word error rate. Since the air travel data was originally derived from speech, a more interesting comparison would be conducted by performing semantic parsing based on the results of the speech recognizer output. As we do not have the access to the DARPA Communicator speech data, experiments were only conducted on the ATIS corpus.

For the ATIS corpus, the training data set consists of 4,978 utterances, as mentioned in Section 4.2. The ATIS-3 DEC94 test set was used as our test set. The word error rate given by the speech recognizer built from the HTK toolkit [34] is 2.7 percent. Table 6 shows the results using MLE and discriminative training when performing semantic parsing directly on the speech recognizer output. The discriminatively trained HVS model outperforms the ML trained model by 12 percent.

## 5.5 Discussion

Comparing the experimental results in the travel domain on the DARPA Communicator data and the ATIS data, we found that the discriminative training approach gives the relative improvement, measured in F-measure, of 4.2 percent on the DARPA Communicator data. However, the relative improvement is only 0.8 percent on the ATIS data. One possible reason is that the ATIS data are relatively simple, while the DARPA Communicator data are more complex. Thus, MLE achieves better performance on ATIS than on the DAPRA Communicator data. As a consequence, the possible range of improvement would be smaller for ATIS. Incorporating discriminative training gives a similar performance on both corpora with the F-measure value of 91.78 percent obtained from ATIS and 91.68 percent obtained from the DARPA Communicator data.

TABLE 4
The Best Performance of 10-Time Sampling Experiments on Various $I$ and $N$ on the ATIS Data

| $N$ \ $I$ | 100 | 200 | 500 | 1000 | 4978 |
|---|---|---|---|---|---|
| 5 | **91.87**% | 91.77% | 91.78% | 91.71% | 87.94% |
| 10 | **91.87**% | 91.82% | 91.76% | 91.71% | 88.77% |
| 20 | 91.84% | 91.8% | 91.72% | 91.73% | 89.29% |

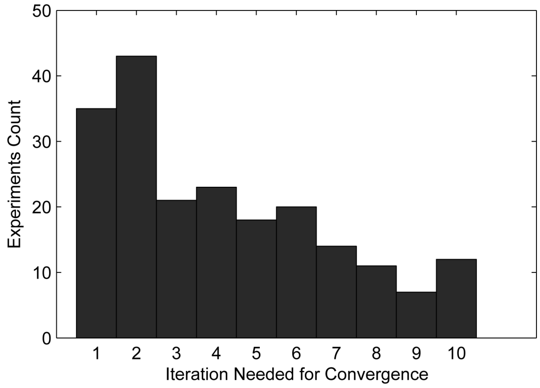
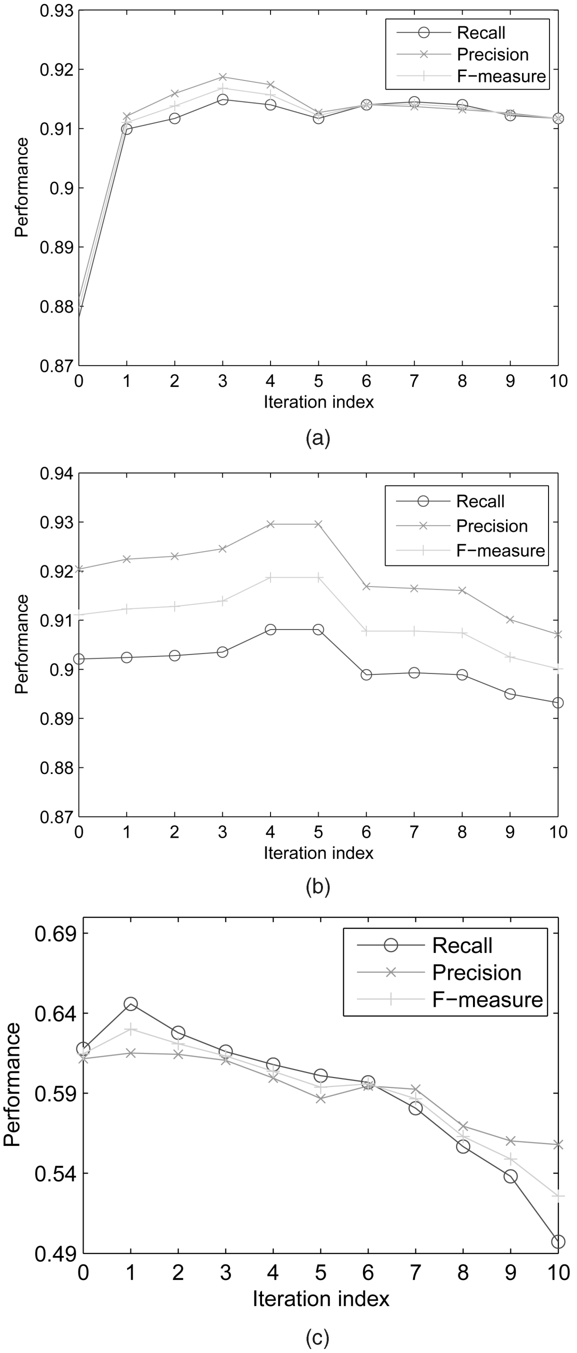


Fig. 6. Histogram of the number of experiments versus the iterations required for convergence on the DARPA Communicator data.

TABLE 5
Performance Comparison of MLE versus Discriminative Training

| Measurement | DARPA Communicator | | ATIS | | GENIA | |
|---|---|---|---|---|---|---|
| | MLE | Discriminative | MLE | Discriminative | MLE | Discriminative |
| Recall | 87.81% | 91.49% | 90.21% | 90.81% | 61.78% | 64.59% |
| Precision | 88.13% | 91.87% | 92.04% | 92.96% | 61.16% | 61.51% |
| F-measure | 87.97% | 91.68% | 91.11% | 91.87% | 61.47% | 63.01% |

Comparing the performance of discriminative training in the travel domain and the bioinformatics domain, the discriminative training approach achieves a relative 2.5 percent improvement on the GENIA corpus compared to the 4.2 percent improvement on the DARPA Communicator Data. The main reason leading to the above result is that the F-measure metric was used in different ways to evaluate the model performance. In the GENIA corpus, the F-measure was used to evaluate the performance of PPI extraction. To correctly extract a PPI, two protein names, one protein interaction keyword, and the hierarchical relations among these three terms must all be identified correctly and simultaneously. That would be only considered as one correct entry in F-measure calculation. Thus, the relative improvement in F-measure in the GENIA corpus is not directly comparable to the improvement in the DARPA Communicator data.



(a)

(b)

(c)

Fig. 7. Performance of the discriminatively trained HVS model versus training iterations. (a) DARPA Communicator $(N = 30, I = 1,000)$. (b) ATIS $(N = 5, I = 100)$. (c) GENIA corpus $(N = 5, I = 200)$.

When viewing the experimental results as relative reductions in the error rate, we found that reductions of about 30.9 percent and 9 percent were achieved in the DARPA Communicator data and the ATIS data, respectively. However, for the GENIA corpus, the relative error reduction rate is only 4 percent. It is therefore important to test the significance levels of the performance improvement. For this purpose, we conducted the statistical test on the three corpora.

For all the three corpora, we constructed 10 models based on the different training data using discriminative training and evaluated their performance on their corresponding test data sets. t-test was employed for the significance test. Table 7 lists the $t$ values for each experiments. The probabilities of the results, assuming the NULL hypothesis, are also shown in the table. To further compare the statistical difference between the performance of MLE and that of discriminative training on the GENIA corpus, we constructed two HVS models using MLE and discriminative training, respectively, and evaluated the two models on 10 different test data sets. Paired Student's t-test was used. Table 7 shows the significance test results. It can be observed that the reduction error rate of 4 percent between MLE and discriminated training for the GENIA corpus is indeed statistically significant.

TABLE 6
Performance Comparison of MLE versus Discriminative Training on the ATIS Speech Recognizer Output

| | MLE | Discriminative | Relative error reduction |
|---|---|---|---|
| Recall | 89.46% | 90.38% | 8.7% |
| Precision | 91.27% | 92.67% | 16.0% |
| F-measure | 90.35% | 91.51% | 12.0% |

TABLE 7
Statistical Test on the Three Corpora, Where $p$ Denotes the Probability of the Result, Assuming the NULL Hypothesis

| Datasets | Hypothesis $H_0$ | $t$ value | $p$ value | Conclusion |
|---|---|---|---|---|
| DARPA Communicator Data | F-measure$_{DIS}$ = F-measure$_{MLE}(87.97\%)$ (One sample T test) | 14.81 | $6.3 * 10^{-8}$ | $t > t_{1-0.005}(9) = 3.69$, reject $H_0$ at the 0.5% signification level. |
| ATIS Data | F-measure$_{DIS}$ = F-measure$_{MLE}(91.1\%)$ (One sample T test) | 4.27 | 0.002 | $t > t_{1-0.005}(9) = 3.69$, reject $H_0$ at the 0.5% signification level |
| GENIA corpus | F-measure$_{DIS}$ = F-measure$_{MLE}(61.47\%)$ (One sample T test) | 4.18 | 0.002 | $t > t_{1-0.005}(9) = 3.69$, reject $H_0$ at the 0.5% signification level |
| | F-measure$_{DIS}$ − F-measure$_{MLE}$ $\geq 1.54\%$ (Paired T test) | 0.04 | 0.484 | $t < t_{1-0.005}(9) = 3.69$, can not reject $\bar{H}_0$ at the 0.5% signification level |

## 6 Conclusion and Future Work

This paper has described how to apply discriminative training to the HVS model on the two different domains: semantic parsing on the DARPA Communicator data and the ATIS data in the travel domain and PPI extraction on the GENIA corpus in the bioinformatics domain. The objective function is based on the minimum parse error criterion. The GDP algorithm is used for estimating the parameters. Experimental results show that the proposed approach exhibits the following advantages:

- *Fast convergence rate*. It can achieve the best performance using only a small amount of training data and converge within three iterations.
- *Improved performance*. It achieves modest improvement compared to the ML training of the HVS model.

In future work, we plan to apply other objective functions to discriminatively train the HVS model. Also, instead of using the $N$-best parse results, we will explore applying discriminative training on the parse lattices directly.

## Appendix

## Derivations of the Update Formulae for Discriminative Training

To calculate the gradient of the loss function$\nabla\ell(W_i, \lambda)$, we break it into two parts and it becomes

$$\nabla\ell = \frac{\partial\ell_i}{\partial d_i}\frac{\partial d(W_i,\lambda)}{\partial\lambda}. \tag{21}$$

Computing the two parts separately, we get

$$\frac{\partial\ell_i}{\partial d_i} = \frac{\gamma}{1+e^{-\gamma d_i}}\frac{e^{-\gamma d_i}}{1+e^{-\gamma d_i}} = \gamma\ell(d_i)(1-\ell(d_i)), \tag{22}$$

$$\begin{aligned}\frac{\partial d(W_i,\lambda)}{\partial\lambda} =& -\frac{1}{P(W_i,C_j,\lambda)}\frac{\partial P(W_i,C_j,\lambda)}{\partial\lambda} + \\ & \frac{\sum_{i,i\neq j} P(W_i,C_i,\lambda)^{\eta-1}\frac{\partial P(W_i,C_i,\lambda)}{\partial\lambda}}{\sum_{i,i\neq j} P(W_i,C_i,\lambda)^{\eta}}.\end{aligned} \tag{23}$$

We can take the partial derivatives with respect to each of the log probability parameters. Let $\lambda = \log P(n|\mathbf{c}')$, we get

$$\begin{aligned}\frac{\partial P(W_i,C_i,\lambda)}{\partial\log P(n|\mathbf{c}')} =& \partial\Big\{P(n|\mathbf{c}')^{I(C_i,n,\mathbf{c}')}\prod_{t=1,n_t\neq n\&\&\mathbf{c}_{t-1}\neq\mathbf{c}'}^{T} \\ & P(n_t|\mathbf{c}_{t-1})P(c_t[1]|c_t[2..D_t])P(w_t|\mathbf{c}_t)\Big\} \\ & /\partial\log P(n|\mathbf{c}') \\ =& I(C_i,n,\mathbf{c}')P(W_i,C_i,\lambda),\end{aligned} \tag{24}$$

where $I(C_i,n,\mathbf{c}')$ denotes the number of times of the operation popping up $n$ semantic tags at the current vector state $\mathbf{c}'$ in the $C_i$ parse tree.

Thus,

$$\begin{aligned}\frac{\partial d(W_i,\lambda)}{\partial\log P(n|\mathbf{c}')} =& -\frac{1}{P(W_i,C_j,\lambda)}I(C_j,n,\mathbf{c}')P(W_i,C_i,\lambda) \\ & + \frac{\sum_{i,i\neq j} P(W_i,C_i,\lambda)^{\eta} I(C_i,n,\mathbf{c}')}{\sum_{i,i\neq j} P(W_i,C_i,\lambda)^{\eta}} \\ =& -I(C_j,n,\mathbf{c}') \\ & + \sum_{i,i\neq j} I(C_i,n,\mathbf{c}')\frac{P(W_i,C_i,\lambda)^{\eta}}{\sum_{i,i\neq j} P(W_i,C_i,\lambda)^{\eta}},\end{aligned}$$

where $C_j$ is the known correct parse tree.

In a similar way, we can get

$$\frac{\partial P(W_i,C_i,\lambda)}{\partial\log P(c[1]|c[2..D])} = I(C_i,c[1],c[2..D])P(W_i,C_i,\lambda), \tag{25}$$

$$\frac{\partial P(W_i,C_i,\lambda)}{\partial\log P(w|\mathbf{c})} = I(C_i,w,\mathbf{c})P(W_i,C_i,\lambda), \tag{26}$$

where $I(C_i,c[1],c[2..D])$ denotes the number of times the operation of pushing the semantic tag $c[1]$ at the current vector state $c[2..D]$ in the $C_i$ parse tree and $I(C_i,w,\mathbf{c})$ denotes the number of times of emitting the $w$ at the state $\mathbf{c}$ in the parse tree $C_i$.

And, finally,

$$\frac{\partial d(W_i,\lambda)}{\partial \log P(c[1]|c[2..D])} = -I(C_j, c[1], c[2..D]) + \sum_{i,i\neq j} I(C_i, c[1], c[2..D]) \frac{P(W_i, C_i, \lambda)^\eta}{\sum_{i,i\neq j} P(W_i, C_i, \lambda)^\eta}, \tag{27}$$

$$\frac{\partial d(W_i,\lambda)}{\partial \log P(w|\mathbf{c})} = -I(C_j, w, \mathbf{c}) + \sum_{i,i\neq j} I(C_i, w, \mathbf{c}) \frac{P(W_i, C_i, \lambda)^\eta}{\sum_{i,i\neq j} P(W_i, C_i, \lambda)^\eta}. \tag{28}$$

Based on the above deductions, we can get the update formulas.

## ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their valuable comments.

## REFERENCES

[1] J. Dowding, R. Moore, F. Andry, and D. Moran, "Interleaving Syntax and Semantics in an Efficient Bottom-Up Parser," *Proc. 32nd Ann. Meeting of the Assoc. for Computational Linguistics,* pp. 110-116, 1994.

[2] W. Ward and S. Issar, "Recent Improvements in the CMU Spoken Language Understanding System," *Proc. ARPA Human Language Technology Workshop (HLT '94),* pp. 213-216, 1994.

[3] M. Collins, "Head-Driven Statistical Models for Natural Language Parsing," PhD dissertation, Univ. of Pennsylvania, 1999.

[4] E. Charniak, "A Maximum Entropy Inspired Parser," *Proc. First Meeting of North Am. Chapter of Assoc. for Computational Linguistics,* pp. 132-139, 2000.

[5] Y. Normandin and S.D. Morgera, "An Improved MMIE Training Algorithm for Speaker-Independent, Small Vocabulary, Continuous Speech Recognition," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing (ICASSP '91),* pp. 537-540, 1991.

[6] B. Juang, W. Chou, and C. Lee, "Statistical and Discriminative Methods for Speech Recognition," *Speech Recognition and Understanding,* Rubio, ed., Springer, NATO ASI Series, 1993.

[7] W. Chou, C. Lee, and B. Juang, "Minimum Error Rate Training Based on N-Best String Models," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing (ICASSP '93),* vol. 2, pp. 652-655, Apr. 1993.

[8] J. Chen and F. Soong, "An N-Best Candidates-Based Discriminative Training for Speech Recognition Applications," *IEEE Trans. Speech and Audio Processing,* vol. 2, pp. 206-216, 1994.

[9] B. Juang, W. Hou, and C. Lee, "Minimum Classification Error Rate Methods for Speech Recognition," *IEEE Trans. Speech and Audio Processing,* vol. 5, pp. 257-265, 1997.

[10] R. Pieraccini, E. Tzoukermann, Z. Gorelov, E. Levin, C.H. Lee, and J.-L. Gauvain, "Progress Report on the Chronus System: ATIS Benchmark Results," *Proc. DARPA Speech and Natural Language Workshop,* pp. 67-71, 1992.

[11] S. Miller, R. Bobrow, R. Ingria, and R. Schwartz, "Hidden Understanding Models of Natural Language," *Proc. 32nd Ann. Meeting of the Assoc. for Computational Linguistics,* pp. 25-32, June 1994.

[12] S. Miller, R. Bobrow, and R. Ingria, "Statistical Language Processing Using Hidden Understanding Models," *Proc. ARPA Human Language Technology Workshop (HLT '94),* pp. 278-282, Mar. 1994.

[13] S. Miller, M. Bates, R. Bobrow, R. Ingria, J. Makhoul, and R. Schwartz, "Recent Progress in Hidden Understanding Models," *Proc. DARPA Speech and Natural Language Workshop,* pp. 276-280, Jan. 1995.

[14] R. Schwartz, S. Miller, D. Stallard, and J. Makhoul, "Language Understanding Using Hidden Understanding Models," *Proc. Fourth Int'l Conf. Spoken Language Processing (ICSLP '96),* Oct. 1996.

[15] S. Fine, Y. Singer, and N. Tishby, "The Hierarchical Hidden Markov Model: Analysis and Applications," *Machine Learning,* vol. 32, pp. 41-62, 1998.

[16] K. Murphy and M. Paskin, "Linear Time Inference in Hierarchical HMMS," *Proc. Neural Information Processing Systems,* Dec. 2001.

[17] E. Charniak, "Immediate-Head Parsing for Language Models," *Proc. 39th Ann. Meeting of the Assoc. for Computational Linguistics,* pp. 124-131, 2001.

[18] J. Henderson, "Inducing History Representations for Broad Coverage Statistical Parsing," *Proc. Joint Meeting of the North Am. Chapter of the Assoc. for Computational Linguistics and the Human Language Technology Conf. (HLT-NAACL '03),* May 2003.

[19] C. Chelba and M. Mahajan, "Information Extraction Using the Structured Language Model," *Proc. Conf. Empirical Methods in Natural Language Processing (EMNLP),* 2001.

[20] Y. He and S. Young, "Semantic Processing Using the Hidden Vector State Model," *Computer Speech and Language,* vol. 19, no. 1, pp. 85-106, 2005.

[21] H.-K. Kuo, E. Fosle-Lussier, H. Jiang, and C. Lee, "Discriminative Training of Language Models for Speech Recognition," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing (ICASSP '02),* vol. 1, pp. 325-328, Apr. 2002.

[22] L. Bahl, P. Brown, P. de Souza, and R. Mercer, "Maximum Mutual Information Estimation of Hidden Markov Model Parameters for Speech Recognition," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing (ICASSP '86),* pp. 49-52, 1986.

[23] P. Brown, "The Acoustic-Modelling Problem in Automatic Speech Recognition," PhD dissertation, Carnegie Mellon Univ., 1987.

[24] P. Gopalakrishnan, D. Kanevsky, A. Nadas, and D. Nahamoo, "An Inequality for Rational Functions with Applications to Some Statistical Estimation Problems," *IEEE Trans. Information Theory,* vol. 37, no. 1, pp. 107-113, 1991.

[25] V. Valtchev, J. Odell, P. Woodland, and S. Young, "Lattice-Based Discriminative Training for Large Vocabulary Speech Recognition," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing (ICASSP '96),* vol. 2, pp. 605-608, May 1996.

[26] D. Povey and P. Woodland, "Minimum Phone Error and I-Smoothing for Improved Discriminative Training," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing (ICASSP '02),* vol. 1, pp. 105-108, Apr. 2002.

[27] J. Henderson, "Discriminative Training of a Neural Network Statistical Parser," *Proc. 42nd Ann. Meeting of the Assoc. for Computational Linguistics,* pp. 95-102, 2004.

[28] L.E. Baum, T. Petrie, G. Soules, and N. Weiss, "A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains," *Ann. Math. Statistics,* vol. 41, pp. 164-171, 1970.

[29] D. Klein and C.D. Manning, "Conditional Structure versus Conditional Estimation in NLP Models," *Proc. Conf. Empirical Methods in Natural Language Processing (ACL '02),* pp. 9-16, 2002.

[30] CUData, *DARPA Communicator Travel Data,* Univ. of Colorado at Boulder, http://communicator.colorado.edu/phoenix, 2004.

[31] D.A. Dahl, M. Bates, M. Brown, W. Fisher, K. Hunicke-Smith, D. Pallett, C. Pao, A. Rudnicky, and E. Shriberg, "Expanding the Scope of the ATIS Task: The ATIS-3 Corpus," *Proc. ARPA Workshop Human Language Technology (HLT '94),* pp. 43-48, 1994.

[32] D. Zhou, Y. He, and C.K. Kwoh, "Extracting Protein-Protein Interactions from the Literature Using the Hidden Vector State Model," *Proc. Int'l Workshop Bioinformatics Research and Applications (IWBRA '06),* pp. 718-725, 2006.

[33] J. Kim, T. Ohta, Y. Tateisi, and J. Tsujii, "GENIA Corpus-Semantically Annotated Corpus for Bio-Textmining," *Bioinformatics,* vol. 9, no. Suppl 1, pp. i180-182, 2003.

[34] HTK, *Hidden Markov Model Toolkit (HTK) 3.2,* Eng. Dept., Cambridge Univ., http://htk.eng.cam.ac.uk, 2002.

**Deyu Zhou** received the BS degree in mathematics and ME degree in computer science from Nanjing University, China, in 2000 and 2003, respectively. He is currently a PhD student in the Informatics Research Centre, University of Reading, United Kingdom. His interests are statistical methods for mining knowledge from texts and biomedical data mining.

**Yulan He** received the BASc (first-class honors) and MEng degrees from Nanyang Technological University, Singapore, in 1997 and 2001, respectively, and the PhD degree from Cambridge University, United Kingdom, in 2004. Between 2004 and 2007, she was an assistant professor with the School of Computer Engineering, Nanyang Technological University. She is currently a lecturer in the Informatics Research Centre, School of Business, University of Reading, United Kingdom. Her current research interests include text and data mining, machine learning, information extraction, natural language processing, and spoken dialogue systems.