2011 IEEE International Workshop on Machine Learning for Signal Processing
September 18-21, 2011, Beijing, China

PERSIAN LANGUAGE UNDERSTANDING USING A TWO-STEP EXTENDED HIDDEN VECTOR STATE PARSER

Fattaneh Jabbari, Hossein Sameti, and Mohammad Hadi Bokaei

Speech Processing Lab, Department of Computer Engineering, Sharif University of Technology
Tehran, Iran
fjabbari@ce.sharif.edu, sameti@sharif.edu, bokaei@ce.sharif.edu

ABSTRACT

The key element of a spoken dialogue system is a spoken language understanding (SLU) unit. Hidden Vector State (HVS) is one of the most popular statistical approaches employed to implement the SLU unit. This paper presents a two-step approach for Persian language understanding. First, a goal detector is used to identify the main goal of the input utterance. Second, after restricting the search space for semantic tagging, an extended hidden vector state (EHVS) parser is used to extract the remaining semantics in each subspace. This will mainly improve the performance of semantic tagger, while reducing the model complexity and training time. Moreover, the need for large amount of data will be reduced importantly due to lowering of data sparseness. Experiments are reported on a Persian corpus, the University Information Kiosk corpus. The experimental results show the effectiveness of the proposed approach compared to HVS and EHVS.

Index Terms— Spoken language understanding, goal detector, two-step approach, semantic tagging

1. INTRODUCTION

The key element of a spoken dialogue system is a spoken language understanding unit. The main purpose of this unit is to transform the input utterances into semantic information. Conventionally, this task was done by means of hand-crafted semantic grammar rules. Such approaches were fragile, laborious, expensive, and needed a lot of expertise. Recently, data-driven (statistical) approaches have been proposed for this problem. An early statistical model was finite state tagger applied on AT & T's CHRONUS system [1]. It models an HMM-like process in which semantics correspond to hidden states and words correspond to states outputs. This model was simple but unable to capture long range dependencies since it was flat concept. BBN's Hidden Understanding Model (HUM) [2] and hierarchical HMMs [3] solve this problem by Probabilistic Context Free Grammars (PCFG), which makes finite state networks recursive, i.e., each state can produce either a word or a subnetwork, but these models require fully annotated Treebank data and tractability issues may raise. A model was proposed in [4] to make a tradeoff between these two extremes, the flat concept and fully recursive model. It is trainable using a lightly annotated data without the need to Treebank. Such an annotation does not consider the actual realized semantic sequence or explicit word/semantic pairs. It only needs a list of valid semantics where the dominance relationship between them is shown with parenthesizing.

HVS model extends discrete Markov model in which each state is a snapshot of push-down automaton. Thus, it can capture the hierarchical semantic relationships. Since the stack operations could be constrained, tractability issues would be avoided. Afterwards, [5] uses three techniques to improve the performance of the HVS model. The main drawback of statistical approaches is the need for large amount of data. The approach proposed in this paper alleviates this problem. This semantic tagger is based on two steps: at the first step, a goal detector is used to detect the category of user utterance. Then at the second step, an EHVS parser is used to label the utterances according to that detected category. This results in a great improvement in EHVS performance because it greatly reduces the search space resulting in data sparseness reduction.

The remainder of this paper is organized as follows. In Section 2 HVS parser is introduced and its mathematical framework is discussed. Section 3 describes the EHVS model. Two-step EHVS parser is then introduced in Section 4. The University Information Kiosk corpus is represented in Section 5. Section 6 reports the experimental results by three models HVS, EHVS, and two-step EHVS. Finally, Section 7 concludes the paper.

2. HIDDEN VECTOR STATE PARSER

The HVS parser is a statistical parser and is considered as an extension of the discrete Markov model. Consider the semantic parse tree shown in Figure 1. It contains a Persian sentence and its semantic parse tree. The English translation comes below the Persian writing. The bottom of the picture shows the vector states related to every word. By starting from each pre-terminal node, i.e. the observed words, and traverse the tree to reach the root, semantic information about each word could be achieved. Considering the

978-1-4577-1623-2/11/$26.00

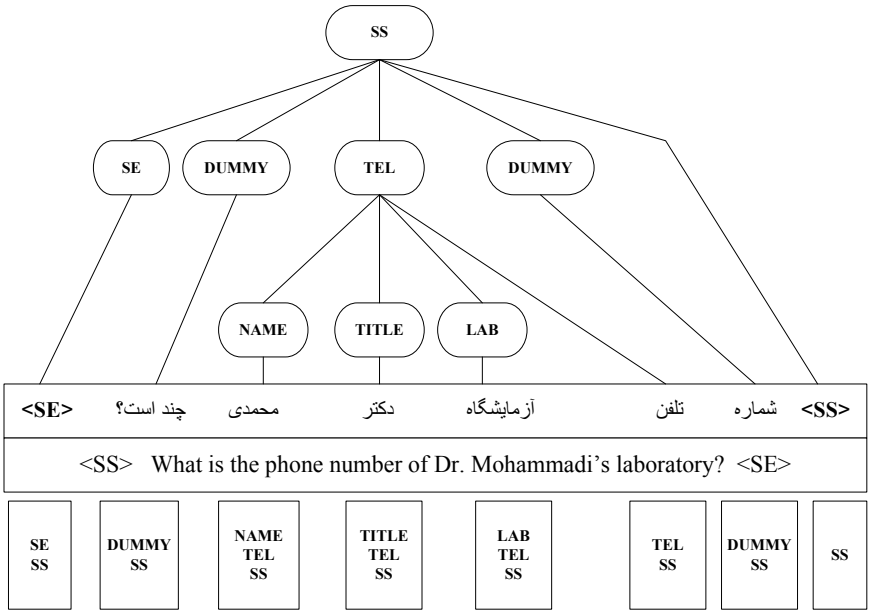

Figure 1.Semantic parse tree and equivalent vector states

semantic sequence of each word as a vector, the semantic tree will be equivalent to a sequence of semantic vectors. Viewing these vector states as hidden variables, the parse tree is equal to a first order vector state Markov model, namely HVS. Each vector state is like a snapshot of a push-down automaton and the transition between different snapshots can be viewed as stack shift operations. Each transition consists of three operations: popping from zero to n concepts from the stack, pushing a new concept to the stack, and generating the next word.

The goal of the HVS is to find the most likely vector state sequence C* given the word sequence W:

$$C^* = \text{argmax}_c \; P(C|W) = \text{argmax}_c \; P(W|C) \; . \; P(C) \qquad (1)$$

where P(C) is semantic model and models the pop and push operations. This is formulated as:

$$P(C) = \prod_{t=1}^{T} P(pop_t | \; c_{t-1}) \; . \; P(c_t[1] \; | c_t[2,...,n]) \qquad (2)$$

where pop_t is the shift operation and takes values in range 0 to n. To reduce the state space to a manageable size, the value of n should be restricted. In many practical applications n is considered to be four. c_t represents the vector state at time t, $c_t[1]$ is the pre-terminal concept for word w_t and $c_t[n]$ is the root element. Additionally, P(W|C) is the lexical model and corresponds to the word generation process. It is given by Equation (3):

$$P(W|C) = \prod_{t=1}^{T} P(w_t|c_t) \qquad (3)$$

where $P(w_t|c_t)$ is the conditional probability of observing w_t given the semantic vector state c_t at time t. More details about HVS are in [4].

3. EXTENDED HIDDEN VECTOR STSTE PARSER

The EHVS has three properties more than the HVS. These added features are: negative examples, left branching, and input parameterization.

3.1. Negative examples

Positive example is a pair of word w and concept c, where w can be generated by concept vector containing c. Similarly, negative example is the negation of positive example, i.e. it is the pair of word w and concept c where w cannot be generated by vector state containing semantic c. Negative examples are not as much informative as positive ones and are used to initialize the lexicalization model.

To improve performance using this feature, the lexical model is initialized uniformly. Then the probability of word observation given vector state c at time t will be penalized according to the negative examples:

$$P(w|c) = \frac{P(w,c)}{\sum_{\bar{w} \in V} P(\bar{w}|c)} \qquad \forall w \in V \qquad (4)$$

where V is the word lexicon, and P(w,c) will be computed by Equation (5).

$$P(w,c) = \begin{cases} \varepsilon & \text{if } w \text{ and } c[1] \text{ are negative examples} \\ \dfrac{1}{|V|} & \text{otherwise} \end{cases} \quad (5)$$

3.2. Left branching

In the HVS model, the transition between vector states at each time contains pushing one concept onto the stack. This leads parse tree to be right-branching. In some languages which are left-branching or both left and right-branching, this will cause parsing problems. To solve this problem, the pushing operation should be changed. In the EHVS model this operation is changed in such a way that, each push operation consists of pushing zero, one, or two concepts onto the stack [6]. Then, not only right-branching trees could be generated, but also left-branching and their combinations could be generated.

To manage the number of new terminals to be pushed, a new hidden variable $push_t$ is introduced. Then the semantic model changes to:

$$P(C) = \prod_{t=1}^{T} P(pop_t|c_{t-1}).P(push_t|c_{t-1}).$$

$$\begin{cases} 1 & push_t=0 \\ P(c_t[1]|c_t[2...4]) & push_t=1 \\ P(c_t[1]|c_t[2...4]).P(c_t[2]|c_t[3,4]) & push_t=2 \end{cases} \quad (6)$$

3.3. Input parameterization

The parser input in the HVS model is the word sequence. However, some other useful information can be used to improve the semantic tagger performance. For example, linguistic knowledge such as word stems, morphological tags, and part-of-speech tags can improve the model performance.

So, the lexical model of HVS could be generalized using input feature vector instead of a single word:

$$P(F|C)=\prod_{t=1}^{T} P(f_t|c_t)=\prod_{t=1}^{T} P(f_t[1],f_t[2],...,f_t[M]|c_t) \quad (7)$$

where $f_t = [f_t[1], f_t[2], \ldots, f_t[M]]$ is the feature vector of length M for word w at time t. To obtain robustness, some approximations are used [7].

4. TWO-STEP EHVS PARSER

In this paper, we divide the semantic tagging problem into two subproblems. The first subproblem is determining the main goal of an utterance, which is like a topic classification problem. This can be solved by pattern recognition techniques. The second subproblem is using semantic tagger to tag the utterance in the subspace of the determined goal. The overall process of the two-step EHVS parser is shown in Figure 2.

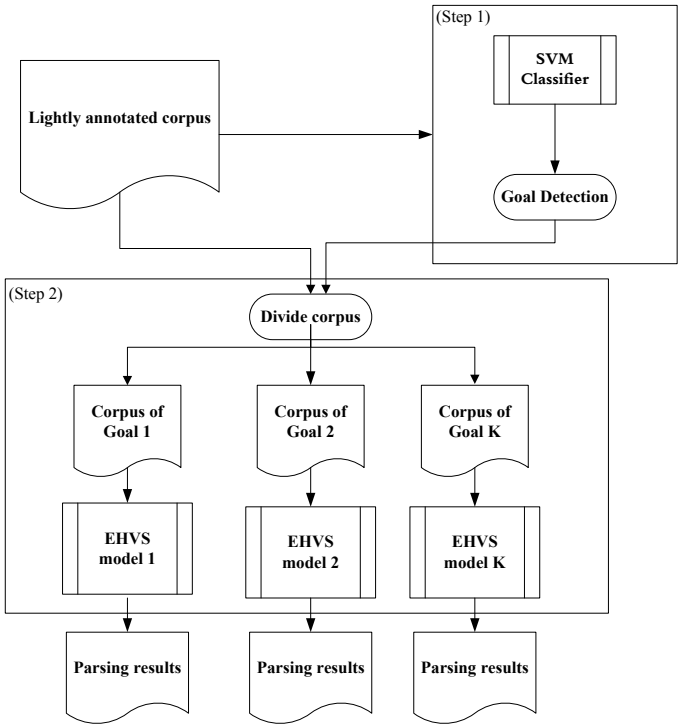

Figure 2.Overall process of the two-step EHVS parser

4.1. Goal detector

The main object in this phase is to identify the goal of the input utterances in order to partition the space of semantics into some smaller subspaces. If there are K goals in a domain, this phase will result in K different semantic labeling tasks. Solving these labeling tasks is easier than the main one since the search spaces are mainly smaller and more alike for each goal.

Many statistical pattern recognition approaches such as maximum a-posteriori (MAP) method [8], artificial neural networks (ANN) [9], Naive Bayes [10], N-Gram [10], and support vector machines (SVM) [10] are applied to solve this problem. In this experiment, SVM is used to extract the utterance goal. SVM is a binary classifier, in which given a set of training data and their corresponding targets, it finds a separating hyper-plane with the maximal margin. This hyper-plane separates the positive instances from the negative ones. The kernel function can be selected according to the dimensionality of feature space and data size. One may not need to map data into a higher dimensional space if the number of features is large, i.e. in such cases, nonlinear mapping will not improve the performance. In this paper linear kernel is used, since the number of features are large enough compared to the dataset size.

To use SVM classifier in a multi-class problem, two approaches exist: one-against-one and one-against-all. In the former case, each pair of classes is separated by trained classifier. In this case if K classes exists, K*(K-1)/2 classifiers are needed. In the latter case, a classifier is trained to separate each class from the rest of the classes; this approach uses K classifiers for a K-class problem. In this paper the first approach is used.

4.2. Goal-related labeling using EHVS

In the second phase of the two-step semantic labeling task, each utterance is labeled according to the detected goal from the previous phase. Suppose that K individual goals exist in a given domain, and then K EHVS parsers should be trained according to the identified topics using utterances related to each goal. In the testing phase, for each test sentence, first the goal is determined using the Goal detector, then the relevant EHVS parser will be chosen to label the sentence wholly.

5. UNIVERSITY INFORMATION KIOSK CORPUS

The University Information Kiosk corpus is a little corpus containing 205 spoken sentences annotated abstractly and hierarchically with semantic annotations. The vocabulary size of the whole corpus is 166. It contains five semantic goals. The following table includes the main goals and frequencies of each goal. It also contains an example for each goal in the third column. For each goal, the first row of the third column contains a Persian sentence, the second row contains its English translation, and the third row is the semantic annotation of the given example.

Table 1.Table of University Information Corpus with goals and examples

Goals	# of sentences	Persian sentence
		English translation
		Semantic annotation
COURSE	37	می‌خواستم بدونم آقای دکتر محمدی این ترم چه درس‌هایی را ارائه داده است.
		I want to know which courses Mr. Mohammadi is presenting this term.
		COURSE(TITLE, NAME)
FIELD	49	تحقیقات آقای محمدی در چه زمینه‌هایی است ؟
		What are the research fields of Mr. Mohammadi?
		FIELD(TITLE, NAME)
OFFICE	38	لطف کنید شماره اتاق آقای محمدی را به من بدهید.
		Please give me the room number of Mr. Mohammadi.
		LAB(TITLE, NAME, NUM)
LAB	32	شما می تونید شماره آزمایشگاه آقای محمدی را به من بدهید ؟
		Could you give me the laboratory number of Mr. Mohammadi?
		LAB(TITLE, NAME, NUM)
TEL	49	شما شماره تلفن آزمایشگاه آقای محمدی را می‌دانید ؟
		Do you know the phone number of Mr. Mohammadi's Laboratory?
		TEL(TITLE, NAME, LAB)

Total number of semantics is 11. This corpus also contains part-of-speech tags for all utterances. Stems of utterances are also included in corpus to take advantage of linguistic knowledge for input parameterization task.

6. EXPERIMENTS AND RESULTS

Experiments are conducted on the University Information Kiosk corpus. In all experiments, the corpus is divided into two parts: 80% is chosen randomly for training and the rest is used for testing.

Two measures can be used to evaluate the performance of the HVS and EHVS parsers as [5] did:

1. Semantic accuracy: is the fraction of guessed semantic which exactly match the reference semantics. It is calculated as:

$$\text{Semantic Accuracy}=\frac{E}{N}*100\% \quad (8)$$

where E is the number of guessed semantics which exactly match the reference semantics and N denotes the number of evaluated semantics.

2. Concept accuracy: this is measured by means of tree edit distance algorithm [11]. It actually calculates the minimum number of substitutions (S), deletions (D), and insertions (I) to convert one semantic tree to another. It is calculated as:

$$\text{Concept Accuracy} = \frac{(N-D-I-S)}{N} * 100\% \quad (9)$$

where N is the number of evaluated semantics.

The second measure seems more reasonable since it does not depend on word alignment and is not tough. So we consider Concept Accuracy measure in our experiments.
The HVS and EHVS parsers are implemented by Graphical Model Toolkit (GMTK) [12].

The input feature vectors to EHVS consist of the observation words themselves, their stems, and their corresponding part-of-speech tags. It is also allowed to have left-right branching in the experiments. Negative examples are not useful in this domain; this feature could probably be better used by expanding the domain. The results of HVS and EHVS are shown in Table 4.

The overall training process of the new method introduced here consists of two parts: Goal detection using SVM, and goal-related labeling by EHVS.
The first step is done using LIBSVM [13]. Each utterance should be converted to a binary feature vector. Three types of feature vectors are used to train the SVM classifier:

a. Each element of the feature vector corresponds to a word in vocabulary of the corpus, it will be one if the given utterance contains the word and zero otherwise. The length of the feature vector is equal to the vocabulary size and is 166.
b. Each element of the feature vector corresponds to a word in the vocabulary of the stems of the corpus, it will be one if the given utterance contains the stem and zero otherwise. The length of the feature vector is 118.
c. In addition to the feature type b, part-of-speech tags of the utterances are used. There are 22 individual pos tags in the corpus. Thus the feature vector will be of length 140.

Since the number of features is large enough taking into consideration the corpus size, the linear kernel is selected for the SVM classifier. Since the dataset is somehow small, the leave-one-out cross-validation (LOOCV) is applied to gain a higher accuracy on SVM. LOOCV involves using a single observation as the validation data, and the remaining observations as the training data. This is repeated until each observation is used once as the validation data. Table 2 shows the SVM accuracy achieved by each type of feature vectors using LOOCV.

Table 2.SVM accuracy by LOOCV for three different feature types

Feature Type	SVM accuracy by LOOCV
a	94.14%
b	97.56%
c	98.04%

Since the classifier accuracy for feature type c is more than the other two types, this classifier is used for the goal detection step. After classifying all of the inputs by SVM, assign the inputs with identical goal to the same group and apply EHVS on each group separately. Again to apply train and test phase on each group, its data is divided randomly into train and test sets by 80% and 20% respectively. The results achieved for each goal by EHVS in the second step is represented in Table 3.

Table 3.EHVS performance for each goal

Goal	Concept Accuracy
COURSE	100%
FIELD	100%
OFFICE	84.38%
LAB	75%
TEL	81.49%

The results of all three experiments are shown in Table 4. Since the EHVS performance on each goal increases greatly, the total performance of the two-step EHVS parser improves effectively.

Table 4.Performance of three parsers

Parser Type	Concept Accuracy
HVS	45.52%
EHVS	58.21%
Two-step EHVS	86.41%

7. CONCLUSIONS AND FUTURE WORK

In this paper, semantic tagging is divided into two subproblems to create a two-step EHVS parser. The first step is organized to determine the main goal of the utterance, and then according to the identified goals, different EHVS parsers will be trained. The new method improves the performance of the EHVS parser mainly because the search space, model complexity, and training time are reduced effectively. Additionally, this will reduce the data sparseness; and hence the need for a large amount of data in statistical approaches reduces.

The future works include applying other types of classifiers for goal detection, and other types of statistical parsers for the second step. Moreover, to reduce the need for labeled data and improve the model, it is worthwhile to apply semi-supervised approaches.

8. REFERENCES

[1] E. Levin, and R. Pieraccini, "CHRONUS, the next generation," *In: Proceedings of the DARPA Speech and Natural, Language Workshop*, Austin, Texas, pp. 269–271, 1995.

[2] S. Miller, R. Schwartz, R. Bobrow, and R. Ingria, "Statistical Language Processing Using Hidden Understanding Models," *In: Proceedings of the workshop on Human Language Technology* , Plainsboro, NJ, pp. 278-282, 1994.

[3] S. Fine, Y. Singer, and N. Tishby, "The hierarchical hidden Markov model: Analysis and applications," *Machine Learning*, Springer, vol. 32, no. 1, pp. 41–62, 1998.

[4] Y. He, and S. Young, "Semantic processing using the hidden vector state model," *Computer Speech and Language*, Elsevier, vol. 19, no. 1, pp. 85–106, 2005.

[5] J. Svec, and F. Jurcicek, "Extended Hidden Vector State Parser", *In: Proceedings of the 12th International Conference on Text, Speech and Dialogue*, Plzen, Czech Republic, pp. 403-410, 2009.

[6] F. Jurcicek, J. Svec, and L. Muller, "Extension of HVS Semantic Parser by Allowing Left-Right Branching," *In: Proceedings of IEEE ICASSP*, Las Vegas, Nevada, USA, pp. 4993-4996, 2008.

[7] J. Svec, F. Jurcicek, and L. Muller, "Parameterization of the input in training the HVS semantic parser," *In: Proceedings of the 10th international conference on Text, speech and dialogue*, Plzen, Czech Republic, pp. 415-422, 2007.

[8] A. Gorin, G. Riccardi, and J. Wright, "How may I help you?," *Speech Communication*, Elsevier, vol. 23, no. 1-2, pp. 113–127, 1997.

[9] C. Wutiwiwatchai, and S. Furui, "Combination of finite state automata and neural network for spoken language understanding," *In: Proceedings of EUROSPEECH*, Geneva, Switzerland, pp. 2761-2764, 2003.

[10] Y. Wang, A. Acero, C. Chelba, B. Frey, and L. Wong, "Combination of statistical and rule-based approaches for spoken language understanding," *In: Proceedings of ICSLP*, Denver, Colorado, pp. 609–612, 2002.

[11] P. Klein, "Computing the edit-distance between unrooted ordered trees," *In: Proceedings of the 6th Annual European Symposium on Algorithms*, London, UK, pp. 91–102, 1998.

[12] J. Bilmes, and G. Zweig, "The graphical models toolkit: An open source software system for speech and time-series processing," *In: Proceedings of IEEE ICASSP*, pp. IV-3916-IV-3919, 2002.

[13] CC. Chang, and CJ. Lin. LIBSVM: a library for support vector machines. http ://www.csie.ntu.edu.tw/~cjlin/libsvm, 2001.