Trajectory Based Word Sense Disambiguation

Xiaojie Wang [†‡] **Yuji Matsumoto** [†]
[†] Graduate School of Information Science, Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara, 630-0192, Japan
[‡] School of Information Engineering, Beijing University of Posts and Technology
Beijing, 100876, China
{xiaoji-w, matsu}@is.naist.jp

Abstract

Classifier combination is a promising way to improve performance of word sense disambiguation. We propose a new combinational method in this paper. We first construct a series of Naïve Bayesian classifiers along a sequence of orderly varying sized windows of context, and perform sense selection for both training samples and test samples using these classifiers. We thus get a sense selection trajectory along the sequence of context windows for each sample. Then we make use of these trajectories to make final k-nearest-neighbors-based sense selection for test samples. This method aims to lower the uncertainty brought by classifiers using different context windows and make more robust utilization of context while perform well. Experiments show that our approach outperforms some other algorithms on both robustness and performance.

1 Introduction

Word sense disambiguation (WSD) has long been a central issue in Natural Language Processing (NLP). In many NLP tasks, such as Machine Translation, Information Retrieval etc., WSD plays a very important role in improving the quality of systems. Many different algorithms have been used for this task, including some machine learning (ML) algorithms, such as Naïve Bayesian model, decision trees, and example based learners. Since different algorithms have different strengths and perform well on different feature space, classifier combination is a reasonable candidate to achieve better performance by taking advantages of different approaches. In the field of ML, ensembles of classifiers have been shown to be successful in last decade (Dietterich 1997). For the specific task of WSD, classifier combination has been received more and more attention in recent years.

Kilgarriff and Rosenzweig (2000) presented the first empirical study. They combined the output of the participating SENSEVAL1 systems via simple voting. Pedersen (2000) built an ensemble of Naïve Bayesian classifiers, each of which is based on lexical features that represent co-occurring words in varying sized windows of context. The sense that receives majority of the votes was assigned as the final selection. Stevenson and Wilks (2001) presented a classifier combination framework where three different disambiguation modules were combined using a memory-based approach. Hoste et al. (2002) used word experts consisted of four memory-based learners trained on different context. Output of the word experts is based on majority voting or weighted voting. Florian et al.(2002) and Florian and Yarowsky (2002) used six different classifiers as components of their combination. They compared several different strategies of combination, which include combining the posterior distribution, combination based on order statistics and several different voting. Klein et al. (2002) combined a number of different first-order classifiers using majority voting, weighted voting and maximum entropy. In Park (2003), a committee of classifiers was used to learn from the unlabeled examples. The label of an unlabeled example is predicted by weighted majority voting. Frank at al. (2003) presented a locally weighted Naïve Bayesian model. For a given test instance, they first chose k-nearest-neighbors from training samples for it, then constructed a Naïve Bayesian classifier by using these k-nearest-neighbors in stead of all training samples.

This paper presents a new combinational approach. We firstly construct a series of Naïve Bayesian classifiers along a sequence of orderly varying sized windows of context, and make sense selection for both training samples and test samples using these classifiers. We thus get a trajectory of sense selection for each sample, and then use the sense trajectory based k-nearest-neighbors to make final decision for test samples.

This method is motivated by an observation that there is an unavoidable uncertainty when a classifier is used to make sense selection. Our approach aims to alleviate this uncertainty and thus make more robust utilization of context while perform well. Experiments show our approach outperform some other algorithms on both robustness and performance.

The remainder of this paper is organized as follows: Section 2 gives the motivation of our approach, describes the uncertainty in sense selection brought by classifiers themselves. In section 3, we present the decision trajectory based approach. We then implement some experiments in section 4, and give some evaluations and discussions in section 5. Finally, we draw some conclusions.

2 The Trajectory of Sense Selection

Our method is originally motivated by an observation on relation between sense selection by a classifier and the context it uses to make this selection.

As well known, context is the only means to identify the sense of a polysemous word. Ide (1998) identified three types of context: micro-context, topical context and domain. In practice, a context window (l , r), which includes l words to the left and r words to the right of the target word, is predetermined by human or chosen automatically by a performance criterion. Only information in the context window is then used for classifiers and disambiguating. What is the best window size for WSD has been long for a problem. Weaver (1955) hoped we could find a minimum value of the window size which can lead to the correct choice of sense for the target ambiguous word. Yarowsky (1994) argued the optimal value is sensitive to the type of ambiguity. Semantic or topic-based ambiguities warrant a larger window (from 20 to 50), while more local syntactic ambiguities warrant a smaller window (3 or 4). Leacock at el (1998) showed the local context is superior to topical context as an indicator of word sense. Yarowsky (2002) suggested that different algorithms prefer different window sizes.

Followed by these works, it is clear that different window sizes might cause different sense selection for an occurrence of the target word even when a same algorithm is used. Yarowsky (2002) gave a investigation on how the performance changes with different window sizes for several different algorithms and several different types of word. In fact, even for human, different window sizes might cause different sense selections for a same occurrence of an ambiguous word. For example, considering word "看"(It has two different senses: "read" and "think") in senesce S1.

S1: 我/	看/	这/本/	书/	值得/	一/	读/.
(I)	(think)	(this)	(book)	(worthy)	(a)	(read)

When we use a context window (1,1), it is not clear which sense should be more possible in this sentence. When we use (3,3), because the collocation with 书 give a very strong indication for 看's sense, it is natural that we select the sense of "read" for 看. When we use window (6,6), we select the sense of "think" for it.

Here, the occurrence of the ambiguous word is the same; it is the difference of context windows that make the sense selection different. Since the context window is a built-in parameter of a classifier, as long as we use a classifier to distinguish an ambiguous word, we had to choose a window size. Supposing a classifier is an observer, choosing a window size is necessary for the observer to implement an observation. Different choices of the window size might cause different observational results for the same occurrence. That means there is an uncertainty brought by observer itself. It reminds us that the relation between the window size and the sense selection is to some extent similar with the relation between a particle's position and its momentum in Heisenberg Uncertainty Principle.

By the Uncertainty Principle, when we measure the position and the momentum of a particle, we cannot measure them with zero-variance simultaneously. In Quantum Theory, the wave-function is used to describe the state of a particle. The method to deal with this problem in Quantum Theory suggests us an idea to deal with the similar problem in WSD.

Firstly, since the existence of the uncertainty of sense selection at different window sizes, sense selection for the target word at only one context window cannot give a complete description of its sense. To grasp a complete description of its sense, it is necessary to get sense selections along a series of observation, i.e. using a sequence of context window to get a trajectory of sense selection.

Secondly, unlike that in Uncertainty Principle, the intuition is that, in most of time, when we have enough observations, we can be doubtlessly sure the sense of the target word. So, we make final unique sense selection based on the trajectory of sense selection. Since the final selection is based on a sense trajectory along different window sizes, we thus think it may helpful to alleviate the uncertainty brought by difference of context windows.

In this way, our approach aims to improve robustness of WSD. Here the robustness means that sense selection is not sensitive to the window size. This kind of robustness is especially important to WSD system in noise or oral corpus, where there are many occasional inserted words near the target word. Besides robustness, to achieve better performance is also necessary, if robustness is at a low level of performance, it is useless.

3 Decision Trajectory Based WSD

In our approach, we firstly use Naïve Bayesian (NB) algorithm to construct a sense selection trajectory along a sequence of orderly varying sized windows of context for each sample, including both training samples and test samples. Then we use k-nearest-neighbors(KNN) algorithm to make final decision for each test sample based on these trajectories.

Let w be an ambiguous word, it has n different senses, $s_1 \ldots s_i \ldots s_n$. Supposing we have q training samples $S_1 \ldots S_j \ldots S_q$, where q_i samples are tagged with sense s_i, $\sum q_i = q$. We present our approach in two stages: training stage and test stage. Figure 1 gives a skeleton of the algorithm.

In the training stage, we first choose a sequence of context windows.

$$T_m : (p_1, \ldots p_k \ldots p_m)$$

Where $p_k = (l_k, r_k)$ is a context window which includes l_k words to the left of word w and r_k words to the right. We call T_m a trajectory of context windows, p_k is a window point in this trajectory. For example, a trajectory ((1,1), (3,3), (5,5), (7,7), (9,9)) includes 5 points.

For each window point p_k in T_m, we construct a classier by using NB algorithm based on context word in p_k. Let $C(p_k)$ denote the classifier, it can be thought as an operator that make sense selection upon samples. With the change of the window point, we can get a operate vector:

$$C = (C(p_1), \ldots, C(p_k), \ldots, C(p_m))$$

```
Training stage:
1. To construct a operator vector $C$ along a
   sequence of context windows $T_m$:
      $C = (C(p_1),...,C(p_k),...,C(p_m))$.
   $C(p_k)$ is a NB classifier learned by all the
   tagged data using $p_k$ as the context
   window.
2. For each training sample $S_j$, to operate $C$
   upon it to construct a sense trajectory, $\omega_j$
   ($j = 1,...,q$).

Test stage:
1. For a new sample $S$, to construct its
   decision trajectory $\omega$ by operating $C$
   upon it.
2. For $j = 1,...,q$, to calculate $d(\omega, \omega_j)$
3. to make KNN-based sense choice for $S$.
```

Figure 1. The algorithm of trajectory-based WSD

For a sample S_j for sense s_i, we use $C(p_k)(S_j)$ to denote using $C(p_k)$ to classify S_j, we can get a sense selection denoted by $\omega_j(p_k)$, i.e., $C(p_k)(S_j) = \omega_j(p_k)$. We call $\omega_j(p_k)$ a point decision. If $\omega_j(p_k) = s_i$, we borrow a term to call S_j an eigen-sample of the operator $C(p_k)$, s_i is its eigenvaluve.

With the change of the window point, we get a sequence of point decisions for sample S_j along the window trajectory T_m, we denoted it by

$$\omega_j = (\omega_j(p_1), \ldots, \omega_j(p_m))$$

We call it decision trajectory of sample S_j along the context windows trajectory T_m. If all elements of ω_j is s_i, i.e. $\omega_j = (s_i, \ldots, s_i)$, we call S_j an eigen-sample of operator C, ω_j is a eigen-trajectory of C.

In this way, we transfer training samples into training decision trajectories, which will be used as instance for final KNN-based sense selection. An eigen-trajectory is a good indication for a sample, but when all the training samples are eigen-samples, it is not a good thing for disambiguating new samples. We will discuss this case in section 5.2.

After finishing training stage of our approach, we have a context windows trajectory T_m, a sequence of classifiers $C(p_k)$ along T_m, and a decision trajectory for each training sample. All these compose of our classifier for a new sample in test. When a new sample is given, we first calculate a decision trajectory ω for it by using C operating upon it. Let

$$\omega = (\omega(p_1), \ldots, \omega(p_m))$$

We then calculate the similarity between ω and ω_j, $j = 1,2,...k$ by using (3.1).

$$Sim(\omega, \omega_j) = \frac{\sum_{i=1}^{m} \delta(\omega(p_i), \omega_j(p_i))}{m} \qquad (3.1)$$

Where $\delta(x, y) = 1$ at $x = y$, and $\delta(x, y) = 0$ at $x \neq y$. We then choose h training decision trajectory samples as ω's h nearest neighbors, supposing that h_i samples are tagged with sense s_i among these nearest neighbors, $\sum h_i = h$, $h_i \leq q_i$, then by solving (3.2), We choose s_{i^*} as the final sense selection for the new sample.

$$i^* = \arg\max_{1 \leq i \leq n} \sum_{1 \leq j \leq h_i} Sim(\omega, \omega_j) \qquad (3.2)$$

If all training samples are eign-samples, the similarity between ω and ω_j for the same sense are the same, (3.2) is changed to:

$$i^* = \arg\max_{1 \leq i \leq n} \{ | \{ \omega(p_k) = i \} |, k = 1,...,m \} \qquad (3.3)$$

Then the final KNN-based decision is simplified to majority voting along the decision trajectory of the new sample.

4 Experiments

4.1 Experimental Data

All experimental data were extracted from Chinese People Daily from January 1995 to December 1996. Eight Chinese ambiguous words were used in the experiments as listed in the first column of Table1. In the Second column, we give some information about samples used in experiments. The number before each bracket is the number of senses. Numbers in each bracket are amounts of samples used for each sense. They were annotated by a Chinese native speaker, and checked by another native speaker. Some samples without inter-agreement between two native speakers had been excluded.

Only word co-occurrences in given windows are used as features through all experiments in this paper.

4.2 Experimental Method

In order to do a comparative study, we have implemented not only our algorithm, but also four other related algorithms in our experiments. They fall into two classes. NB (Manning and Schutze 1999) and KNN (Ng and Lee 1996) are two components of our approach. Locally weighted NB(LWNB, Frank et al. 2003) and Ensemble NB(ENB Pedersen 2000) are two combinational approaches. Since our aim is to compare not only the performance but also the robustness of these algorithms, we implemented each algorithm in following way.

We note our approach TB_KNN when (3.2) is used for final decision, and TB_VOTE when (3.3) is used for final decision.

We firstly constructed a sequence of context windows $p_k = (l_k, r_k)$ $k = 1,...,40$ in following way:

1. Initiate: $l_1 = 0, r_1 = 1$
2. Generate next window:

$$\begin{cases} l_{k+1} = l_k, r_{k+1} = r_k + 1 & if \quad l_k = r_k \\ l_{k+1} = l_k + 1, r_{k+1} = r_k & if \quad l_k < r_k \end{cases}$$
$$k = 1,...,39$$

We then constructed a sequence of window trajectories.

$$T_i = (p_1,..., p_i) \qquad i = 1,...,40$$

We implemented TB_KNN and TB_VOTE on each trajectory from T_1 to T_{40}.

Obviously, our T_i -based sense selection and p_i -based selection in fact make use of same context surrounding the target word. p_i is the biggest window along T_i. We implemented NB classifiers (noted by P) from p_1 to p_{40}.

KNN was implemented along the same sequence of context window, from p_1 to p_{40}.

For the implementation of algorithm LWNB, we used the measure in (Ng and Lee 1996) to find k nearest neighbors for each sample, and then constructed a NB classifier according to (Frank2003). This algorithm was also implemented for each context window along the sequence from p_1 to p_{40}.

ENB was implemented according to (Petersen 2000). Different left and right window sizes we used is (1,2,3,4,5,6,10,15,20). Since one implementation of this algorithm make use of all these different window sizes. It cannot be implemented along above windows sequence, so there is only one implementation for this algorithm.

For each ambiguous word, we implemented above experiments respectively, each experiment was a 10-fold cross-validation, at each time 90% of the data were used as training data, 5% were used as development data, and other 5% were used as test data.

4.3 Experimental Results

We give the results curves for word "告" in Figure 2 and for word "想" in Figure 3. In both figures, x-axis is the context window, from (0,1) to (20,20), y-axis is F-measure, and different marker style is for different algorithms. Results curves for other six target words have similar shapes.

We list a summary of results for all 8 words in Table 1. In TB_KNN column, there are three values: mean, maximum and standard variance of F-measure of 40 different trajectories from T_1 to T_{40}. Results are summarized in the same way in column TB_VOTE. For column P, KNN and LWNB, three values are mean, maximum and standard variance of F-measure of 40 different points from p_1 to p_{40}. In column ENB, there is only one F-measure.

5 Evaluation

5.1 Comparison with other algorithms

As we have mentioned, we compare results of each algorithm on both performance and robustness. Performance can be compared directly from F-measure point-wise along a sequence of context windows(or trajectories). We also use mean and maximum (max) along the sequence to give an overall comparison. Robustness of an algorithm means that sense decision varies gracefully with the change of context windows (or trajectories) it uses. Intuitionally, it can be reflected by a context-performance curve, a flat curve is more robust than a sharp one. We also use standard variance (S.V.) along a sequence of sense selection to give an overall comparison. A sequence with small standard variance is more robust than that with a big one.

From Figures 2 and 3, we can get an intuitional impression that TB_KNN not only achieve the best performance at most of points, but also has the flattest curve shape. This means TB_KNN outperforms other algorithm on both performance and robustness. This can be detailed in Table 1 by comparing mean/max/S.V. of TB_KNN with their correspondences in other algorithms.

Comparing values in the TB_KNN column with their correspondences in column P, we can find all values of TB_KNN are consistently better than those in P. For "mean" and "max", a bigger one is better, while for S.V., a little one is better. Comparing values in the TB_KNN column with their correspondences in column KNN, we can find nearly all values of TB_KNN are better than those in KNN. (Except that KNN's max and S.V. for word "穿" are better then those in TB_KNN). All differences are significant. This means our decision trajectory based classifier is better than a NB classifier or a KNN classifier. The combination takes advantages of both NB and KNN methods. It seems that KNN directly based on word co-occurrence features suffers deeply from data sparseness. While KNN based on decision trajectory can alleviate the influence of data sparseness. In our final KNN decision, sense selection is also not sensitive to the number of nearest neighbors.

Comparing values in TB_KNN column with their correspondences in column LWNB, we can find most of values in TB_KNN are better than their correspondences in LWNB. But the differences are not so bigger than those described

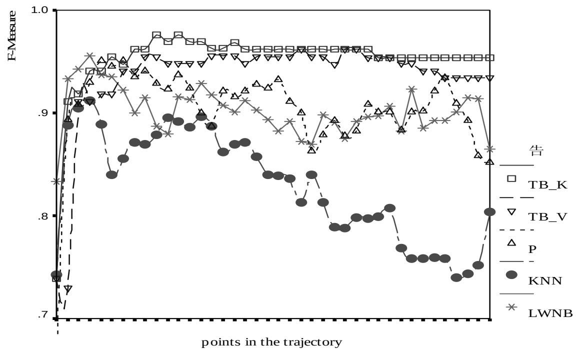

Figure 2. context-performance curves for "告"

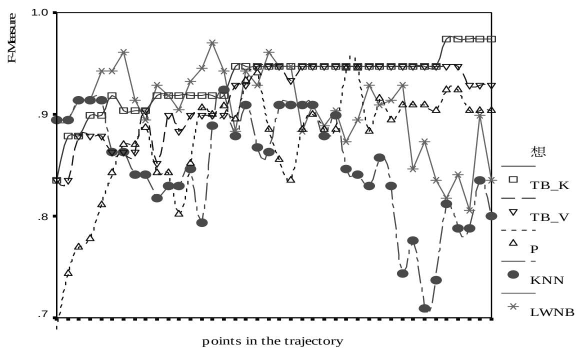

Figure 3. context-performance curves for "想"

Word	Num-Sen	TB_KNN	TB_VOTE	P	KNN	LWNB	ENB
待	3(68,73,35)	**0.88**/0.91/**0.03**	0.86/0.89/0.03	0.83/0.88/0.06	0.86/0.91/0.04	0.87/**0.92**/0.04	0.89
告	3(31,62,64)	**0.95**/**0.98**/**0.04**	0.94/0.96/0.05	0.90/0.95/0.05	0.83/0.91/0.05	0.90/0.96/**0.04**	0.96
看	3(25,28,18)	**0.89**/**0.94**/**0.03**	0.88/**0.94**/0.04	0.80/0.90/0.10	0.69/0.82/0.07	0.76/0.87/0.07	0.82
存	4(42,36,31,28)	**0.80**/**0.84**/**0.04**	0.79/0.83/0.04	0.74/0.83/0.06	0.75/0.81/0.05	0.74/0.80/ 0.04	0.79
想	2(24,33)	**0.93**/ **0.97**/**0.03**	0.92/**0.97**/0.04	0.88/0.95/0.05	0.85/0.92/0.05	0.91/ **0.97**/0.04	0.89
穿	2(40,36)	**0.91**/0.96/**0.06**	0.89/0.94/0.07	0.85/0.96/0.18	0.89/**0.97**/0.06	0.87/**0.97**/0.06	0.84
换	2(43,52)	**0.86**/**0.89**/**0.03**	0.84/0.87/0.04	0.83/0.88/0.04	0.73/0.80/0.03	0.82/0.88/0.04	0.89
爱	2(15,15)	**0.83**/ **0.92**/**0.05**	0.82/ 0.89/0.05	0.77/0.87/0.07	0.49/0.82/0.13	0.79/0.89/0.06	0.77

Table 1 result summary

in above paragraph, especially when the number of training samples is relatively big. In Frank et al.(2003), the number of training samples is large.(Most of them are more than several hundreds.) They used 50 local training samples to construct a NB classifier. It is always impossible in our experiments and in most WSD tasks.

Although not all of the values of mean in TB_KNN column are bigger than their correspondences in ENB, all maximums are bigger (or equal) than those in ENB. Comparing with ENB, We think the trajectory based approach may make use of NB decisions in a more systematical way than selecting some classifiers for voting in ENB, and also, our approach receives benefits from the final KNN decision, which can make some exceptions under consideration.

Let us give a discussion on how our trajectory-based approach makes use of information in context.

Firstly, although each NB classifier use bag-of-words as its features, because window size for NB classifiers is extended sequentially, the decision trajectory thus reflects influences brought by context words in different positions. That is to say, changing the position of a co-occurrence word in a sentence might cause different final decision in trajectory-based approach. While in point-based approach, as long as the co-occurrence word is in the context window, a classifier based on bag-of-words features always makes the same selection no matter how to change the position of that word. From this view, the trajectory-based approach in fact makes use of position information of words in context.

Secondly, because of its implicit utilization of position information of context words, it may make use of information from some decisions locally correct but globally wrong. For example, we consider sentence S1 in section 2 again.

S1:我/ 看/ 这/本/ 书/ 值得/ 一/ 读/.
(I) (think) (this) (book) (worthy) (a) (read)

On the one hand, as we have said, when we use context window (3,3), we select the sense of "read" for 看. Although it is a wrong sense selection for this word in this sentence (when context window is (6,6)), it is a correct selection for the local collocation (when 看 collocates with 书, its sense is "read"). By saving this information, we cannot only make use of information of sense selection for the sentence, but also information for this collocation. In other words, the sentence S1 gives us two samples for different senses of the target word.

On the other hand, that a polysemous word changes their probability for different sense with the change of context window is one type of pattern for sense ambiguity, the trajectory based approach seems an efficient way to grasp this pattern of ambiguity.

5.2 Trajectory

In TB_KNN, we need to calculate a sense decision trajectory for each training sample, not all of these trajectories are eigen-trajecories. In TB_VOTE, we don't calculate sense decision trajectories for training samples, all training decision trajectories are regarded as eigen-trajectory, final decision for a new sample reduces to majority voting along the trajectory. Comparing TB_KNN and TB_VOTE, we can find that both performance and robustness of TB_VOTE fall. This means existence of non-eigen-trajectory is in fact helpful, which can make some exceptions under consideration by using KNN.

In above experiments, we generated a trajectory by adding one context word each time. We further explored if a looser trajectory can get the same performance. We first excluded even points in original trajectories in above experiments to get some new trajectories. For example, by excluding even points of the trajectory $T_{40} = \{p_1,..., p_{40}\}$, we got:

$$T'_{20} = \{p_1,..., p_{2k-1},.., p_{39}\}, k = 1,...,20$$

Note this T'_{20} is different from T_{20} in above experiments, where T_{20} is:

$$T_{20} = \{p_1,..., p_k,.., p_{20}\}, k = 1,...,20$$

In this way, we got 20 different trajectories TG2: $T'_1,...,T'_{20}$, T'_j includes half number of points comparing with its correspondence T_{2j} in above experiments. The longest trajectory includes 20 points. We repeated above TB_KNN experiment along these new trajectories. Results are listed in column TB_KNN TG2 in Table2. We excluded even points to generate TG3 and TG4 which include at most 10 and 5 points respectively in their trajectories. We also repeated same TB_KNN experiment on TG3 and TG4.

	TB_KNN TG2	TB_KNN TG3	TB_KNN TG4
待	0.87/0.90/0.03	0.87/0.91/0.03	0.86/0.89/0.03
告	0.95/0.97/0.01	0.95/0.96/0.01	0.94/0.95/0.02
看	0.90/0.93/0.02	0.90/0.91/0.02	0.90/0.93/0.03
存	0.80/0.84/0.02	0.78/0.82/0.03	0.77/0.81/0.02
想	0.93/0.97/0.03	0.93/0.95/0.02	0.92/0.96/0.03
穿	0.91/0.94/0.06	0.91/0.93/0.06	0.90/0.94/0.09
换	0.86/0.90/0.03	0.86/0.90/0.04	0.82/0.85/0.03
爱	0.83/0.92/0.05	0.85/0.92/0.05	0.82/0.92/0.07

Table 2: shorter length in the trajectory

From Table 2, we can find that performance of classifiers using trajectories with small number of points do not decrease significantly. That is to say, a shorter trajectory can also achieve good performance.

6 conclusions

This paper presents a new type of classifier combination method. We firstly construct a sequence of NB classifiers along orderly varying sized windows of context, and get a trajectory of sense selection for each sample, then use the sense trajectory based KNN to make final decision for test samples. Experiments show that our approach outperforms some other algorithms on both robustness and performance.

We will do further investigations on the trajectory to see if there exists some skeletal points like quantum numbers in the wavefunction in Quantum Theory.

References

Thomas G. Dietterich. 1997. Machine Learning Research: Four Current Directions. *AI Magazine*. Vol. 18, No. 4 pp.97-136.

Radu Florian, Silviu Cucerzan, C Schafer and D. Yarowsky. 2002. Combining Classifiers for Word Sense Disambiguation. *Journal of Natural Language Engineering*. Vol. 8 No.4.

Radu Florian and D. Yarowsky. 2002. Modeling Consensus: Classifier Combination for Word Sense Disambiguation. In *Proceedings of EMNLP'02*, pp25-32.

Eibe Frank, M. Hall and Bernhard Pfahringer. 2003. Locally Weighted Naïve Bayes. *Proceedings of the Conference on Uncertainty in Artificial Intelligence*.

Véronique Hoste, I. Hendrickx, W. Daelemans, and A. van den Bosch.2002. Parameter optimization for machine-learning of word sense disambiguation. *Natural Language Engineering*,8(3).

Nancy Ide, J Veronis.1998. Introduction to the Special Issue on Word Sense Disambiguation: The State of the Art. *Computational Linguistics*, 24(1):1-40.

Dan Klein, K. Toutanova, H. Tolga Ilhan, S. D. Kamvar, and C. D. Manning. 2002. Combining Heterogeneous Classifiers for Word-Sense Disambiguation. In *Workshop on Word Sense Disambiguation at ACL 40*, pages 74-80.

Adam Kilgarriff and J. Rosenzweig (2000). Framework and results for English Senseval. Computers and the Humanities. 34(1):15-48.

Chris D. Manning and H. Schutze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press.

Rada Mihalcea. 2002. Word Sense Disambiguation Using Pattern Learning and Automatic Feature Selection, *Journal of Natural Language and Engineering*, 8(4):343-358.

Hwee Tou Ng, Hian Beng Lee. 1996. Integrating Multiple Knowledge Sources to Disambiguate Word Sense: An Exemplar-Based Approach. In *Proceedings of the Thirty-Fourth ACL*.

Ted Pedersen 2000. A Simple Approach to Building Ensembles of Naive Bayesian Classifiers for Word Sense Disambiguation. In the *Proceedings of the NAACL-00*.

David Yarowsky 1994. Decision Lists for Lexical Ambiguity Resolution: Application to Accent Restoration in Spanish and French." In *Proceedings of the 32nd ACL*. pp. 88-95.

David Yarowsky and R. Florian.2002. Evaluating Sense Disambiguation Performance Across Diverse Parameter Spaces. *Journal of Natural Language Engineering*, Vol.8, No 4.