A Hybrid Generative/Discriminative Framework to Train a Semantic Parser from an Un-annotated Corpus

Deyu Zhou and Yulan He
Information Research Centre
The University of Reading
Reading, RG6 6BX, UK
d.zhou@rdg.ac.uk, y.he@rdg.ac.uk

Abstract

We propose a hybrid generative/discriminative framework for semantic parsing which combines the hidden vector state (HVS) model and the hidden Markov support vector machines (HM-SVMs). The HVS model is an extension of the basic discrete Markov model in which context is encoded as a stack-oriented state vector. The HM-SVMs combine the advantages of the hidden Markov models and the support vector machines. By employing a modified K-means clustering method, a small set of most representative sentences can be automatically selected from an un-annotated corpus. These sentences together with their abstract annotations are used to train an HVS model which could be subsequently applied on the whole corpus to generate semantic parsing results. The most confident semantic parsing results are selected to generate a fully-annotated corpus which is used to train the HM-SVMs. The proposed framework has been tested on the DARPA Communicator Data. Experimental results show that an improvement over the baseline HVS parser has been observed using the hybrid framework. When compared with the HM-SVMs trained from the fully-annotated corpus, the hybrid framework gave a comparable performance with only a small set of lightly annotated sentences.

1 Introduction

Semantic parsing maps the natural language sentences to complete formal meaning representations. Traditionally, research in the field of semantic parsing can be divided into two categories: rule-based approaches and statistical approaches. Based on hand-crafted semantic grammar rules, rule-based approaches fill slots in semantic frames using word pattern and semantic tokens (Dowding et al., 1994; Ward and Issar, 1994). Such rule-based approaches are typically domain-specific and often fragile. Statistical approaches are generally based on stochastic models. They can be further categorized into three types: generative approaches, discriminative approaches and a hybrid of the two. Generative approaches learn the joint probability model, $P(W, C)$, of input sentence W and its semantic tag sequence C, compute $P(C|W)$ using the Bayes rule, and then take the most probable tag sequence C. The hidden Morkov model (HMM), being a generative model, has been predominantly used in statistical semantic parsing. It models sequential dependencies by treating a semantic parse sequence as a Markov chain, which leads to an efficient dynamic programming formulation for inference and learning. The hidden vector state (HVS) model (He and Young, 2005) is a discrete HMM model in which each HMM state represents the state of a push-down automaton with a finite stack size. State transitions are factored into separate stack pop and push operations constrained to give a tractable search space. The result is a model which is complex enough to capture hierarchical structure but which can be trained automatically from only lightly annotated data. Discriminative approaches directly model posterior probability $P(C|W)$ and

Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008), pages 1113–1120
Manchester, August 2008

learn mappings from W to C. One representative example is support vector machines (SVMs) (Vapnik, 1995). More recently, the hidden Markov support vector machines (HM-SVMs) (Altun et al., 2003) have been proposed which combine the flexibility of kernel methods with the idea of HMMs to predict a label sequence given an input sequence. However, HM-SVMs require full annotated corpora for training which are difficult to obtain in practical applications. On the other hand, the HVS model can be easily trained from only lightly annotated corpora. It is thus interesting to explore the feasibility to combine the advantages of the HVS model and the HM-SVMs.

We propose a hybrid generative/discriminative framework here where a modified K-means clustering method is first applied to select a small set of the most representative sentences automatically from an un-annotated corpus. These sentences together with their abstract annotations are used to train an HVS model which could be subsequently applied on the whole corpus to generate semantic parsing results. The most confident semantic parsing results are selected to generate a fully-annotated corpus which is used to train the HM-SVMs. Experimental results show that an improvement over the baseline HVS parser has been achieved using the hybrid framework. When compared with the HM-SVMs trained from the fully-annotated corpus, the hybrid framework gave a comparable performance with only a small set of lighted annotated sentences.

The rest of this paper is organized as follows. Section 2 reviews other proposed hybrid generative/discriminative frameworks in recent years. Section 3 briefly describes the HVS model and the HM-SVMs followed by the presentation of the proposed hybrid framework. In Section 4, experimental setup and results are discussed. Finally, Section 5 concludes the paper.

2 Related Work

Combination of generative and discriminative models for data classification has recently attracted much interests in the machine learning community. It has been shown theoretically and experimentally in (Jaakkola and Haussler, 1998; Ng and Jordan, 2002; Bouchard and Triggs, 2004) that the hybrid model combines the complementary powers of the both models. The first extensive study on hybrid models were discussed in (Jaakkola and Haussler, 1998) where discriminative features were extracted using generative models and were later used in discriminative models. More recently, the HM-SVMs (Altun et al., 2003) have been proposed which incorporate kernel methods into HMMs to predict a label sequence given an input sequence.

There have also been several studies on exploring the hybrid generative/discriminative frameworks which combine the generative and discriminative models in a pipelined way. One example is the hybrid framework proposed in (Abou-Moustafa et al., 2004) for sequential data classification. The framework employs HMMs to map the variable length sequential data into a fixed size P-dimensional vector that can be classified using any discriminative model. Experiments were conducted on the NIST database for handwritten digits and results showed a better recognition rate than that of standard HMMs. Another example is the hybrid generative/discriminative approach proposed in (Holub et al., 2008) for detecting and classifying object categories in the machine version domain. In this approach, "Fisher Kernels" were used to retain most of the desirable properties of generative methods and a discriminative setting was used to increase the classification performance. Experimental results showed significant performance improvement over the generative counterpart.

3 Methodologies

This section first introduces the hidden vector state (HVS) model and the hidden Markov support vector machines (HM-SVMs) followed by the presentation of the proposed hybrid generative/discriminative framework.

3.1 Hidden Vector State Model

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 the most likely semantic representation can be found through statistical decoding. If assuming that the hidden data take the form of a semantic parse tree C then the model should be a push-down automata which can generate the pair $\langle W, C \rangle$ through some canonical sequence of moves $D = (d_1 \cdots d_T)$. That is,

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

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 non-terminal semantic category label onto the stack;
3. generating the next word.

When considering a constrained form of automata where the stack is 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 Hidden Vector State (HVS) model in which conventional grammar rules are replaced by three probability tables.

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

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

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 $0, \ldots, D_{t-1}$ and $c_t[1] = c_{w_t}$ is the new preterminal semantic tag assigned to word w_t at word position t.

3.2 Hidden Markov Support Vector Machines

To learn a function that assigns to a sequence of words $W = (w_1 \cdots w_T), w_i \in \mathbf{W}, i = 1, \ldots T$ a sequence of semantic tags $C = \mathbf{c}_1 \mathbf{c}_2 \ldots \mathbf{c}_T, \mathbf{c}_i \in \mathbf{C}, i = 1, \ldots T$, a common approach is to determine a discriminant function $F : \mathcal{W} \times \mathcal{C} \rightarrow \mathbb{R}$ that assigns a score to every input $W \in \mathcal{W} := \mathbf{W}^*$ and every semantic tag sequence $\mathbf{C} \in \mathcal{C} := \mathbf{C}^*$, where $\mathbf{W}^*$ denotes the Kleene closure of $\mathbf{W}$. In order to obtain a prediction $f(W) \in \mathcal{C}$, the function is maximized with respect to $f(W) = \underset{C \in \mathcal{C}}{\text{argmax}} F(W, C)$.

In particular, the function $F(W, C)$ is assumed to be linear in some combined feature representation of W and C in HM-SVMs (Altun et al., 2003), $F(W, C) := \langle w, \Phi(W, C) \rangle$. Given a set of training data $(W_i, C_i), i = 1, \ldots N$, the parameters w are adjusted so that the true semantic tag sequence C_i scores higher than all other tag sequences $C \in \mathcal{C}_i := \mathcal{C} \backslash C_i$ with a large margin. To achieve the goal, the following optimization problem is solved instead.

$$\begin{aligned} \min_{\xi_i \in \mathbb{R}, w \in \mathcal{F}} \quad & Cons \sum_i \xi_i + \frac{1}{2} \|w\|^2 \qquad (3) \\ \text{s.t.} \quad & \langle w, \Phi(W, C_i) \rangle - \langle w, \Phi(W, C) \rangle \geq 1 - \xi_i, \\ & \forall i = 1, \ldots N \text{ and } C \in \mathcal{C} \backslash C_i \end{aligned}$$

where ξ_i is non-negative slack variables allowing one to increase the global margin by paying a local penalty on some outlying examples, and $Cons$ dictates the desired trade off between margin size and outliers. To solve the equation 3, the dual of the equation is solved instead. The solution w^* can be written as

$$w^* = \sum_{i=1}^{N} \sum_{C \in \mathcal{C}} \alpha_i(C) \Phi(W_i, C), \qquad (4)$$

where $\alpha_i(C)$ is the Lagrange multiplier of the constraint associated with example i and C_i.

3.3 A Hybrid Generative/Discriminative Framework for Semantic Parsing

The framework of combining the HVS model and the HM-SVMs is illustrated in Figure 1. It consists of three main stages, *Representative Sentences Selection*, *Fully Annotated Corpus Generation*, and *HM-SVM Training and Testing*. Each of them is discussed in details below.

- *Representative Sentences Selection.* Given an un-annotated corpus, the modified K-means clustering algorithm is first employed to select the most representative sentences for annotation. This is to avoid annotating the whole corpus and hopefully the model trained from the subset of the original corpus would still give a similar performance when compared with the model trained from the full corpus. The modified K-means clustering algorithm is described in Figure 3.3.

  Initially, k different sentences are randomly selected as the initial centroids. Then, each sentence s_i in the training data is assigned to one of the k clusters based on the similarity measurement which will be discussed later.

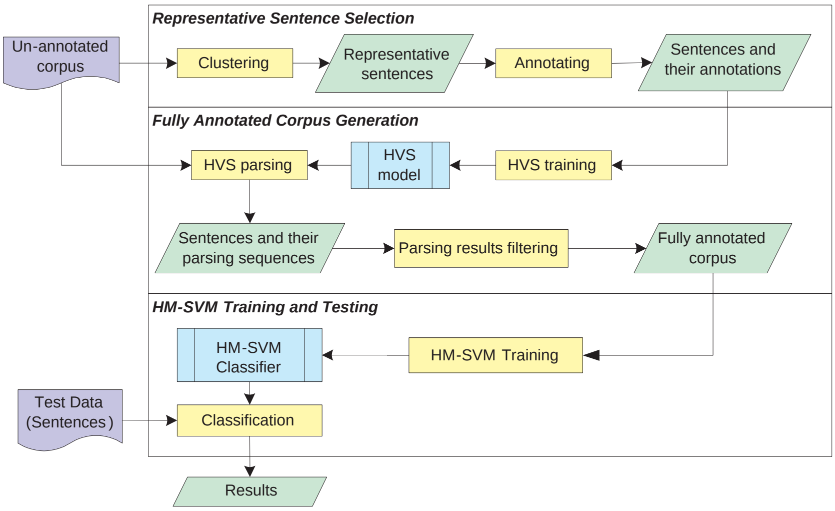


Figure 1: The hybrid generative/discriminative framework for semantic parsing.

After that, the centroids of the k clusters are recalculated. The above process repeats until there are no further changes in the centroids.

The similarity between two sentences is calculated based on sequence alignment. Suppose $\mathbf{a} = a_1 a_2 \cdots a_n$ and $\mathbf{b} = b_1 b_2 \cdots b_m$ are the two word sequences of length of n and m, $Sim(i, j)$ is defined as the score of the optimal alignment between the initial segment from a_1 to a_i of $\mathbf{a}$ and the initial segment from b_1 to b_j of $\mathbf{b}$, where $Sim(i, j)$ is recursively calculated as follows:

$$\begin{aligned} Sim(i,0) &= 0, i = 1, 2, ...n \\ Sim(0,j) &= 0, j = 1, 2, ...m \\ Sim(i,j) &= \max \begin{cases} 0, \\ Sim(i-1, j-1) + s(a_i, b_j), \\ Sim(i-1, j) + s(a_i, '-'), \\ Sim(i, j-1) + s('-', b_j) \end{cases} \end{aligned}$$

Here $s(a_i, b_j)$ is the score of aligning a_i with b_j and is defined as:

$$s(a_i, b_j) = \log \left[\frac{p(a_i, b_j)}{p(a_i) \times p(b_j)} \right] \quad (5)$$

where, $p(a_i)$ denotes the occurrence probability of the word a_i and $p(a_i, b_j)$ denotes the probability that a_i and b_j appear at the same position in two aligned sequences.

To ensure that content words containing key information are weighted more heavily than the less relevant words such as function words, a score matrix can then be built and dynamic programming is used to find the largest score between two sequences. The distance between the two sentences is defined as the negation of the similarity between them.

After generating k clusters, the centroid in each of the clusters is selected as the representative sentence for annotation. This results in an exactly k sentences being selected. There are however two ways to construct the annotated corpus depending on the neighborhood threshold value d. When $d = 1$, the annotated corpus only contains k sentences. When $d < 1$, both the centroid and some of its nearest neighboring sentences in each cluster will receive the same abstract annotation. Thus, the annotated corpus will contain more than k sentences. It has to be noted that in both cases, only k sentences (centroids) need to be annotated.

- *Fully Annotated Corpus Generation.* An HVS model is trained from the annotated corpus constructed in the previous stage which is then applied to the original un-annotated corpus to generate a semantic tag sequence for each sentence. The generated semantic tag sequences essentially define the explicit word/tag alignments which could serve as the full annotation required by HM-SVMs training. However, the HVS model does not guarantee to parse the sentences with 100% accuracy. Based on a user-defined parse probabil-

```
Input:    A set of sentences $S = \{s_i, i = 1, \ldots, N\}$, a distance threshold $\alpha$, a neighborhood threshold $d$
Output:   A set of representative sentences $R = \{r_j, j = 1, \ldots, M\}$, and a set of the centroids of the
          generated clusters $Cent$
Algorithm:
  1.      For each $s_i \in S$, set $Flag_i = 1$. Initialize $R$ and $Cent$ to be empty sets.
  2.      Select sentences from $S$ with $Flag$ equal to 1 , then reconstruct $\hat{S} = \{s_j | Flag_j = 1, j =
          1, \ldots, \hat{N}\}$, $\hat{N}$ is the number of sentences with $Flag$ equal to 1 in $S$.
  3.      Randomly select $k$ different sentences $c_k$ from $\hat{S}$, the default value of $k$ is 1000. Construct $k$
          clusters $C = \{c_l\}, l = 1, \ldots, k$. Set NumOfFlag = $\|\hat{S}\|$
  4.      Loop for each sentences $s_i \in \hat{S}$
            Loop for each cluster $c_l$
              Calculate the distance between $s_i$ and the centroid of $c_l$. $Dist_{il}$ = Distance($s_i, c_l$).
              If $Dist_{il} < \alpha$, then $c_l = c_l \bigcup s_i$, set $Flag_i$ = 0, ExitLoop.
            EndLoop
            If $Flag_i \neq 0$, then find the cluster $l' = \underset{l}{\text{argmin}}\{Dist_{il}, l = 1, \ldots, k\}$, $c_{l'} = c_{l'} \bigcup s_i$
          EndLoop
          If $\|\{s_i | s_i \in \hat{S}, Flag_i = 1\}\|$ <NumOfFlag, then set NumOfFlag = $\|\{s_i | s_i \in \hat{S}, Flag_i = 1\}\|$,
          go to Step 4.
          $Cent = Cent \bigcup Cent_l, l = 1, \ldots, k, \|c_l\| \neq 0$.
  5.      If NumOfFlag $> 0$ then Go to step 2.
          Else
            $R = R \bigcup Cent$.
            Construct $\|Cent\|$ clusters $\hat{C} = \{c_l\}, l = 1, \ldots, \|Cent\|$.
            Loop for each sentences $s_i \in S$
              Find the cluster $l' = \underset{l}{\text{argmin}}\{Dist_{il}, l = 1, \ldots, \|Cent\|\}$, $c'_l = c'_l \bigcup s_i$
              If $Dist_{il'} < d$, then $R = R \bigcup s_i$
            EndLoop
          EndIf
```

Figure 2: A modified K-means clustering method.

ity threshold, the most confident parse results are selected for the construction of the fully annotated corpus.

- *HM-SVMs Training and Testing.* Given the fully annotated corpus generated in the previous stage, the HM-SVMs can then be trained which could later be used to derive the semantic tag sequences for the test data.

4 Experiment

Experiments have been conducted on the DARPA Communicator data (CUData, 2004) which 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[1]. The DARPA Communicator data were collected in 461 days. 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 set consist of 12702 utterances while the test set contains 1178 utterances.

The abstract annotation used for training and the reference annotation needed for testing were derived by hand correcting the Phoenix parse results. For example, for the sentence "Show me flights from Boston to New York", the abstract annotation would be
`FLIGHT(FROMLOC(CITY) TOLOC(CITY))`. Such an annotation need only list a set of valid semantic concepts and the dominance relationships between them without considering the actual realized concept sequence or attempting to identify explicit word/concept pairs. Thus, it avoids the need for expensive tree-bank style annotations.

To evaluate the performance of the model, a reference frame structure was derived for every test set sentence consisting of slot/value pairs. An example of a reference frame is:

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 F -measure on slot/value pairs, which combines

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

Table 1: Feature templates used in HM-SVMs. w_i is the current word, and $w_1, \ldots, w_n$ is the entire sentence.

Current word	w_i
Previous word	w_{i-1}
Word two back	w_{i-2}
Next word	w_{i+1}
Word two ahead	w_{i+2}
Bigram features	w_{i-2}, w_{i-1} w_{i-1}, w_i w_i, w_{i+1} w_{i+1}, w_{i+2}

Table 2: The number of representative sentences vs the different settings of α and d.

α d	0.5	0.6	0.7	0.8	0.9
1	350	663	1068	1743	2763
0.6	6878	7596	9810	9640	11872

the precision (P) and recall (R) values with equal weight and is defined as $F = (P + R)/2PR$.

In all the subsequent experiments, the open source SVM^{hmm}(Tsochantaridis et al., 2005) [2] has been used to train and test the HM-SVMs. The features used in the HM-SVMs are listed in Table 1.

4.1 Comparison between HVS and HM-SVMs

In the modified K-means clustering algorithm described in Figure 3.3, the number of representative sentences depends on both the distance threshold α and the neighborhood threshold d. Table 2 shows the number of representative sentences obtained by varying α and d.

First, a set of experiments were conducted to compare the performance of the HVS model with the HM-SVMs only without incorporating the hybrid framework. Based on the different values of d and α, we constructed different sets of the annotated corpus. For example, when $\alpha = 0.5$, there are 350 clusters generated from a total of 12702 sentences in the un-annotated corpus. The centroid from each of the cluster is then selected for annotation. These 350 sentences were annotated with abstract annotation for the HVS model training. And they were also fully annotated by providing word-level annotations for HM-SVMs training.

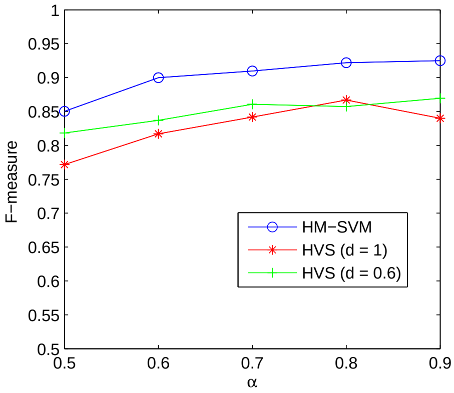


Figure 3: Comparisons of the performance of HVS and HM-SVMs on different α.

Since only abstract annotations need to be provided for the HVS model training, it is possible to automatically enlarge the annotated corpus by including some of the neighboring sentences if the same annotation of the centroid can be assigned to them. This is controlled by varying the neighborhood threshold d. If $d = 1$, then the annotated corpus only contains 350 sentences with their abstract annotations. If varying d to 0.6, then for each cluster, some of the neighboring sentences also receive the same abstract annotation as that of the centroid, thus the annotated corpus is enlarged to contain 6878 sentences.

The performance comparison of the HVS and the HM-SVMs is shown in Figure 3. It can be observed that in general, HM-SVMs outperforms HVS. This is not surprising as HM-SVMs was trained from the fully annotated corpus. The HVS model based on $d = 0.6$ achieved better performance over the one based on $d = 1$ since the enlarged annotated corpus was used for training. The best performance given by HM-SVMs is 92.5% of F-measure when $\alpha = 0.9$ and 2793 annotated sentences were used for training, while the HVS model gave an F-measure of 86.9%.

Though HM-SVMs outperforms HVS by 5.5%, it should be noted that the time consumed for preparing the fully annotated corpus for HM-SVMs is far more than the time spent for abstract annotation for HVS as shown in Figure 4. When $\alpha = 0.5$, annotating 350 sentences with the explicit word/semantic tag mappings took about 17.5 hours while abstract annotation only took about 3 hours. When $\alpha = 0.9$, the time spent on fully annotating 2763 sentences is almost six times that of abstract annotation.

[2] http://www.cs.cornell.edu/People/tj/svm_light/svm_hmm.html

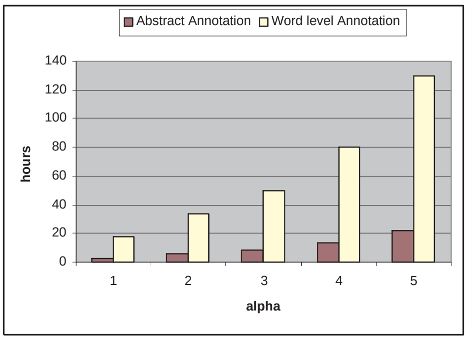


Figure 4: Comparison of time Consuming in preparing training data for the HVS model and the HM-SVMs.

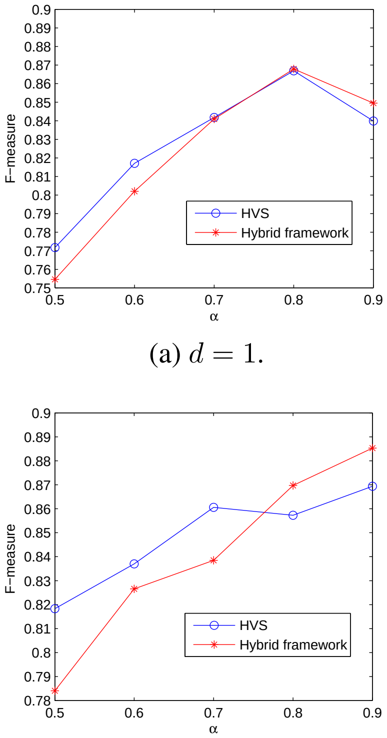
4.2 Comparison between HVS and the Hybrid Framework with Clustering

Figure 5 shows the performance comparison between the HVS model and the hybrid framework by varying α. It can be observed that when the size of the annotated corpus is small, the HVS model outperforms the hybrid framework. However, with increased number of annotated sentences, the hybrid framework achieves the better performance. For both the HVS model and the hybrid framework, improved performance is observed by training the model/framework from the augmented annotated corpus with the neighboring sentences automatically added in (cf. Figure 5(a) and (b)). We notice from Figure 5(a) that when the number of annotated sentences increases from 1743 to 2763, the performances of both the HVS model and the hybrid framework decrease. By analyzing the clustering results generated from the modified K-means algorithm, we found that some of the clusters formed actually contain those rare sentences and they represent the outliers of the original training set. This therefore leads to the decreased performance of the HVS model and the hybrid framework.

With only 2763 annotated sentences, the hybrid framework trained under $d = 0.6$ achieves 88.5% in F-measure which results in a relative error reduction of 12% when compared with the HVS model where the F-measure value of 87.0% was obtained.

4.3 Comparison between HVS and the Hybrid Framework without Clustering

Experiments have also been conducted to compare the performance of the HVS model and the hybrid framework without employing the modified K-means clustering algorithm to automatically select the most representative sentences for annotation. That is, the whole corpus of 12702 sentences were provided with abstract annotations. Both the HVS model and the hybrid framework were trained from the training set which was formed by randomly selecting the annotated sentences from the original corpus. Figure 6 illustrates the performance of the HVS model and the hybrid framework versus the varying sizes of the training data. Here 10-fold cross validation was performed and the F-measure value at each point of the curve in Figure 6 was calculated by averaging the performance of the 10 experiments each time with different training set of the same size.



(a) $d = 1$.

(b) $d = 0.6$.

Figure 5: Comparison of the performance of the HVS model and the hybrid framework

It can be observed that the performance of both the HVS model and the proposed hybrid framework increases with the increased size of the training data. The hybrid framework outperforms the HVS model when the size of the training data is beyond 6000. The improvement is more substantial by incorporating more training data.

The best performance achieved by the HVS model and the proposed hybrid framework is listed

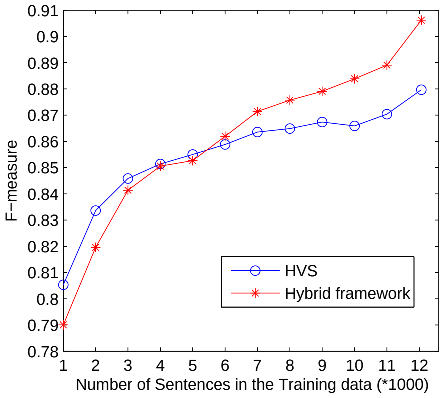


Figure 6: Performance of the HVS model and the hybrid framework vs the size of the training data.

in Table 3. It can be observed that the hybrid framework gives the relative error reduction of 22% when compared with the performance of the HVS model where only 87.97% was achieved.

Table 3: Performance comparison between HVS and the hybrid framework.

Measurement	HVS	Hybrid Framework
Recall	87.81%	90.99 %
Precision	88.13%	90.25%
F-measure	87.97%	90.62%

It should also be noted that the best performance of 87.97% in F-measure was obtained when the HVS model was trained on the whole annotated corpus of 12702 sentences. By employing the clustering method to select the most representative sentences, the hybrid framework trained with less than one fourth of the original training data (2763 annotated sentences) achieves 88.53% in F-measure, which is better than that of the HVS model.

5 Conclusions

This paper presented a hybrid framework by combining the HVS model and the HM-SVMs for semantic parsing. Experimental results show that 22% relative error reduction in F-measure was obtained using the hybrid framework on the DARPA Communicator Data when compared with the performance of the HVS model. Furthermore, employing the modified K-means clustering algorithm to automatically select the most representative sentences for annotation greatly reduces the human effort for annotation. With only 2763 annotated sentences, the hybrid framework gives the better performance compared with the HVS model trained on the full 12702 annotated sentences. Also, the hybrid framework gives the comparable performance with that of the HM-SVMs but without the use of the expensive word-level annotations.

References

Abou-Moustafa, K.T., C.Y. Suen, and M. Cheriet. 2004. A generative-discriminative hybrid for sequential data classification. In *Acoustics, Speech, and Signal Processing, 2004 (ICASSP '04)*, volume 5, pages 805–808.

Altun, Y., I. Tsochantaridis, and T. Hofmann. 2003. Hidden markov support vector machines. In *International Conference in Machine Learning*, pages 3–10.

Bouchard, Guillaume and Bill Triggs. 2004. The tradeoff between generative and discriminative classifiers. In *Proc. of COMPSTAT 2004*, pages 721–728.

CUData. 2004. Darpa communicator travel data. university of colorado at boulder. Avaiable from http://communicator.colorado.edu/phoenix.

Dowding, J., R. Moore, F. Andry, and D. Moran. 1994. Interleaving syntax and semantics in an efficient bottom-up parser. In *Proc. of the 32th Annual Meeting of the Association for Computational Linguistics*, pages 110–116, Las Cruces, New Mexico, USA.

He, Yulan and Steve Young. 2005. Semantic processing using the hidden vector state model. *Computer Speech and Language*, 19(1):85–106.

Holub, Alex D., Max Welling, and Pietro Perona1. 2008. Hybrid generative-discriminative visual categorization. *International Journal of Computer Vision*, 77:239–258.

Jaakkola, T. and D. Haussler. 1998. Exploiting generative models in discriminative classifiers. In *Proc. of Advances in Neural Information Processing 11*.

Ng, A. and M. Jordan. 2002. On generative vs. discriminative classifiers: A comparison of logistic regression and naive bayes. In *Proc. of Advances in Neural Information Processing 15*, pages 841–848.

Tsochantaridis, Ioannis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. 2005. Large margin methods for structured and interdependent output variables. *J. Mach. Learn. Res.*, 6:1453–1484.

Vapnik, Vladimir N. 1995. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA.

Ward, W. and S. Issar. 1994. Recent improvements in the cmu spoken language understanding system. In *Proc. of the workshop on Human Language Technology*, pages 213–216, Plainsboro, New Jerey, USA.