Semi-Supervised Training of a Kernel PCA-Based Model for Word Sense Disambiguation

Weifeng SU weifeng@cs.ust.hk
Marine CARPUAT marine@cs.ust.hk
Dekai WU[1] dekai@cs.ust.hk

Human Language Technology Center
HKUST
Department of Computer Science
University of Science and Technology, Clear Water Bay, Hong Kong

Abstract

In this paper, we introduce a new semi-supervised learning model for word sense disambiguation based on *Kernel Principal Component Analysis* (*KPCA*), with experiments showing that it can further improve accuracy over supervised KPCA models that have achieved WSD accuracy superior to the best published individual models. Although empirical results with supervised KPCA models demonstrate significantly better accuracy compared to the state-of-the-art achieved by either naïve Bayes or maximum entropy models on Senseval-2 data, we identify specific sparse data conditions under which supervised KPCA models deteriorate to essentially a most-frequent-sense predictor. We discuss the potential of KPCA for leveraging unannotated data for partially-unsupervised training to address these issues, leading to a composite model that combines both the supervised and semi-supervised models.

1 Introduction

Wu *et al.* (2004) propose an efficient and accurate new supervised learning model for word sense disambiguation (WSD), that exploits a nonlinear Kernel Principal Component Analysis (KPCA) technique to make predictions implicitly based on generalizations over feature combinations. Experiments performed on the Senseval-2 English lexical sample data show that KPCA-based word sense disambiguation method is capable of outperforming other widely used WSD models including naïve Bayes, maximum entropy, and SVM models.

Despite the excellent performance of the supervised KPCA-based WSD model on average, though, our further error analysis investigations have suggested certain limitations. In particular, the supervised KPCA-based model often appears to perform poorly when it encounters target words whose contexts are highly dissimilar to those of any previously seen instances in the training set. Empirically, the supervised KPCA-based model nearly always disambiguates target words of this kind to the most frequent sense. As a result, for this particular subset of test instances, the precision achieved by the KPCA-based model is essentially no higher than the precision achieved by the most-frequent-sense baseline model (which simply always selects the most frequent sense for the target word). The work reported in this paper stems from a hypothesis that the most-frequent-sense strategy can be bettered for this category of errors.

This is a case of data sparseness, so the observation should not be very surprising. Such behavior is to be expected from classifiers in general, and not just the KPCA-based model. Put another way, even though KPCA is able to generalize over combinations of dependent features, there must be a sufficient number of training instances from which to generalize.

The nature of KPCA, however, suggests a strategy that is not applicable to many of the other conventional WSD models. We propose a model in this paper that takes advantage of unsupervised training using large quantities of unannotated corpora, to help compensate for sparse data.

Note that although we are using the WSD task to explain the model, in fact the proposed model is not limited to WSD applications. We have hypothesized that the KPCA-based method is likely to be widely applicable to other NLP tasks; since data sparseness is a common problem in many NLP tasks, a weakly-supervised approach allowing the KPCA-based method to compensate for data sparseness is highly desirable. The general technique we describe here is applicable to any similar classification task where insufficient labeled training data is available.

The paper is organized as follows. After a brief look at related work, we review the baseline supervised WSD model, which is based on Kernel PCA. We then discuss how data sparseness affects the model, and propose a new semi-supervised model that takes advantage of unlabeled data, along with a composite model that combines both the supervised and semi-supervised models. Finally, details of the experimental setup and comparative results are given.

2 Related work

The long history of WSD research includes numerous statistically trained methods; space only permits us to summarize a few key points here. Naïve Bayes models (e.g., Mooney (1996), Chodorow *et al.* (1999), Pedersen (2001), Yarowsky and Florian (2002)) as well as maximum entropy models (e.g., Dang and Palmer (2002), Klein and Manning (2002)) in particular have shown a large degree of success for WSD, and have established challenging state-of-the-art benchmarks. The Senseval series of evaluations facilitates comparing the strengths and weaknesses of various WSD models on common data sets, with Senseval-1 (Kilgarriff and Rosenzweig,

[1]The author would like to thank the Hong Kong Research Grants Council (RGC) for supporting this research in part through grants RGC6083/99E, RGC6256/00E, and DAG03/04.EG09.

1999), Senseval-2 (Kilgarriff, 2001), and Senseval-3 held in 1998, 2001, and 2004 respectively.

3 Supervised KPCA baseline model

Our baseline WSD model is a supervised learning model that also makes use of *Kernel Principal Component Analysis* (KPCA), proposed by (Schölkopf *et al.*, 1998) as a generalization of PCA. KPCA has been successfully applied in many areas such as de-noising of images of hand-written digits (Mika *et al.*, 1999) and modeling the distribution of non-linear data sets in the context of shape modelling for real objects (Active Shape Models) (Twining and Taylor, 2001). In this section, we first review the theory of KPCA and explanation of why it is suited for WSD applications.

3.1 Kernel Principal Component Analysis

The *Kernel Principal Component Analysis* technique, or *KPCA*, is a method of nonlinear principal component extraction. A nonlinear function maps the n-dimensional input vectors from their original space R^n to a high-dimensional feature space F where linear PCA is performed. In real applications, the nonlinear function is usually not explicitly provided. Instead we use a kernel function to implicitly define the nonlinear mapping; in this respect KPCA is similar to Support Vector Machines (Schölkopf *et al.*, 1998).

Compared with other common analysis techniques, KPCA has several advantages:

- As with other kernel methods it inherently takes *combinations* of predictive features into account when optimizing dimensionality reduction. For natural language problems in general, of course, it is widely recognized that significant accuracy gains can often be achieved by generalizing over relevant feature combinations (e.g., Kudo and Matsumoto (2003)).
- We can select suitable kernel function according to the task we are dealing with and the knowledge we have about the task.
- Another advantage of KPCA is that it is good at dealing with input data with very high dimensionality, a condition where kernel methods excel.

Nonlinear principal components (Diamantaras and Kung, 1996) may be defined as follows. Suppose we are given a training set of M pairs (x_t, c_t) where the observed vectors $x_t \in R^n$ in an n-dimensional input space X represent the context of the target word being disambiguated, and the correct class c_t represents the sense of the word, for $t = 1, .., M$. Suppose Φ is a nonlinear mapping from the input space R^n to the feature space F. Without loss of generality we assume the M vectors are centered vectors in the feature space, i.e., $\sum_{t=1}^{M} \Phi(x_t) = 0$; uncentered vectors can easily be converted to centered vectors (Schölkopf *et al.*, 1998). We wish to diagonalize the covariance matrix in F:

$$C = \frac{1}{M} \sum_{j=1}^{M} \Phi(x_j) \Phi^T(x_j) \quad (1)$$

To do this requires solving the equation $\lambda v = Cv$ for eigenvalues $\lambda \geq 0$ and eigenvectors $v \in F$. Because

$$Cv = \frac{1}{M} \sum_{j=1}^{M} (\Phi(x_j) \cdot v) \Phi(x_j) \quad (2)$$

we can derive the following two useful results. First,

$$\lambda (\Phi(x_t) \cdot v) = \Phi(x_t) \cdot Cv \quad (3)$$

for $t = 1, .., M$. Second, there exist α_i for $i = 1, ..., M$ such that

$$v = \sum_{i=1}^{M} \alpha_i \Phi(x_i) \quad (4)$$

Combining (1), (3), and (4), we obtain

$$M\lambda \sum_{i=1}^{M} \alpha_i (\Phi(x_t) \cdot \Phi(x_i)) = \sum_{i=1}^{M} \alpha_i (\Phi(x_t) \cdot \sum_{j=1}^{M} \Phi(x_j)) (\Phi(x_j) \cdot \Phi(x_i))$$

for $t = 1, .., M$. Let $\hat{K}$ be the $M \times M$ matrix such that

$$\hat{K}_{ij} = \Phi(x_i) \cdot \Phi(x_j) \quad (5)$$

and let $\hat{\lambda}_1 \geq \hat{\lambda}_2 \geq \ldots \geq \hat{\lambda}_M$ denote the eigenvalues of $\hat{K}$ and $\hat{\alpha}^1$,..., $\hat{\alpha}^M$ denote the corresponding complete set of normalized eigenvectors, such that $\hat{\lambda}_t(\hat{\alpha}^t \cdot \hat{\alpha}^t) = 1$ when $\hat{\lambda}_t > 0$. Then the lth nonlinear principal component of any test vector x_t is defined as

$$y_t^l = \sum_{i=1}^{M} \hat{\alpha}_i^l (\Phi(x_i) \cdot \Phi(x_t)) \quad (6)$$

where $\hat{\alpha}_i^l$ is the lth element of $\hat{\alpha}^l$.

3.2 Why is KPCA suited to WSD?

The potential of nonlinear principal components for WSD can be illustrated by a simplified disambiguation example for the ambiguous target word "art", with the two senses shown in Table 1. Assume a training corpus of the eight sentences as shown in Table 2, adapted from Senseval-2 English lexical sample corpus. For each sentence, we show the feature set associated with that occurrence of "art" and the correct sense class. These eight occurrences of "art" can be transformed to a binary vector representation containing one dimension for each feature, as shown in Table 3.

Extracting nonlinear principal components for the vectors in this simple corpus results in nonlinear generalization, reflecting an implicit consideration of combinations of features. Table 2 shows the first three dimensions of the principal component vectors obtained by transforming each of the eight training vectors x_t into (a) principal component vectors z_t using the linear transform obtained via PCA, and (b) nonlinear principal component vectors y_t using the nonlinear transform obtained via KPCA as described below.

Table 1: A tiny corpus for the target word "art", adapted from the Senseval-2 English lexical sample corpus (Kilgarriff 2001), together with a tiny example set of features. The training and testing examples can be represented as a set of binary vectors: each row shows the correct class c for an observed vector x of five dimensions.

	TRAINING	design/N	media/N	the/DT	entertainment/N	world/N	**Class**
x_1	He studies **art** in London.						1
x_2	Punch's weekly guide to the world of the **arts**, entertainment, media and more.		1	1	1	1	
x_3	All such studies have influenced every form of **art**, design, and entertainment in some way.	1			1		1
x_4	Among the technical **arts** cultivated in some continental schools that began to affect England soon after the Norman Conquest were those of measurement and calculation.			1			2
x_5	The **Art** of Love.			1			2
x_6	Indeed, the **art** of doctoring does contribute to better health results and discourages unwarranted malpractice litigation.			1			2
x_7	Countless books and classes teach the **art** of asserting oneself.			1			2
x_8	Pop **art** is an example.						1
	TESTING						
x_9	In the world of design **arts** particularly, this led to appointments made for political rather than academic reasons.	1		1		1	1

Table 2: The original observed training vectors (showing only the first three dimensions) and their first three principal components as transformed via PCA and KPCA.

	Observed vectors	**PCA-transformed vectors**	**KPCA-transformed vectors**	**Class**
t	(x_t^1, x_t^2, x_t^3)	(z_t^1, z_t^2, z_t^3)	(y_t^1, y_t^2, y_t^3)	c_t
1	(0, 0, 0)	(-1.961, 0.2829, 0.2014)	(0.2801, -1.005, -0.06861)	1
2	(0, 1, 1)	(1.675, -1.132, 0.1049)	(1.149, 0.02934, 0.322)	1
3	(1, 0, 0)	(-0.367, 1.697, -0.2391)	(0.8209, 0.7722, -0.2015)	1
4	(0, 0, 1)	(-1.675, -1.132, -0.1049)	(-1.774, -0.1216, 0.03258)	2
5	(0, 0, 1)	(-1.675, -1.132, -0.1049)	(-1.774, -0.1216, 0.03258)	2
6	(0, 0, 1)	(-1.675, -1.132, -0.1049)	(-1.774, -0.1216, 0.03258)	2
7	(0, 0, 1)	(-1.675, -1.132, -0.1049)	(-1.774, -0.1216, 0.03258)	2
8	(0, 0, 0)	(-1.961, 0.2829, 0.2014)	(0.2801, -1.005, -0.06861)	1

Similarly, for the test vector x_9, Table 3 shows the first three dimensions of the principal component vectors obtained by transforming it into (a) a principal component vector z_9 using the linear PCA transform obtained from training, and (b) a nonlinear principal component vector y_9 using the nonlinear KPCA transform obtained obtained from training. The vector similarities in the KPCA-transformed space can be quite different from those in the PCA-transformed space. This causes the KPCA-based model to be able to make the correct

Table 3: Testing vector (showing only the first three dimensions) and its first three principal components as transformed via the trained PCA and KPCA parameters. The PCA-based and KPCA-based sense class predictions disagree.

	Observed vectors	**PCA-transformed vectors**	**KPCA-transformed vectors**	**Predicted Class**	**Correct Class**
t	(x_t^1, x_t^2, x_t^3)	(z_t^1, z_t^2, z_t^3)	(y_t^1, y_t^2, y_t^3)	$\hat{c}_t$	c_t
9	(1, 0, 1)	(-0.3671, -0.5658, -0.2392)		2	1
9	(1, 0, 1)		(4e-06, 8e-07, 1.111e-18)	1	1

class prediction, whereas the PCA-based model makes the wrong class prediction.

What permits KPCA to apply stronger generalization biases is its implicit consideration of *combinations* of feature information in the data distribution from the high-dimensional training vectors. In this simplified illustrative example, there are just five input dimensions; the effect is stronger in more realistic high dimensional vector spaces. Since the KPCA transform is computed from unsupervised training vector data, and extracts generalizations that are subsequently utilized during supervised classification, it is possible to combine large amounts of unsupervised data with reasonable smaller amounts of supervised data.

Interpreting this example graphically can be illuminating even though the interpretation in three dimensions is severely limiting. Figure 1(a) depicts the eight original observed training vectors x_t in the first three of the five dimensions; note that among these eight vectors, there happen to be only four unique points when restricting our view to these three dimensions. Ordinary linear PCA can be straightforwardly seen as projecting the original points onto the principal axis, as can be seen for the case of the first principal axis in Figure 1(b). Note that in this space, the sense 2 instances are surrounded by sense 1 instances. We can traverse each of the projections onto the principal axis in linear order, simply by visiting each of the first principal components z_t^1 along the principle axis in order of their values, i.e., such that

$$z_1^1 \leq z_8^1 \leq z_4^1 \leq z_5^1 \leq z_6^1 \leq z_7^1 \leq z_2^1 \leq z_3^1 \leq z_9^1$$

It is significantly more difficult to visualize the non-linear principal components case, however. Note that in general, there may not exist *any* principal axis in X, since an inverse mapping from F may not exist. If we attempt to follow the same procedure to traverse each of the projections onto the first principal axis as in the case of linear PCA, by considering each of the first principal components y_t^1 in order of their value, i.e., such that

$$y_4^1 \leq y_5^1 \leq y_6^1 \leq y_7^1 \leq y_9^1 \leq y_1^1 \leq y_8^1 \leq y_3^1 \leq y_2^1$$

then we must arbitrarily select a "quasi-projection" direction for each y_t^1 since there is no actual principal axis toward which to project. This results in a "quasi-axis" roughly as shown in Figure 1(c) which, though not precisely accurate, provides some idea as to how the non-linear generalization capability allows the data points to be grouped by principal components reflecting nonlinear patterns in the data distribution, in ways that linear

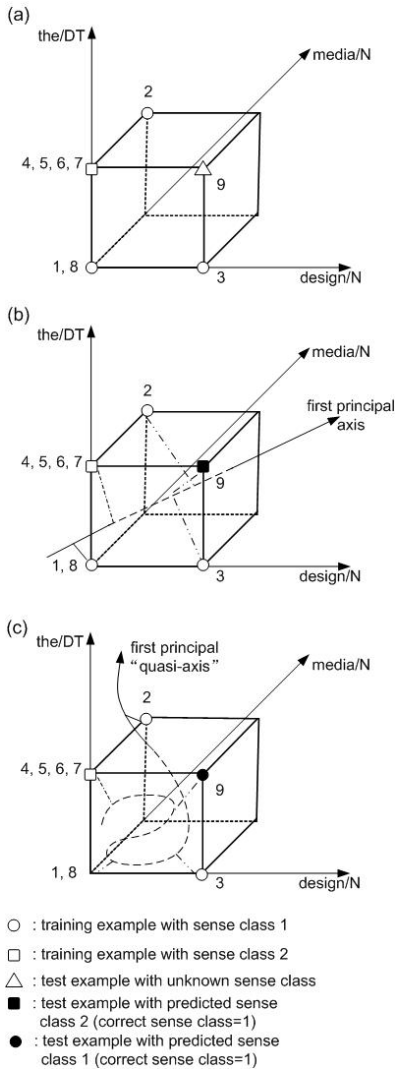

Figure 1: Original vectors, PCA projections, and KPCA "quasi-projections" (see text).

PCA cannot do. Note that in this space, the sense 1 instances are already better separated from sense 2 data points. Moreover, unlike linear PCA, there may be up to M of the "quasi-axes", which may number far more than five. Such effects can become pronounced in the high dimensional spaces are actually used for real word sense disambiguation tasks.

3.3 Algorithm

To extract nonlinear principal components efficiently, note that in both Equations (5) and (6) the explicit form of $\Phi(x_i)$ is required only in the form of $(\Phi(x_i) \cdot \Phi(x_j))$, i.e., the dot product of vectors in F. This means that we can calculate the nonlinear principal components by substituting a kernel function $k(x_i, x_j)$ for $(\Phi(x_i) \cdot \Phi(x_j))$ in Equations (5) and (6) without knowing the mapping Φ explicitly; instead, the mapping Φ is implicitly defined by the kernel function. It is always possible to construct a mapping into a space where k acts as a dot product so long as k is a continuous kernel of a positive integral operator (Schölkopf *et al.*, 1998).

Thus we train the KPCA model using the following algorithm:

1. Compute an $M \times M$ matrix $\hat{K}$ such that

$$\hat{K}_{ij} = k(x_i, x_j) \tag{7}$$

2. Compute the eigenvalues and eigenvectors of matrix $\hat{K}$ and normalize the eigenvectors. Let $\hat{\lambda}_1 \geq \hat{\lambda}_2 \geq \ldots \geq \hat{\lambda}_M$ denote the eigenvalues and $\hat{\alpha}^1$,..., $\hat{\alpha}^M$ denote the corresponding complete set of normalized eigenvectors.

To obtain the sense predictions for test instances, we need only transform the corresponding vectors using the trained KPCA model and classify the resultant vectors using nearest neighbors. For a given test instance vector x, its *l*th nonlinear principal component is

$$y_t^l = \sum_{i=1}^{M} \hat{\alpha}_i^l k(x_i, x_t) \tag{8}$$

where $\hat{\alpha}_i^l$ is the *i*th element of $\hat{\alpha}^l$.

For our disambiguation experiments we employ a polynomial kernel function of the form $k(x_i, x_j) = (x_i \cdot x_j)^d$, although other kernel functions such as gaussians could be used as well. Note that the degenerate case of $d = 1$ yields the dot product kernel $k(x_i, x_j) = (x_i \cdot x_j)$ which covers linear PCA as a special case, which may explain why KPCA always outperforms PCA.

4 Semi-supervised KPCA model

4.1 Utilizing unlabeled data

In WSD, as with many NLP tasks, features are often interdependent. For example, the features that represent words that frequently co-occur are typically highly interdependent. Similarly, the features that represent synonyms tend to be highly interdependent.

It is a strength of the KPCA-based model that it generalizes over combinations of interdependent features. This enables the model to predict the correct sense even when the context surrounding a target word has not been previously seen, by exploiting the similarity to feature combinations that *have* been seen.

However, in practice the labeled training corpus for WSD is typically relatively small, and does not yield enough training instances to reliably extract dependencies between features. For example, in the Senseval-2 English lexical sample data, for each target word there are only about 120 training instances on average, whereas on the other hand we typically have thousands of features for each target word.

The KPCA model can fail when it encounters a target word whose context contains a combination of features that may in fact be interdependent, but are not similar to any combinations that occurred in the limited amounts of labeled training data. Because of the sparse data, the KPCA model wrongly considers the context of the target word to be *dissimilar* to those previously seen—even though the contexts may in truth be similar. In the absence of any contexts it believes to be similar, the model therefore tends simply to predict the most frequent sense.

The potential solution we propose to this problem is to add much larger quantities of unannotated data, with which the KPCA model can first be trained in unsupervised fashion. This provides a significantly broader dataset from which to generalize over combinations of dependent features. One of the advantages of our WSD model is that during KPCA training, the sense class is not taken into consideration. Thus we can take advantage of the vast amounts of cheap unannotated corpora, in addition to the relatively small amounts of labeled training data. Adding a large quantity of unlabeled data makes it much likelier that dependent features can be identified during KPCA training.

4.2 Algorithm

The primary difference of the semi-supervised KPCA model from the supervised KPCA baseline model described above lies in the eigenvector calculation step. As we mentioned earlier, in KPCA-based model, we need to calculate the eigenvectors of matrix K, where $K_{ij} = (\Phi(x_i) \cdot \Phi(x_j))$. In the supervised KPCA model, training vectors such as x_i and x_j are only drawn from the labeled training corpus. In the semi-supervised KPCA model, training vectors are drawn from both the labeled training corpus and a much larger unlabeled training corpus. As a consequence, the maximum number of eigenvectors in the supervised KPCA model is the minimum of the number of features and the number of vectors from the labeled training corpus, while the maximum number of eigenvectors for the semi-supervised KPCA model is the minimum of the number of features and total number of vectors from the combined labeled and unlabeled training corpora.

However, one would not want to apply the semi-supervised KPCA model indiscriminately. While it can be expected to be valuable in cases where the data was too sparse for reliable training of the supervised KPCA model, at the same time it is important to note that the unlabeled data is typically drawn from quite different distributions than the labeled data, and may therefore be expected to introduce a new source of noise.

We therefore define a *composite* semi-supervised KPCA model based on the following assumption. If we are sufficiently confident about the prediction made by the supervised KPCA model as to the predicted sense

for the target word, we need not resort to the semi-supervised KPCA method. On the other hand, if we are not confident about the supervised KPCA model's prediction, we then turn to the semi-supervised KPCA model and take its classification as the predicted sense.

Specifically, the composite model uses the following algorithm to combine the sense predictions of the supervised and semi-supervised KPCA models in order to disambiguate the target word in a given test instance x:

1. **let** s_1 be the predicted sense of x using the supervised KPCA baseline model
2. **let** c be the similarity between x and its most similar training instance
3. **if** $c \geq t$ or $s_1 \neq s_{\text{mf}}$ (where t is a preset threshold, and s_{mf} is the most frequent sense of the target word):
    - **then** predict the sense of the target word of x to be s_1
    - **else** predict the sense of the target word of x to be s_2, the sense predicted by the semi-supervised KPCA model

The two conditions checked in step 3 serve to filter those instances where the supervised KPCA baseline model is confident enough to skip the semi-supervised KPCA model. In particular:

- The threshold t specifies a minimum level of the supervised KPCA baseline model's confidence, in terms of similarity. If $c \geq t$, then there were training instances that were of sufficient similarity to the test instance so that the model can be confident that a correct disambiguation can be predicted based only on those similar training instances. In this case the semi-supervised KPCA model is not needed.
- If s_1 is not the most frequent sense s_{mf} of the target word, then there is strong evidence that the test instance should be disambiguated as s_1 because this is overriding an otherwise strong tendency to disambiguate the target word to the most frequent sense. Again, in this case the semi-supervised KPCA model should be avoided.

The threshold t is defined to rise as the relative frequency of the most frequent sense falls. Specifically, $t = 1 - P(s_{\text{mf}}) + c$ where $P(s_{\text{mf}})$ is the probability of most frequent sense in the training corpus and c is a small constant. This reflects the assumption that the higher the probability of the most frequent sense, the less likely that a test instance disambiguated as the most frequent sense is wrong.

5 Experimental setup

We evaluated the composite semi-supervised KPCA model using data from the Senseval-2 English lexical sample task (Kilgarriff, 2001)(Palmer *et al.*, 2001). We chose to focus on verbs, which have proven particularly difficult to disambiguate. Our task consists in disambiguating several instances of 16 different target verbs.

Table 4: The semi-supervised KPCA model outperforms supervised naïve Bayes and maximum entropy models, as well as the most-frequent-sense and supervised KPCA baseline models.

	Fine-grained accuracy	Coarse-grained accuracy
Most frequent sense	41.4%	51.7%
Naïve Bayes	55.4%	64.2%
Maximum entropy	54.9%	64.1%
Supervised KPCA	57.0%	66.6%
Composite semi-supervised KPCA	**57.4%**	**67.2%**

For each target word, training and test instances manually tagged with WordNet senses are available. There are an average of about 10.5 senses per target word, ranging from 4 to 19. All our models are evaluated on the Senseval-2 test data, but trained on different training sets. We report accuracy, the number of correct predictions over the total number of test instances, at two different levels of sense granularity.

The supervised models are trained on the Senseval-2 training data. On average, 137 annotated training instances per target word are available.

In addition to the small annotated Senseval-2 data set, the semi-supervised KPCA model can make use of large amounts of unannotated data. Since most of the Senseval-2 verb data comes from the Wall Street Journal, we choose to augment the Senseval-2 data by collecting additional training instances from the Wall Street Journal Tipster corpus. In order to minimize the noise during KPCA learning, we only extract the sentences in which the target word occurs. For each target word, up to 1500 additional training instances were extracted. The resulting training corpus for the semi-supervised KPCA model is more than 10 times larger than the Senseval-2 training set, with an average of 1637 training instances per target word.

The set of features used is as described by Yarowsky and Florian (2002) in their "feature-enhanced naïve Bayes model", with position-sensitive, syntactic, and local collocational features.

6 Results

Table 4 shows that the composite semi-supervised KPCA model improves on the high-performance supervised KPCA model, for both coarse-grained and fined-grained sense distinctions. The supervised KPCA model significantly outperforms a naïve Bayes model, and a maximum entropy model, which are among the top performing models for WSD. Note that these results are consistent with the larger study of supervised models conducted by Wu *et al.* (2004). The composite semi-supervised KPCA model outperforms all of the three supervised models, and in particular, it further improves the

Table 5: Semi-supervised KPCA is not necessary when supervised KPCA is very confident.

	Fine-grained accuracy	Coarse-grained accuracy
Supervised KPCA	**62.1%**	**71.3%**
Semi-supervised KPCA	57.1%	67.1%

Table 6: Semi-supervised KPCA outperforms supervised KPCA when supervised KPCA is not confident: adding training data helps when there are no similar instances in the training set.

	Fine-grained accuracy	Coarse-grained accuracy
Supervised KPCA	30.8%	44.11%
Semi-supervised KPCA	**38.3%**	**51.47%**

accuracy of the supervised KPCA model.

Overall, with the addition of the semi-supervised model, the accuracy for disambiguating the verbs increases from 57% to 57.4% on the fine-grained task, and from 66.6% to 67.2% on the coarse-grained task.

In our composite model, the supervised KPCA model predicts senses with high confidence for more than 94% of the test instances. The predictions of the semi-supervised model are used for the remaining 6% of the test instances. Table 5 shows that it is not necessary to use the semi-supervised training model for all the training instances. In fact, when the supervised model is confident, its predictions are significantly more accurate than those of the semi-supervised model alone.

When the predictions of the supervised KPCA model are not accurate, the semi-supervised KPCA model outperforms the supervised model. This happens when (1) there is no training instance that is very similar to the test instance considered and when (2) in the absence of relevant features to learn from in the small annotated training set, the supervised KPCA model can only predict the most frequent sense for the current target. In these conditions, our experiment results in Table 6 confirm that the semi-supervised KPCA model benefits from the large additional training data, suggesting it is able to learn useful feature conjunctions, which help to give better predictions.

The composite semi-supervised KPCA model therefore chooses the best model depending on the degree of confidence of the supervised model. All the KPCA weights, for both the supervised and the semi-supervised model, have been pre-computed during training, and it is therefore inexpensive to switch from one model to the other at testing time.

7 Conclusion

We have proposed a new composite semi-supervised WSD model based on the Kernel PCA technique, that employs both supervised and semi-supervised components. This strategy allows us to combine large amounts of cheap unlabeled data with smaller amounts of labeled data. Experiments on the hard-to-disambiguate verbs from the Senseval-2 English lexical sample task confirm that when the supervised KPCA model is insufficiently confident in its sense predictions, taking advantage of the semi-supervised KPCA model trained with the unlabeled data can help to give a better prediction. The composite semi-supervised KPCA model exploits this to improve upon the accuracy of the supervised KPCA model introduced by Wu *et al.* (2004).

References

Martin Chodorow, Claudia Leacock, and George A. Miller. A topical/local classifier for word sense identification. *Computers and the Humanities*, 34(1-2):115–120, 1999. Special issue on SENSEVAL.

Hoa Trang Dang and Martha Palmer. Combining contextual features for word sense disambiguation. In *Proceedings of the SIGLEX/SENSEVAL Workshop on Word Sense Disambiguation: Recent Successes and Future Directions*, pages 88–94, Philadelphia, July 2002. SIGLEX, Association for Computational Linguistics.

Konstantinos I. Diamantaras and Sun Yuan Kung. *Principal Component Neural Networks*. Wiley, New York, 1996.

Adam Kilgarriff and Joseph Rosenzweig. Framework and results for English Senseval. *Computers and the Humanities*, 34(1):15–48, 1999. Special issue on SENSEVAL.

Adam Kilgarriff. English lexical sample task description. In *Proceedings of Senseval-2, Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 17–20, Toulouse, France, July 2001. SIGLEX, Association for Computational Linguistics.

Dan Klein and Christopher D. Manning. Conditional structure versus conditional estimation in NLP models. In *Proceedings of EMNLP-2002, Conference on Empirical Methods in Natural Language Processing*, pages 9–16, Philadelphia, July 2002. SIGDAT, Association for Computational Linguistics.

Taku Kudo and Yuji Matsumoto. Fast methods for kernel-based text analysis. In *Proceedings of the 41set Annual Meeting of the Asoociation for Computational Linguistics*, pages 24–31, 2003.

S. Mika, B. Schölkopf, A. Smola, K.-R. Müller, M. Scholz, and G. Rätsch. Kernel PCA and de-noising in feature spaces. *Advances in Neural Information Processing Systems*, 1999.

Raymond J. Mooney. Comparative experiments on disambiguating word senses: An illustration of the role of bias in machine learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Philadelphia, May 1996. SIGDAT, Association for Computational Linguistics.

Martha Palmer, Christiane Fellbaum, Scott Cotton, Lauren Delfs, and Hoa Trang Dang. English tasks: All-words and verb lexical sample. In *Proceedings of Senseval-2, Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 21–24, Toulouse, France, July 2001. SIGLEX, Association for Computational Linguistics.

Ted Pedersen. Machine learning with lexical features: The Duluth approach to SENSEVAL-2. In *Proceedings of Senseval-2, Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 139–142, Toulouse, France, July 2001. SIGLEX, Association for Computational Linguistics.

Bernhard Schölkopf, Alexander Smola, and Klaus-Rober Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5), 1998.

C. J. Twining and C. J. Taylor. Kernel principal component analysis and the construction of non-linear active shape models. In *Proceedings of BMVC20001*, 2001.

Dekai Wu, Weifeng Su, and Marine Carpuat. A Kernel PCA method for superior word sense disambiguation. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, Barcelona, July 2004.

David Yarowsky and Radu Florian. Evaluating sense disambiguation across diverse parameter spaces. *Natural Language Engineering*, 8(4):293–310, 2002.