IMPROVING THE PERFORMANCE OF BAYESIAN AND SUPPORT VECTOR CLASSIFIERS IN WORD SENSE DISAMBIGUATION USING POSITIONAL INFORMATION

Tapio Pahikkala, Sampo Pyysalo, Jorma Boberg, Aleksandr Mylläri and Tapio Salakoski

Department of Information Technology, University of Turku and
Turku Centre for Computer Science TUCS,
Lemminkäisenkatu 14 A FIN-20520 Turku, FINLAND, firstname.lastname@it.utu.fi

ABSTRACT

We explore word position-sensitive models and their realizations in word sense disambiguation tasks when using Naive Bayes and Support Vector Machine classifiers. It is shown that a straightforward incorporation of word positional information fails to improve the performance of either method on average. However, we demonstrate that our special kernel that takes into account word positions statistically significantly improves the classification performance. For Support Vector Machines, we apply this kernel instead of the ordinary Bag-of-Words kernel, and for the Bayes classifier the kernel is used for smoothed density estimation. We discuss the benefits and drawbacks of position-sensitive and kernel-smoothed models as well as analyze and evaluate the effects of these models on a subset of the Senseval-3 data.

1. INTRODUCTION

Word Sense Disambiguation (WSD) is the process of resolving the meaning of a word by deciding between a number of possible word senses. A word is said to be polysemous if it has several senses. The word "bank" is a traditional example of a polysemous word: "bank" can refer to a river bank, a financial institution, or the building where a financial institution resides, for example.

In order to disambiguate the sense of an ambiguous word, any WSD method has to incorporate information about its context, that is, the words surrounding it in the text. The Bag-of-Words (BoW) model is a typical choice for many text classification tasks, including WSD tasks. In the BoW model, the order of the words is discarded and only the number of occurrences of each word is taken into account. Several alternative models that (partly) preserve word order information have been proposed, including models using N-gram and collocation features. Moreover, sequence-based similarity measures such as word sequence kernels (Cancedda et al. (2003)) have been proposed. In WSD tasks, the BoW model discards not only the order of the words, but also information on the positions of the words with respect to the word to be disambiguated. General sequence-based features and similarity measures also fail to take into account this kind of information.

In this paper, we consider models that take into account word positional information. A straightforward way to incorporate the word positions is to represent each word of the context together with its position and to consider these word-position pairs as distinct features (see e.g. Audibert (2004)). We will refer this approach to as basic word position-sensitive (BP) model. Ginter et al. (2004) presented a weighted BoW approach, where the context words are weighted in such a way that the words closer to the ambiguous word receive higher values, motivated by the assumption that closer words are more relevant for disambiguation. In Pahikkala et al. (2005), we introduced a position-sensitive kernel function which generalizes the aforementioned approaches. Using context-sensitive spelling error correction as a model WSD problem, it was demonstrated that this kernel can improve the performance of the Support Vector Machine (SVM) classifier in natural language disambiguation tasks. In this paper, we further analyze this kernel function and consider two models, which we will here call the smoothed word position-sensitive (SP) and smoothed word position and distance sensitive (SPD) models. For the Naive Bayes classifier, these models are realized as data representations using kernel density estimation techniques to obtain class conditional probabilities of word-position features. For the SVMs, these models are realized as kernel functions.

We argue that word positional information can play an important role in WSD tasks and explore the use of this information in the two popular classifiers. The classification performance of the Naive Bayes classifier is evaluated with the BoW, BP, SP and SPD representations on a subset of the Senseval-3 data (Mihalcea et al. (2004)) and compared to the performance of SVMs with corresponding kernel functions.

This paper is organized as follows. In Section 2 we introduce the definition of a context and present the Naive Bayes and SVM classifiers. In Section 3 the data used in our experiments and the performance evaluation criteria are presented. Section 4 introduces the proposed models and their realizations with the Naive Bayes and SVM classifiers. We also discuss the benefits and drawbacks of the models. In Section 5 results on Senseval-3 test data are presented and discussed. Finally, in Section 6 we summa-

rize the results and present some ideas for improving the models.

2. BINARY CLASSIFICATION WITH NAIVE BAYES AND SVM CLASSIFIERS

We consider WSD as a binary classification task, in which the training set S is comprised of m labeled examples $(x_1, y_1), \ldots, (x_m, y_m)$, where $x_i \in X$ are training data points and $y_i \in Y$, $Y = \{-1, +1\}$, are the corresponding class labels. In our case, the training set S is $\{(\overline{\tau}_1, y_1), \ldots, (\overline{\tau}_m, y_m)\}$, where $y_i \in \{-1, +1\}$ correspond to the word senses, and the contexts $\overline{\tau}_i$ are defined as follows. Let τ denote a word to be disambiguated and let $\overline{\tau} = (\tau_{-t}, \ldots, \tau_{-1}, \tau_0, \tau_1, \ldots, \tau_r)$, $\tau_0 = \tau$, be the context of τ. The words preceding τ are $\tau_{-t}, \ldots, \tau_{-1}$ in the order they appear in the text, and correspondingly $\tau_1, \ldots, \tau_r$ are the words which follow τ in the text. For a word τ_p, the index p is referred to as its position. Next we define the effect of a context span parameter s when considering contexts. For fixed s, we take always the largest context $\overline{\tau} = (\tau_{-t}, \ldots, \tau_{-1}, \tau_0, \tau_1, \ldots, \tau_r)$ so that $t \leq s$ and $r \leq s$. Note that if there exist s words preceding and following the word to be disambiguated, then $t = r = s$, otherwise $t < s$ or $r < s$. Furthermore, let V be a set of all distinct words of all the contexts in the training set.

2.1. Naive Bayes Classifier

Let W be the set of all the features of $\overline{\tau}$. These features depend on the data representations considered in Section 4. For the Naive Bayes classifier, we use the following decision function:

$$f(\overline{\tau}) = P(+1) \prod_{w \in W} P(w|+1) - P(-1) \prod_{w \in W} P(w|-1), \tag{1}$$

where $P(w|+1)$ and $P(w|-1)$ are the probabilities that the feature w appears in a positive and in a negative example, respectively, and $P(+1)$ and $P(-1)$ are the prior probabilities of the positive and negative classes.

The probabilities can be directly estimated from the training data using maximum likelihood estimation (MLE) as follows. For each class $y \in Y$ and feature $w \in W$,

$$P(y) = \frac{\mathrm{N}(y)}{\sum_{y' \in Y} \mathrm{N}(y')}, \tag{2}$$

$$P(w|y) = \frac{\mathrm{N}(w, y)}{\sum_{w' \in W} \mathrm{N}(w', y)}, \tag{3}$$

where $\mathrm{N}(y')$ is the number of examples in the class $y' \in Y$, and $\mathrm{N}(w, y)$ is the number of times feature w appears in the examples of the class y. The MLE estimates are typically smoothed to avoid zero probabilities in prediction; in this paper we use Add-one smoothing, where all numbers of feature occurrences are incremented by one over the counted value (see e.g. Chen and Goodman (1996)).

2.2. SVM Classifier

We consider SVMs as a special case of the following regularization problem known as Tikhonov regularization (for a more comprehensive introduction, see e.g. Rifkin (2002); Vapnik (1998)):

$$\min_f \sum_{i=1}^{m} l(f(\overline{\tau}_i), y_i) + \lambda \|f\|_k^2, \tag{4}$$

where l is the loss function used by the learning machine, $f : X \to Y$ is a function which maps the input vectors $x \in X$ to the output labels $y \in Y$, $\lambda \in \mathbb{R}_+$ is a regularization parameter, and $\|\cdot\|_k$ is the norm in the Reproducing Kernel Hilbert Space defined by a positive definite kernel function k. The second term is called a regularizer. With SVMs we use linear soft margin loss function (also called hinge loss):

$$l(f(\overline{\tau}), y) = \max(1 - yf(\overline{\tau}), 0).$$

By the Representer Theorem, the minimizer of (4) has the following form:

$$f(\overline{\kappa}) = \sum_{i=1}^{m} a_i k(\overline{\kappa}, \overline{\tau}_i),$$

where $a_i \in \mathbb{R}$ and k is the kernel function associated with the Reproducing Kernel Hilbert Space mentioned above.

Kernel functions are similarity measures of data points in the input space X, and they correspond to an inner product in a feature space H to which the input space data points are mapped. Formally, kernel functions are defined as

$$k(\overline{\tau}, \overline{\kappa}) = \langle \Phi(\overline{\tau}), \Phi(\overline{\kappa}) \rangle,$$

where $\Phi : X \to H$. The input space X can be any set, in our case, it is the set of contexts.

Fow SVMs, we use the so called normalized kernel

$$\tilde{k}(\overline{\tau}, \overline{\kappa}) = \frac{k(\overline{\tau}, \overline{\kappa})}{\sqrt{k(\overline{\tau}, \overline{\tau}) k(\overline{\kappa}, \overline{\kappa})}}$$

in order to normalize the data in the feature space (see e.g. Graf et al. (2003)).

3. PERFORMANCE MEASURE AND EVALUATION DATA

In this section, we describe the dataset used in the experiments and how the performance of the classification methods with various representations and kernels was measured.

3.1. Measure of performance

We measure the performance of the classifiers using the area under the ROC curve (AUC) (see e.g. Fawcett (2003)). The ROC curve is a relation between the true-positive and the false-positive rates at various classification thresholds. ROC is preferable to other popular measures, such as accuracy and precision-recall analysis, because it captures classifier performance over the whole

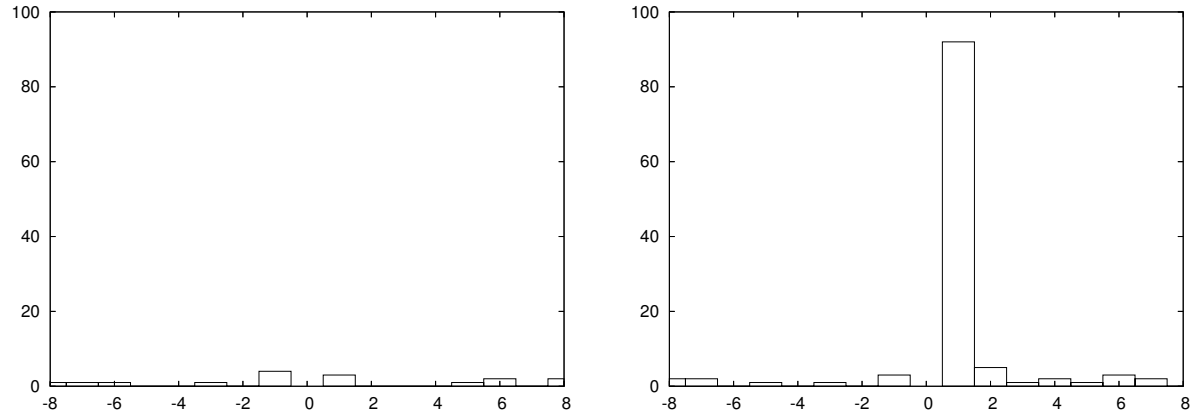

Figure 1. Number of occurrences of the word "to" at different positions (x-axis) in the context of the word "appear" in the sense "to come into view" (left) and "to seem" (right). The occurrence of the word "to" immediately after the word "appear" is a very strong indicator of the second sense.

dataset instead of a single cutoff point and is invariant with respect to the prior class probabilities. AUC corresponds to the probability that given a randomly chosen positive example and a randomly chosen negative example, the classifier will correctly determine which is which.

3.2. Data used for evaluation

To evaluate the performance of the methods and models, we use the Senseval-3 English lexical sample train and test datasets EnglishLS.train and EnglishLS.test[1] (Mihalcea et al. (2004)), where words are to be disambiguated between senses defined by WordNet (for nouns and adjectives) and Wordsmyth (for verbs).

To facilitate parameter estimation, analysis and the use of the basic AUC measure for performance evaluation, we performed the following simplifying processing steps on the data: we apply the defined sense-mapping giving "coarse" senses, accept only one correct answer per instance, consider only binary classification between the two most common senses, and examine only those WSD tasks where the minority class contains at least 50 instances.

The Senseval-3 dataset consists of separate training and test sets. We used ten times repeated stratified 10-fold cross-validation on the training sets to perform parameter and performance estimation on the various models considered in Section 4, and performed single tests on the test data to validate the results and estimate statistical significance. With the exception of the validation results presented in Section 5, all results discussed below are from the parameter estimation phase.

4. MODELING OF CONTEXTS

In this section, we present and evaluate the various data representations and kernels used with Naive Bayes and SVM classifiers.

[1] Available at http://www.senseval.org/senseval3

4.1. Bag-of-Words model

The common BoW model is used with the two classification methods as follows. For the Naive Bayes classifier, the class conditional probabilities $P(v|y)$ corresponding to (3), that is, the probability that the word v appears in a context belonging to the class y, can be estimated using MLE:

$$P(v|y) \quad = \quad \frac{\mathrm{N}(v,y)}{\sum_{v' \in V} \mathrm{N}(v',y)}, \tag{5}$$

where $\mathrm{N}(v,y)$ denotes how many times the word v has occurred in the contexts with the class y in the training set. For SVMs, we can use the BoW kernel, defined as

$$k(\overline{\tau},\overline{\kappa}) = \sum_{v \in V} \mathrm{N}(v,\overline{\tau})\mathrm{N}(v,\overline{\kappa}), \tag{6}$$

where $\mathrm{N}(v,\overline{\tau})$ is the number of occurrences of the word v in the context $\overline{\tau}$.

4.2. Word position-sensitive models

Next we consider and evaluate the effect of different word position-sensitive representations and kernels on classification performance and demonstrate how an alternative to the strict binary division between position-insensitive and position-sensitive models can overcome data sparseness issues and improve the performance.

4.2.1. Basic word position-sensitive model

Let s be a context span parameter and let $\mathrm{N}(v,p,y)$, where $v \in V$, $-s \leq p \leq s$, $y \in Y$, denote how many times the word v has occurred at position p in the contexts with class y in the training set (see Figure 1 for illustration). When determining $\mathrm{N}(v,p,y)$, we consider only the contexts that have the position p.

For the Naive Bayes classifier, we present the basic word position-sensitive (BP) representation, a straightforward way to incorporate word positional information. The class conditional probability $P(v,p|y)$ corresponding to

(3), that is, the probability that the word v appears at the position p in a context belonging to the class y is estimated as follows:

$$P(v,p|y) = \frac{\mathrm{N}(v,p,y)}{\sum_{v' \in V} \sum_{p'=-s}^{s} \mathrm{N}(v',p',y)}. \quad (7)$$

For SVMs, we can define a BP kernel analogously to the BoW kernel:

$$k(\overline{\tau},\overline{\kappa}) = \sum_{v \in V} \sum_{p=-s}^{s} \mathrm{N}(v,p,\overline{\tau})\mathrm{N}(v,p,\overline{\kappa}), \quad (8)$$

where $\mathrm{N}(v,p,\overline{\tau}) = 1$ if $\overline{\tau}$ has the position p and the word v is at the position p, and otherwise $\mathrm{N}(v,p,\overline{\tau}) = 0$.

Compared to the BoW model, position-sensitive modeling of contexts has an obvious potential advantage: it is capable of capturing differences in the relationship between features and senses with respect to the positions of the words. An illustrative example of this in disambiguating the meaning of the verb "appear" between the senses "to seem" and "to come into view" is the occurrence of the word "to" in the context. While in the position-insensitive BoW model the word "to" is only a relatively weak indicator of the sense "to seem", in the BP model it can be observed that the occurrence of the word "to" immediately after the word to be disambiguated is a very strong indicator of this sense, while occurrences of the word in other positions are not good indicators of either sense (see Figure 1). Though the difference is perhaps exceptionally notable in this example, similar distinctions are likely to be found for other words also. The BP model thus allows the classifiers to distinguish between weak and strong features that would not be considered separate in the BoW model.

While the BP model preserves strictly more information than the BoW model, it has the potential drawback of notably increasing the sparseness of the data. This in turn has the effect of reducing the accuracy of the Naive Bayes maximum likelihood estimates, and diagonalizing the kernel matrix used by SVMs.

	Model	
Classifier	BoW	BP
Bayes	**85.2**	80.6
SVM	**83.6**	81.4

Table 1. Performance with the BoW and BP models.

The performance of the classifiers with these two models is given in Table 1. The performances are averaged over the datasets, where the optimal context span s is selected separately for both classifiers, both models and each dataset from $2^0, 2^1, \ldots, 2^8$. The BP model decreases the performance of both of the methods. This suggests that on average the potential performance benefits of the BP model are overweighed by the drawbacks discussed above.

4.2.2. Smoothed word position-sensitive model

By introducing the smoothed word position-sensitive (SP) model, we aim to identify intermediates between the opposites of the position-insensitive BoW model and the BP model. Intuitively, SP relaxes the requirement of BP that words must occur exactly at the same position to be considered as the same feature. We will now consider means to realize the SP model when using the Naive Bayes and SVM classifiers.

John and Langley (1995) suggest to use kernel density estimation (we refer to Silverman (1986) for more information on kernel density estimation) when estimating continuous variables for Bayesian classifiers. While the word-position random variable is discrete, and hence a histogram is a natural way to estimate its density, the estimate can still be bumpy because of the lack of training data. This problem can be solved by using a Parzen density estimate instead. A popular way to do it is to use a Gaussian kernel,

$$g(p,q) = e^{-\theta(p-q)^2}, \quad (9)$$

whose width is controlled by the parameter θ. The estimate of the class conditional probability of a certain word-position pair is then a modification of (3):

$$P(v,p|y) = \frac{\sum_{q=-s}^{s} \mathrm{N}(v,q,y)g(p,q)}{\sum_{v' \in V} \sum_{p',q'=-s}^{s} \mathrm{N}(v',q',y)g(p',q')}, \quad (10)$$

that is, the estimate is a convolution of the sample empirical distribution of the word position with the Gaussian kernel (see e.g. Hastie et al. (2001)). Note that the add-one smoothing described in Section 2.1 is performed for each word-position feature after the Parzen density estimate is made and the normalization is then performed over all word-position pairs in the class y.

For SVMs, the SP kernel is defined as

$$k(\overline{\tau},\overline{\kappa}) = \sum_{v \in V} \sum_{p,q=-s}^{s} \mathrm{N}(v,p,\overline{\tau})\mathrm{N}(v,q,\overline{\kappa})g(p,q). \quad (11)$$

The parameter θ controls the extent of the smoothing so that for large values of θ the smoothed model approaches the BP model, while for $\theta = 0$ the SP model matches the BoW model. The SP model thus generalizes over both models and allows intermediates between these two extremes.

	Model		
Classifier	BoW	BP	SP
Bayes	85.2	80.6	**86.7**
SVM	83.6	81.4	**85.4**

Table 2. Performance with the BoW, BP and SP model.

Performance with the smoothed model is given in Table 2. For the SP model, we performed a grid search of the parameters s (as above) and θ (on the logarithmic scale, including in addition the values of 0 and ∞). The results

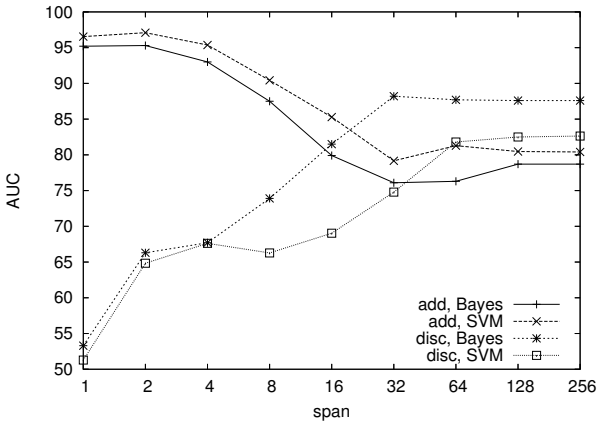

Figure 2. Effect of context span on disambiguation performance with the words "add" and "disc" with Bayes and SVM using the BoW model.

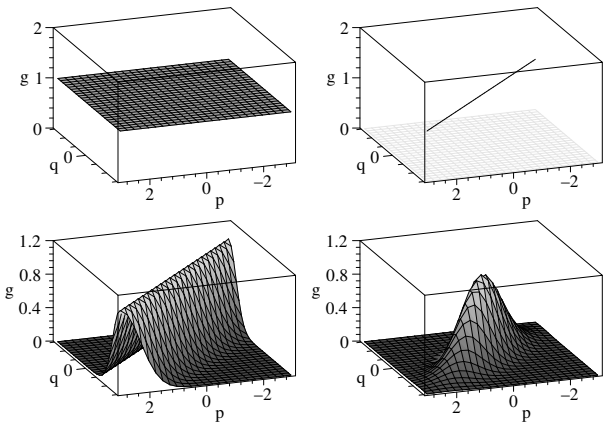

Figure 3. Function $g(p, q)$ corresponding to four models: BoW (top left), BP (top right), SP (bottom left), and SPD (bottom right).

indicate that with an appropriate setting of the θ parameter, the SP model outperforms both the BoW and BP models in the parameter selection phase.

4.2.3. Incorporating distance-based smoothing

In this section, we explore in detail the effect of increasing context size on classification performance and discuss the incorporation of positional information also from words that are far from the word to be disambiguated using distance-based smoothing.

The size of the context has a well-documented effect on the performance of WSD methods (see e.g. Yarowsky and Florian (2002)). As in the case of the choice between position-insensitive and position-sensitive models, there are both intuitive benefits and drawbacks for increasing the context size.

The words that are closest to the word to be disambiguated are likely to be more important than words that are farther in the context. Thus, limiting the size of the context may allow the classification method to better focus on the most relevant features and decreases the amount of noise. On the other hand, limiting the size of the context increases the sparseness of the data. Further, even very distant words can be relevant when deciding the correct sense, especially in cases where the one sense per discourse assumption (see e.g. Yarowsky (1995)) holds.

A balance between the positive and negative effects of large contexts can be found by estimating performance for several context sizes and selecting the cutoff size that gives best performance. As illustrated in Figure 2, this optimum can vary greatly depending on the problem: for the word "disc", the performance is poor for very short contexts and improves almost monotonically with increasing the context size. Conversely, for "add", performance peaks at the short context size of 2 and drops notably as the size increases. The overall effect of the context size is similar for both Bayes and SVM with these problems when using the BoW model.

While short contexts discard most of the potentially available information in the context, short contexts may nevertheless represent real optima for models such as BoW (as for the word "add" above). Indeed, for both classifiers and all three models considered above, average performance over all ambiguous words peaks at remarkably low values of the context span parameter – between two and eight – suggesting that none of the models allows the machine learning method to benefit from information carried by distant words (see Figure 5 below). Therefore, we next consider a model where the distant words may contribute to the disambiguation performance.

Using a fixed cutoff makes the implicit assumption that the words within the cutoff distance are all equally important, and words that are further carry no importance. To more accurately capture the intuition that the importance of words decreases smoothly with distance from the word to be disambiguated, we can adopt a model where the contribution of the words is smoothed according to this distance. To perform this smoothing, we use the function

$$g(p, q) = e^{-\alpha(p^2+q^2)} + \beta, \tag{12}$$

where p and q are distances from the word to be disambiguated. The parameter α controls the effect of the distances and β defines a "minimum weight" given to words at any distance.

This distance-based smoothing function can be combined with the position-based smoothing defined above, yielding the function

$$g(p, q) = e^{-\alpha(p^2+q^2)-\theta(p-q)^2} + \beta. \tag{13}$$

The model that we obtain when using this function in (10) and (11) will be referred to as the smoothed word position and distance sensitive (SPD) model. This model generalizes over all three models considered above by choosing the appropriate parameter values. Setting $\alpha = \beta = 0$ corresponds to the SP model, and if we further choose a very large value of θ, this model approaches the BP model. The BoW model is obtained by setting $\alpha = \theta = \beta = 0$ (see

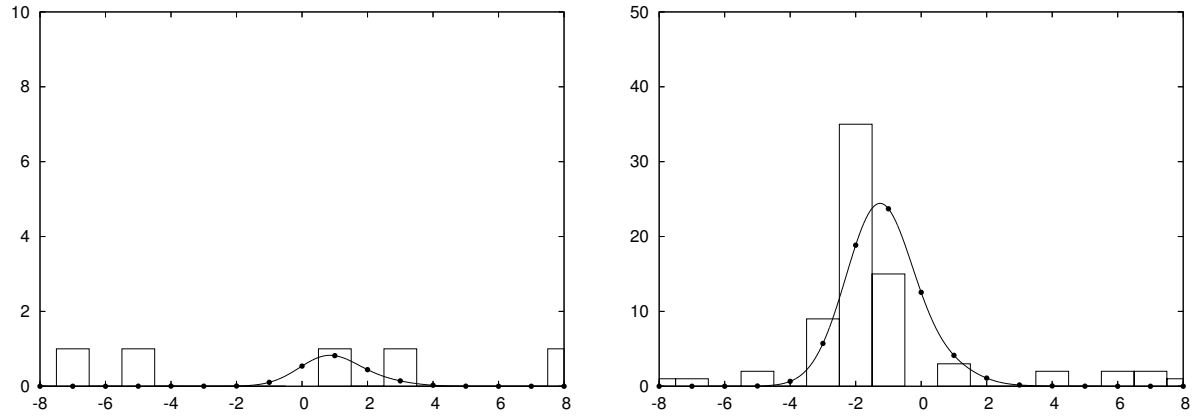

Figure 4. Number of occurrences of "?" (question mark) at different positions (x-axis) in the context of the word "ask" in the sense "to request or seek" (left) and "to question; inquire" (right). The continuous lines are the corresponding kernel smoothed numbers of occurrences. Note that the scales of the y-axis are different in the two plots.

Figure 3). For SVM, using the function (13) corresponds to the SPD kernel we introduced in Pahikkala et al. (2005).

The SPD representation is illustrated in Figure 4 using the occurrences of "?" (question mark) in the context of the ambiguous word "ask". Question marks occur frequently in nearby positions before "ask" in the sense "to question", and occurrences in other positions are relatively rare in either sense. Position-based smoothing spreads the bumps at the nearby positions, while the distance-based smoothing causes the density to vanish at distant positions. Thus, the smoothed numbers of word occurrences may be more useful as question marks occur at far away positions close to random in both senses and hence do not indicate either sense.

	Model			
Classifier	BoW	BP	SP	SPD
Bayes	85.2	80.6	86.7	**87.3**
SVM	84.6	81.4	85.4	**87.7**

Table 3. Performance with the BoW, BP, SP and SPD models.

The performance with separately optimized spans is given in Table 3. The parameters were optimized with a full grid search for s and θ (as above), α (on the logarithmic scale, including in addition the value of 0) and β (from $0.0, 0.02, \ldots, 0.1$). These results indicate that SPD outperforms BoW, BP, and SP in the parameter selection phase.

The average performance of the Naive Bayes and SVM classifiers with the BoW, BP, SP and SPD representations and kernels with respect to the context span is plotted in Figure 5. While the BoW, BP and SP models all fail to benefit from large context spans, the performance with SPD increases almost monotonically with context span. This suggests that with appropriate parameters, the SPD model performs as expected, that is, the words further in the context contribute to disambiguation. In addition, for any choice of the span parameter, SPD outperforms the other considered models in the parameter selection phase.

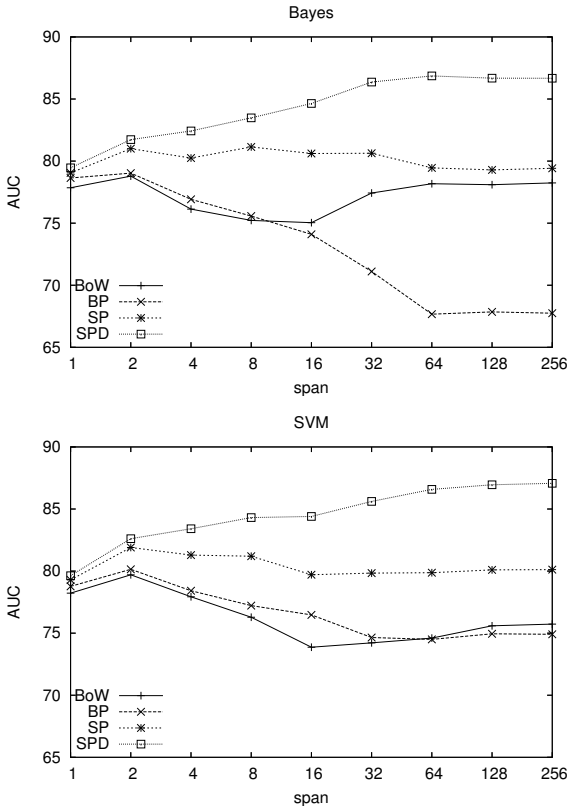

Figure 5. Performance with the BoW, BP, SP and SPD representations and kernels with respect to context span: Naive Bayes (top) and SVM (bottom). Performance is averaged over all datasets. The context span is on a logarithmic scale.

5. EXPERIMENTS ON TEST DATA

In addition to the cross-validation experiments on the EnglishLS.train dataset discussed in the previous sections, we tested the performance of the various models on a sub-

		Representation			
Word	#	BoW	BP	SP	SPD
add.v	125	95.1	96.0	96.0	**96.9**
appear.v	109	88.3	91.9	**93.1**	93.0
argument.n	100	67.1	64.5	66.0	**70.9**
ask.v	109	93.3	96.1	**96.3**	95.4
atmosphere.n	67	**71.0**	58.6	**71.0**	70.9
degree.n	118	94.5	97.2	**97.9**	97.3
disc.n	65	98.9	67.5	98.9	**99.1**
image.n	54	**97.0**	73.1	**97.0**	**97.0**
note.v	64	68.6	**75.8**	75.5	75.3
paper.n	78	84.7	84.5	84.6	**88.2**
performance.n	73	**95.7**	60.1	**95.7**	**95.7**
produce.v	83	**85.5**	70.2	**85.5**	85.4
shelter.n	68	69.7	79.3	**80.1**	79.3
sort.n	92	80.2	**83.2**	80.5	82.1
AVERAGE		85.0	78.4	87.0	**87.6**

		Kernel			
Word	#	BoW	BP	SP	SPD
add.v	125	95.5	95.8	95.8	**97.4**
appear.v	109	88.8	**91.3**	91.1	90.8
argument.n	100	66.1	64.4	64.1	**70.0**
ask.v	109	88.6	94.2	94.7	**94.8**
atmosphere.n	67	62.1	58.8	62.1	**66.7**
degree.n	118	96.1	**98.0**	97.4	96.6
disc.n	65	98.3	68.5	98.3	**98.7**
image.n	54	85.7	74.1	73.9	**88.9**
note.v	64	67.6	66.3	61.6	**71.5**
paper.n	78	80.1	84.3	88.6	**91.6**
performance.n	73	**85.8**	55.7	**85.8**	**85.8**
produce.v	83	**78.6**	69.0	**78.6**	77.9
shelter.n	68	82.8	78.9	**83.5**	82.6
sort.n	92	81.7	**83.4**	82.6	**83.4**
AVERAGE		82.7	77.3	82.7	**85.5**

Table 4. Test results and test set sizes (#) for Naive Bayes (left) and SVMs (right). The results of the best performing models per ambiguous word are typed in bold.

set of the EnglishLS.test dataset, formed as described in Section 3.2. For both classifiers and each model, we chose the parameter combination that resulted in the best performance in cross-validation, and then performed training on the EnglishLS.train dataset and prediction on the EnglishLS.test dataset. Performance was measured as the area under the ROC curve as above, and statistical significance was tested using standard paired two-tailed t-tests.

Table 4 (left) gives the test results for the various representations with the Naive Bayes classifier. As suggested by earlier results, the BP representation is on average notably worse than the BoW representation, by more than 30 percentage units in some cases. Nevertheless, for some words the performance appears to increase even with this basic representation. For the SP representation, the performance is better or equal to the BoW performance for all but two words and statistically significantly better on average ($p < 0.05$). For the SPD representation, the performance is again better than that of the baseline BoW representation for all but two ambiguous words, and the average performance is significantly better ($p < 0.01$).

The test results for SVM are given in Table 4 (right) and mirror the results for Bayes for most cases. The BP kernel performs worse than BoW on average, and particularly notably worse for some of the same words ("disc", "performance") as for Bayes. Surprisingly, the SP kernel only reaches the performance level of BoW; this failure is discussed in more detail below. Similarly to Bayes, the SPD kernel performs statistically significantly better ($p < 0.01$) than the baseline kernel. One notable difference between the Bayes and SVM results is in the relative performance of the SP model. For Bayes, SP significantly outperforms BoW, but the difference between the SP and SPD models is only 0.6 percentage units on average and not statistically significant. For SVM, the SP kernel fails to outperform BoW, and is significantly outperformed by the SPD kernel ($p < 0.05$).

As SP is a generalization that includes both BoW and BP as special cases, cases where SP performs worse than either BoW or BP suggest a failure of the parameter selection strategy. Similarly, as SPD generalizes over SP, cases where its performance is worse than that of any other of the models may suggest that overfitting has occurred. As each of these types of failures occur for both the Naive Bayes and SVM classifiers for several words, the test results presented here indicate that the used parameter selection strategy may not have been appropriate for the small training sets, which consisted on average of 174 examples. In our preliminary experiments, we used stratified 10-fold cross-validation instead of the ten times repeated stratified 10-fold cross-validation for which results are reported here. Without the repetition, we observed even more severe overfitting in parameter estimation. Nevertheless, even the repeated cross-validation strategy failed to select the optimal parameters in many cases.

While the development of alternate parameter selection strategies falls outside the scope of this paper, we note that our previous results suggest that when parameters are appropriately selected, the SPD kernel achieves systematically better results as one could expect: in Pahikkala et al. (2005), using 1000 examples and 10-fold cross-validation in parameter estimation, we observed no notable overfitting and a significant and more substantial performance advantage with the position-sensitive kernels with SVMs. Nevertheless, these test results emphasize an important property generally related to the use of more powerful models; as the capacity of the models increases, so does the risk of overfitting.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we have explored the use of position-sensitive representations and kernels for improving the performance of Bayesian and Support Vector classifiers in word sense disambiguation tasks. We demonstrated

that the basic word position-sensitive (BP) model fails to improve performance, and speculated that the increased sparseness of data using this model may be a main source of this failure. We addressed this issue through the use of the smoothed position-sensitive (SP) model, and found that in the parameter selection phase this model performs better than the BoW and BP models, indeed for any choice of context size. We additionally discussed the effect of increasing context size and explored the use of a smoothed word position and distance sensitive (SPD) model to allow the beneficial incorporation of information from words that are distant from the word to be disambiguated. When validating the models with the test data, we found that while the results indicated some failures in the applied parameter selection strategy, the SPD models achieve statistically significantly better results ($p < 0.01$) than the BoW baseline for both classification methods studied. We expect that using an appropriate parameter selection strategy and sufficiently large data sets, the performance of the SP and SPD models could be further improved. We conclude that position-sensitive models offer a promising alternative to commonly used position-insensitive models and that the model can be used to improve the performance of both Naive Bayes and Support Vector Machine classifiers.

To increase the applicability of the position-sensitive models to small datasets, the study of parameter selection methods may be a useful future direction. Further validation of the performance of the Naive Bayes classifier with the kernel-smoothed position-sensitive representations should also be performed on other datasets and WSD problems. Moreover, as many elements of the SP and SPD models are independent of the features on which they are applied, the new models could be combined with features other than words, such as part-of-speech, collocation, or N-gram features, giving further opportunities to improve classification performance.

7. ACKNOWLEDGEMENTS

This work was supported by TEKES, the Finnish National Technology Agency.

References

Audibert, L. (2004). Word sense disambiguation criteria: a systematic study. In *Proceedings of 20th International Conference on Computational Linguistics*, pages 910–916, Geneva, Switzerland. Association for Computational Linguistics.

Cancedda, N., Gaussier, E., Goutte, C., and Renders, J.-M. (2003). Word-sequence kernels. *Journal of Machine Learning Research*, 3:1059–1082.

Chen, S. F. and Goodman, J. (1996). An empirical study of smoothing techniques for language modeling. In Joshi, A. and Palmer, M., editors, *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, pages 310–318, Santa Cruz, Ca. Association for Computational Linguistics.

Fawcett, T. (2003). Roc graphs: Notes and practical considerations for data mining researchers. Technical Report HPL-2003-4, HP Labs, Palo Alto, Ca.

Ginter, F., Boberg, J., Järvinen, J., and Salakoski, T. (2004). New techniques for disambiguation in natural language and their application to biological text. *Journal of Machine Learning Research*, 5:605–621.

Graf, A., Smola, A., and Borer, S. (2003). Classification in a normalized feature space using support vector machines. *IEEE Transactions on Neural Networks*, 14(3):597–605.

Hastie, T., Tibshirani, R., and Friedman, J. H. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Verlag, New York.

John, G. H. and Langley, P. (1995). Estimating continuous distributions in bayesian classifiers. In Besnard, P. and Hanks, S., editors, *Proceedings of the Eleventh Annual Conference on Uncertainty in Artificial Intelligence*, pages 338–345, San Mateo. Morgan Kaufmann Publishers.

Mihalcea, R., Chklovski, T., and Kilgarriff, A. (2004). The senseval-3 english lexical sample task. In Mihalcea, R. and Edmonds, P., editors, *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 25–28, Barcelona, Spain. Association for Computational Linguistics.

Pahikkala, T., Pyysalo, S., Ginter, F., Boberg, J., Järvinen, J., and Salakoski, T. (2005). Kernels incorporating word positional information in natural language disambiguation tasks. In *Proceedings of the Eighteenth International Florida Artificial Intelligence Research Society Conference (FLAIRS 2005)*, Clearwater Beach, Florida. To appear.

Rifkin, R. (2002). *Everything Old Is New Again: A Fresh Look at Historical Approaches in Machine Learning.* PhD thesis, MIT.

Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis*. Chapman & Hall, London.

Vapnik, V. (1998). *Statistical Learning Theory*. Wiley, New York.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In Uszkoreit, H., editor, *Proceedings of the Thirty-Third conference on Association for Computational Linguistics*, pages 189–196, Cambridge, Massachusetts. Association for Computational Linguistics.

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