

Figure 8: Number of examples needed for average error to reach 0.3. From left to right: random, uncertainty, maximal distance and lookahead sampling methods.

contains more than one region of some class. Then, the selective sampling algorithm must consider not only the examples from the hypothesis boundary, but must also explore large unsampled regions. The lack of 'exploration' element in 'uncertainty' and 'maximal distance' sampling methods often results in a failure in such cases. The benefit of a lookahead selective sampling method can be seen by comparing the number of examples needed to reach some pre-defined accuracy, Figure 8.

Counting the classification of one point (including finding 1 or 2 labeled neighbors) as a basic operation, the uncertainty and maximal distance methods have time complexity of $O(|X|)$ while the straightforward implementation of lookahead selective sampling has a time complexity of $O(|X|^2)$ (we need to compute class probabilities for all points in the instance space after each lookahead hypothesis). This higher complexity, however, is well justified for a natural setup, where we are ready to invest computational resources to save time for a human expert whose role is to label an examples.

References

- Aha, D. W.; Kibler, D.; and Albert, M. K. 1991. Instance-based learning algorithms. *Machine Learning* 6(1):37–66.
- Angluin, D. 1988. Queries and concept learning. *Machine Learning* 2(3):319–42.
- Blake, C.; Keogh, E.; and Merz, C. 1998. UCI repository of machine learning databases [<http://www.ics.uci.edu/~mllearn/MLRepository.html>] University of California, Irvine, Dept. of Information and Computer Sciences.
- Cohn, D. A.; Atlas, L.; and Lander, R. 1994. Improving generalization with active learning. *Machine Learning* 15(2):201–21.
- Cover, T. M., and Hart, P. E. 1967. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory* 13(1):21–27.
- Dagan, I., and Engelson, S. P. 1995. Committee-based sampling for training probabilistic classifiers. In *Machine Learning - International Workshop then Conference - 1995; conf 12*, 150–157. Morgan Kaufmann.
- Davis, D. T., and Hwang, J.-N. 1992. Attentional focus training by boundary region data selection. In *IJCNN*, volume 1, 676–81. IEEE.
- Eldar, Y.; Lindenbaum, M.; Porat, M.; and Zeevi, Y. Y. 1997. The farthest point strategy for progressive image sampling. *IEEE Transactions on Image Processing* 6(9):1305–15.
- Freund, Y.; Seung, H. S.; Shamir, E.; and Tishbi, N. 1997. Selective sampling using the query by committee algorithm. *Machine Learning* 28(2-3):133–68.
- Frey, P. W., and Slate, D. J. 1991. Letter recognition using holland-style adaptive classifiers. *Machine Learning* 6(2):161–82.
- Hasenjager, M., and Ritter, H. 1996. Active learning of the generalized high-low-game. In *ICANN*, xxv+922, 501–6. Springer-Verlag.
- Hasenjager, M., and Ritter, H. 1998. Active learning with local models. *Neural Processing Letters* 7(2):107–17.
- Krogh, A., and Vedelsby, J. 1994. Neural network ensembles, cross validation, and active learning. In *NIPS*, volume 7, 231–8. MIT Press.
- Lang, K. J., and Witbrock, M. J. 1988. Learning to tell two spirals apart. In *Proceedings of the Connectionist Models Summer School*, 52–59. Morgan Kaufmann.
- Lewis, D. D., and Catlett, J. 1994. Heterogeneous uncertainty sampling for supervised learning. In *Machine Learning - International Workshop then Conference - 1994; conf 11*, 148–156.
- Lindenbaum, M.; Markovich, S.; and Rusakov, D. 1999. Selective sampling by random field modelling. Technical Report CIS9906, Technion - Israel Institute of Technology.
- MacKay, D. J. 1998. Introduction to gaussian processes. *NATO ASI series. Series F, Computer and system sciences*. 168:133.
- Papoulis, A. 1991. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill series in electrical engineering, Communications and signal processing. McGraw-Hill, Inc., 3rd edition.
- RayChaudhuri, T., and Hamey, L. 1995. Minimisation of data collection by active learning. In *IEEE ICNN*, volume 3, 6 vol. 1+3219, 1338–41. IEEE.
- Seung, H. S.; Opper, M.; and Sompolinsky, H. 1992. Query by committee. In *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, v+452, 287–94. ACM; New York, NY, USA.
- Williams, C. K. I., and Barber, D. 1998. Bayesian classification with gaussian processes. *IEEE PAMI* 20(12):1342.
- Wong, E., and Hajek, B. 1985. *Stochastic Processes in Engineering Systems*. Springer-Verlag.

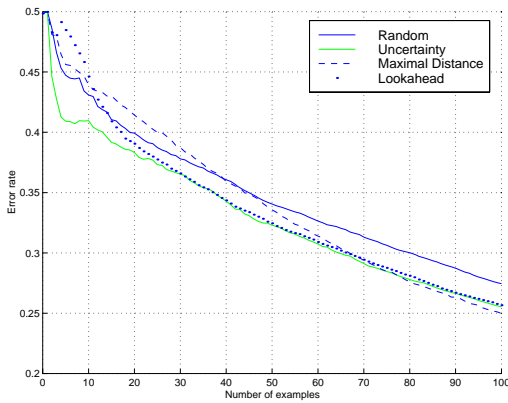


Figure 4: Learning rate graphs for various selective sampling methods applied to the “two spirals” data.

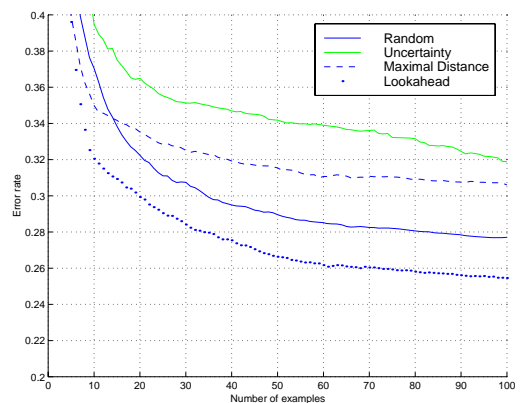


Figure 6: Learning rate graphs for various selective sampling methods applied to the “two gaussians” data.

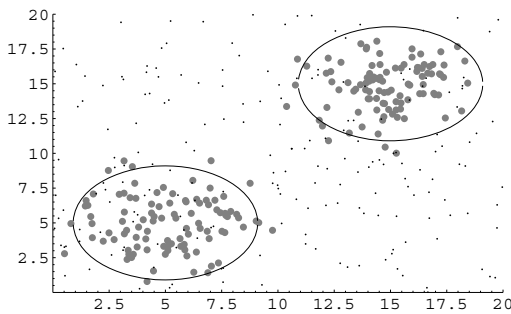


Figure 5: A feature space with bayes decision boundaries (only 400 points are shown) for “two gaussians” data.

’Two Gaussians’ Data

The test set of “two gaussians” consists of two dimensional vectors belonging to two classes with equal a priori probability (0.5). The distribution of class 1 is uniform over the region $[0, 20] \times [0, 20]$ and the distribution of class 0 consists of two symmetric gaussians, with means in points (5, 5) and (15, 15) and covariance matrix $\Sigma = 2^2 I$, illustrated in Figure 5. The bayes error is 0.18207.

The learning rate of the various selective sampling methods is shown in Figure 6. We can see that apparently the uncertainty and maximal distance selective sampling methods fail to detect one of the gaussians, resulting in higher error rates. This is due to fact that these methods consider sampling only at the existing boundary.

Letters Data

The *letter recognition database* (contributed to UCI Machine learning repository (Blake, Keogh, & Merz 1998) by Frey and Slate (1991) consists of 20000 feature vectors belonging to 26 classes that represent capital letters of Latin alphabet. Since our current implementation works only with binary classification, we converted the database to such by changing all letters from ‘a’ to ‘m’

to 0 and all the letters from ‘n’ to ‘z’ to 1. The learning rate of the various selective sampling methods is shown in Figure 7. The lookahead selective sampling algorithm outperforms other selective sampling methods in this particularly hard domain, where every class consists of many different (associated with the different letters).

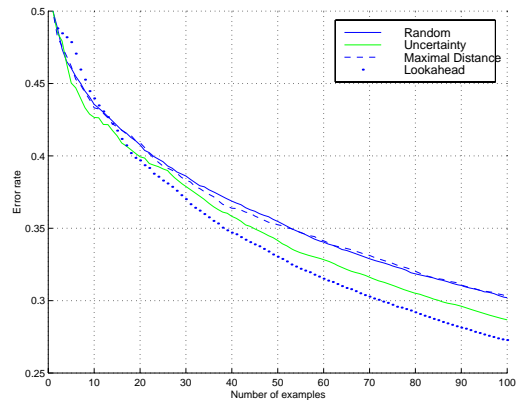


Figure 7: Learning rate graphs for various selective sampling methods applied to the letters dataset.

Discussion

Nearest neighbor classifiers are often used when little or no information is available about the instance space structure. There, the loose, minimalistic specification of the instance space labeling structure, which is implied by the distance based random field model, seems to be adequate. We also observe that large changes in the covariance function had no significant effect on the classification performance.

The experiments show that lookahead sampling method performs better or comparatively to other selective sampling algorithms on both artificial and real domains. It is especially strong when the instance space

See the experimental part for an evaluation of some covariance function and for their use in estimating the parameters. With this method, every sampled point influences the estimated probability. In practice, such long range influence is non-intuitive and is also computationally expensive. Therefore, in practice, we neglect the influence of all except the two closest neighbors. This choice gives a higher probability to the nearest neighbor class and is therefore consistent with 1-*NN* classification. One deficiency of this estimation process is that the estimated probabilities are not guaranteed to lie in the required $[0, 1]$ range. When such overflows indeed happen (very rarely), we correct them by clipping the estimate. This deficiency is corrected in more complex estimation procedures, described in the full version (Lindenbaum, Markovich, & Rusakov 1999). (The framework we use is similar to *Bayesian Classification via Gaussian Process Modeling* (MacKay 1998; Williams & Barber 1998)

Experimental Evaluation

We have implemented our random-field based lookahead algorithm and tested it on several problems, comparing its performance with several other selective sampling methods.

Experimental Methodology

The algorithm described in the previous sections allows us to heuristically choose the covariance function, $\gamma(d)$. In the experiments described here, every class contained a nearly equal number of examples and therefore we assume that the a priori class probabilities are equal. This implies that $\gamma(0) = 0.25$. We choose an exponentially decreasing covariance function (common in image processing) $\gamma(d) = 0.25e^{-d/\sigma}$. We tested the effect of a range of σ values on the performance of the algorithm and found that changing σ had almost no effect (these results are included in the full version (Lindenbaum, Markovich, & Rusakov 1999)

The lookahead algorithm was compared with the following three selective sampling algorithms, which represent the most common choices (see introduction):

- *Random sampling*: The algorithm randomly selects the next example. While this method looks unsophisticated, it has the advantage of yielding a uniform exploration of the instance space. This method actually corresponds to a *passive* learning model.
- *Uncertainty sampling*: The method selects the example which the current classifier is most uncertain about. The uncertainty for each example depends on the ratio between the distances to the closest labeled neighbors of different classes. This method tends to sample on the existing border, and while for some decision boundaries that may be beneficial, for others it may be a source for serious failure (as will be shown in the following subsections).
- *Maximal distance*: An adaptation of the method described by Hasenjager and Ritter (1998). This

method selects the example from the set of all unlabeled points that have different labels among their three nearest classified neighbors. The example selected is the one which is most distant from its closest labeled neighbor.

The basic measurement used for the experiments is the expected error rate. For each selective sampling method and for each dataset the following procedure was applied:

1. 1000 examples from the dataset were drawn randomly - this is a set used for selective sampling and learning X , the rest 19000 examples (all datasets included 20000 examples) were used *only* for the evaluation of error rates of the resulting classifiers.
2. The selective sampling algorithm was applied to the chosen set, X . After selection of each example, the error rate of the current hypothesis, h (which is the nearest neighbor classifier), was calculated using the test set of 19000 examples put aside.
3. Steps 1, 2 were performed 100 times and the average error rate was calculated.

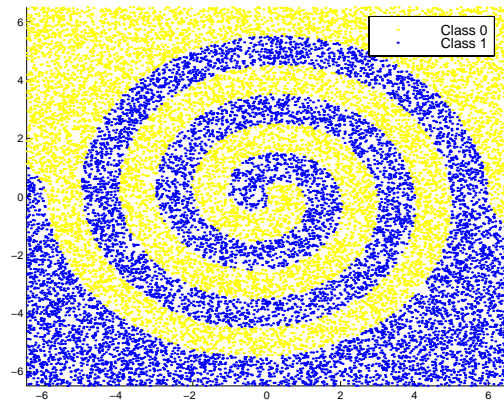


Figure 3: The feature space of the “two spirals” data.

The ‘Two Spirals’ Problem

The *two spirals* problem was studied by a number of researchers (Lang & Witbrock 1988; Hasenjager & Ritter 1998). This is an artificial problem where the task is to distinguish between two spirals of uniform density in XY -plane, as shown in Figure 3. (The code for generating these spirals was based on (Lang & Witbrock 1988)) The Bayes error of such classification is zero since the classes are perfectly separable. The learning rate of the various selective sampling methods is shown in Figure 4. All three non-random methods demonstrated comparable performance, better than random sampling. In the next experiment we will show that other methods lack one of the basic properties required from selective sampling algorithms - exploration - and fail in the datasets consisting of separated regions of the same classification.

distribution of consistent target functions. First, consider a specific target f . Let $I_{f,h}$ be a binary indicator function, where $I_{f,h}(x) = 1$ iff $f(x) = h(x)$, and let $\alpha_f(h)$ denote the accuracy of hypothesis h relative to f : $\alpha_f(h) = \int_{x \in \mathbb{R}^d} I_{f,h}(x) \mathbf{p}(x) dx$. Recall that $\mathbf{p}(x)$ is the probability density function specifying the instance distribution over \mathbb{R}^d . Let $\mathcal{A}_L(D)$ denote the expected accuracy of a hypothesis produced by learning algorithm L :

$$\mathcal{A}_L(D) = E_{f|D}[\alpha_f(h = L(D))] = E_{f|D} \left[\int_{x \in \mathbb{R}^d} I_{f,h}(x) \mathbf{p}(x) dx \right] = \int_{x \in \mathbb{R}^d} P(f(x) = h(x)|D) \mathbf{p}(x) dx \quad (1)$$

where $P(f(x) = h(x)|D)$ is the probability that a random target function f consistent with D will be equal to h in the point x , i.e. $P(f(x) = h(x)|D) = E_{f|D}[f(x) = h(x)]$.

Note that $P(f(x) = h(x)|D)$ is the probability that a particular point x gets the correct classification. Therefore, for every given hypothesis h , estimating the class probabilities $P(f(x) = 0|D), P(f(x) = 1|D)$, gives also the accuracy estimate (from Equation 1):

$$\mathcal{A}_L(D) \approx \sum_{x \in X} P(f(x) = h(x)|D) / |X|. \quad (2)$$

(The number of examples in X is assumed to be finite). Thus the problem of evaluating the utility measure as the classifier accuracy is translated into the problem of estimating the class probabilities. Assuming that the probability computation model is correct, the optimal selective sampling strategy is one that uses $U_L^*(D) \triangleq \mathcal{A}_L(D)$ as the utility function.

Random Field Model for Feature Space Classification

Feature vectors from the same class tend to cluster in the feature space (though sometimes the clusters are quite complex). Therefore close feature vectors share the same label more often than not. This intuitive observation, which is the rationale for the nearest neighbor classification approach, is used here to estimate the classes of unlabeled feature points and their uncertainties.

Mathematically, this observation is described by assuming that the label of every point is a random variable, and that these random variables are mutually dependent. Such dependencies are usually described (in a higher than 1-dimensional space) by *random field models*. In the probabilistic setting, estimating the classification of unlabeled vectors and their uncertainties is equivalent to calculating the conditional class probabilities from the labeled data, relying on the random field model. In the full version of the paper (Lindenbaum, Markovich, & Rusakov 1999), we consider several options for such estimates. This shorter version focuses on one particular model.

Thus, we assume that the classification of an instance space is a *sample function* of a binary valued *homogeneous isotropic random field* (Wong & Hajek 1985) characterized by a *covariance* function decreasing with a distance. (see (Eldar *et al.* 1997) where a similar method was used for progressive image sampling.) That is: let x_0, x_1 be points in X and let θ_0, θ_1 be their classifications, i.e. random variables that can have values of 0 or 1. The homogeneity and isotropy properties imply that the expected values of θ_0 and θ_1 are equal, i.e. $E[\theta_0] = E[\theta_1] = \bar{\theta}$, and the covariance between θ_0 and θ_1 is specified only by the distance between x_0 and x_1 :

$$C[\theta_0, \theta_1] = E[(\theta_0 - \bar{\theta})(\theta_1 - \bar{\theta})] \triangleq \gamma(d(x_0, x_1)) \quad (3)$$

where $\gamma: \mathbb{R}^+ \rightarrow (-1, 1)$ is a covariance function with $\gamma(0) = Var[\theta] = E[(\theta - \bar{\theta})^2] = P_0 P_1$, where $P_0, P_1 = 1 - P_0$ are the a priori class probabilities. Usually we will assume that γ is decreasing with the distance and that $\lim_{r \rightarrow \infty} \gamma(r) = 0$. Note that the random field model specifies (indirectly) a distribution of target functions.

In estimation, one tries to find the value of some unobserved random variable, from observed values of other, related, random variables, and prior knowledge about their joint statistics.

The class probabilities associated with some feature vector are uniquely specified by the conditional mean of its associated random variable (r.v.) This conditional mean is also the best estimator for the r.v. value in the least squares sense (Papoulis 1991). Therefore, the widely available methods for *mean square error* (MSE) estimation can be used for estimating the class probabilities.

We choose a linear estimator, for which a closed form solution, described below, is available. Let θ be the binary r.v. associated with some unlabeled feature vector, x_0 , and let $\theta_1, \dots, \theta_n$ be the known labels r.v. associated with the feature vectors, x_1, \dots, x_n , that were already sampled. Now let

$$\hat{\theta} = \alpha_0 + \sum_{i=1}^n \alpha_i \theta_i \quad (4)$$

be the estimate of the unknown label. The estimate uses the known labels and relies on unknown coefficients which should be set so that the MSE, $\epsilon_{mse} = E[(\hat{\theta} - \theta_0)^2]$ is minimized.

The optimal linear approximation in the MS sense (Papoulis 1991) is described by:

$$\hat{\theta} = E[\theta_0] + \bar{\mathbf{a}} \cdot (\vec{\theta} - E[\vec{\theta}])^t \quad (5)$$

where $\bar{\mathbf{a}}$ is an n -dimensional vector specified by the covariance values:

$$\begin{aligned} \bar{\mathbf{a}} &= \mathbf{R}^{-1} \cdot \bar{\mathbf{r}}, \\ R_{ij} &= E[(\theta_i - E[\theta])(\theta_j - E[\theta])], \\ r_i &= E[(\theta_0 - E[\theta])(\theta_j - E[\theta])]. \end{aligned} \quad (6)$$

(\mathbf{R} is an $n \times n$ matrix, and $\bar{\mathbf{a}}, \bar{\mathbf{r}}$ are n -dimensional vectors). The values of \mathbf{R} and $\bar{\mathbf{r}}$ are specified by the random field model:

$$\begin{aligned} R_{ij} &= \gamma(d(x_i, x_j)), \\ r_i &= \gamma(d(x_0, x_i)). \end{aligned} \quad (7)$$

a *teacher* (also called an oracle or an expert) which labels instances by 0 or 1, $f : \mathcal{X} \rightarrow \{0, 1\}$. A *learning algorithm* takes a set of *classified examples*, $\{(x_1, f(x_1)), \dots, (x_n, f(x_n))\}$, and returns a *hypothesis* h , $h : \mathcal{X} \rightarrow \{0, 1\}$. Throughout this paper we assume that $\mathcal{X} = \mathbb{R}^d$.

Let X be an *instance space* - a set of objects drawn randomly from \mathcal{X} according to distribution \mathbf{p} . Let $D \subset X$ be a finite set of classified examples. A selective sampling algorithm S_L with respect to learning algorithm L takes X and D , and returns an unclassified element of X . An *active learning process* can be described as follows:

1. $D \leftarrow \emptyset$
2. $h \leftarrow L(\emptyset)$
3. While stop-criterion is not satisfied do:
 - (a) Apply S_L and get the next example, $x \leftarrow S_L(X, D)$.
 - (b) Ask the teacher to label x , $\omega \leftarrow f(x)$
 - (c) Update the labeled examples set, $D \leftarrow D \cup \{(x, \omega)\}$
 - (d) Update the classifier, $h \leftarrow L(D)$
4. Return classifier h

The stop criterion may be a limit M on the number of examples that the teacher is willing to classify or a lower bound on the classifier accuracy. We will assume here the first case. The goal of the selective sampling algorithm is to produce a sequence of length M which leads to a best classifier according to some given criterion.

Lookahead Algorithms for Selective Sampling

Knowing that we are allowed to ask for exactly M labels allows, in principle, to consider all object sequences of length M . Not knowing the labeling of these objects, however, prevents us from evaluating the resulting classifiers directly. One way to overcome this difficulty is to consider the selective sampling process as an interaction between the learner and the teacher. At each stage the learner must select an object from the set of unclassified instances and the teacher assigns one of the possible labels to the selected object. This interaction can be represented by a “game tree” of $2M$ levels such as the one illustrated Figure 1.

We can use such a tree representation to develop a lookahead algorithm for selective sampling. Let $U_L(D)$ be a *utility evaluation function* that is capable of appraising a set D as examples for a learning algorithm L . Let us define a k -deep lookahead algorithm for selective sampling with respect to learning algorithm L as illustrated on Figure 2.

Note that this algorithm is a specific case of a decision theoretic agent, and that, while it is specified for maximizing the expected utility, one can be, for exam-

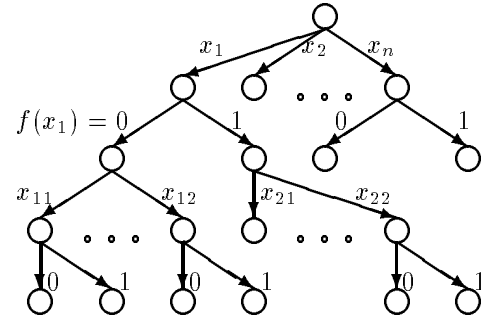


Figure 1: Selective sampling as a game.

$\mathbf{S}_L^k(\mathbf{X}, \mathbf{D})$:
 Select $x' \in X$ with maximal expected utility:

$$x' = \arg \max_{x \in X} E_\omega[U_L^*(X, D \cup \{(x, \omega)\}, k - 1)]$$

where $U_L^*(X, D, k)$ is a recursive utility propagation function:

$$U_L^*(X, D, k) = \begin{cases} U_L(D) & k = 0 \\ \max_x E_\omega[U_L^*(X, D', k - 1)] & k > 0 \end{cases}$$

where $D' = D \cup \{(x, \omega)\}$ and the expected value $E_\omega[\cdot]$ is taken according to conditional probabilities for classification of x given D , $P(f(x) = \omega | D)$.

Figure 2: Lookahead algorithm for selective sampling.

ple, pessimistic and consider a *minimax* approach. In our implementation we use a simplified one-step lookahead algorithm:

$\mathbf{S}_L^*(\mathbf{X}, \mathbf{D})$:
 Select $x \in X$ with maximal expected utility, $E_{\omega \in \{0, 1\}}[U_L(D \cup \{(x, \omega)\})]$, which is equal to:

$$P(f(x) = 0 | D) \cdot U_L(D \cup \{(x, 0)\}) + P(f(x) = 1 | D) \cdot U_L(D \cup \{(x, 1)\})$$

The actual use of the lookahead example selection scheme relies on two choices:

- The utility function $U_L(D)$.
- The method for estimating $P(f(x) = 0 | D)$ (and $P(f(x) = 1 | D)$).

Two particular choices are considered in the next sections.

The Classifier Accuracy Utility Function

Taking a Bayesian approach, we specify the utility of the classifier as its expected accuracy relative to the

Selective Sampling for Nearest Neighbor Classifiers

Michael Lindenbaum
mic@cs.technion.ac.il

Shaul Markovich
shaulm@cs.technion.ac.il

Dmitry Rusakov
rusakov@cs.technion.ac.il

Computer Science Department,
Technion - Israel Institute of Technology,
32000, Haifa, Israel

Abstract

In the *passive*, traditional, approach to learning, the information available to the learner is a set of classified examples, which are randomly drawn from the instance space. In many applications, however, the initial classification of the training set is a costly process, and an intelligently selection of training examples from unlabeled data is done by an *active* learner.

This paper proposes a lookahead algorithm for example selection and addresses the problem of active learning in the context of nearest neighbor classifiers. The proposed approach relies on using a *random field* model for the example labeling, which implies a dynamic change of the label estimates during the sampling process.

The proposed selective sampling algorithm was evaluated empirically on artificial and real data sets. The experiments show that the proposed method outperforms other methods in most cases.

Introduction

In many real-world domains it is expensive to label a large number of examples for training, and the problem of reducing training set size, while maintaining the quality of the resulting classifier, arises. A possible solution to this problem is to give the learning algorithm some control over the inputs on which it trains.

This paradigm is called *active learning*, and is roughly divided into two major subfields: learning with *membership queries* and *selective sampling*. In learning with membership queries (Angluin 1988) the learner is allowed to construct artificial examples, while selective sampling deals with selection of informative examples from a large set of unclassified data.

Selective sampling methods have been developed for various classification learning algorithms: for neural networks (Davis & Hwang 1992; Cohn, Atlas, & Lander 1994), for the *C4.5* rule-induction algorithm (Lewis & Catlett 1994) and for HMM (Dagan & Engelson 1995).

The goal of the research described in this paper is to develop a selective sampling methodology for nearest neighbor classification learning algorithms. The

nearest neighbor (Cover & Hart 1967; Aha, Kibler, & Albert 1991) algorithm is a non-parametric classification method, useful especially when little information is known about the structure of the distribution, implying that parametric classifiers are harder to construct. The problem of active learning for nearest neighbor classifiers was considered by Hasenjager and Ritter (1998). They proposed querying in points which are the farthest from previously sampled examples, i.e. in the vertices of Voronoi diagram of the points labeled so far. This method, however, falls under the membership queries paradigm and is not suitable for selective sampling.

Most existing selective sampling algorithms focus on choosing examples from regions of uncertainty. One approach to define uncertainty is to specify a committee (Seung, Opper, & Sompolinsky 1992) or an ensemble (Krogh & Vedelsby 1994) of hypotheses consistent with the sampled data and then to choose an example on which the committee members most disagree. *Query By Committee* is an active research topic, and strong theoretical results (Freund *et al.* 1997) along with practical justifications (Dagan & Engelson 1995; Hasenjager & Ritter 1996; RayChaudhuri & Hamey 1995) were achieved. It is not clear, however, how to apply this method to nearest-neighbor classification.

This paper introduces a lookahead approach to selective sampling that is suitable for nearest neighbor classification. We start by formalizing the problem of selective sampling and continue with a lookahead based framework which chooses the next example (or sequence of examples) in order to maximize the expected utility (*goodness*) of the resulting classifier. The major components needed to apply this framework are an utility function for appraising classifiers and a posteriori class probability estimates for points in the instance space. We propose a *random field* model for the feature space classification structure. This model serves as the basis for a class probability estimation. The merit of our approach is empirically demonstrated on artificial and real problems.

The Selective Sampling Process

We consider here the following selective sampling paradigm. Let \mathcal{X} be a set of objects. Let f be