Selective Sampling Using Random Field Modelling

Michael Lindenbaum mic@cs Shaul Markovitch shaulm@cs Dmitry Rusakov rusakov@cs

Abstract

Most existing inductive learning algorithms assume the availability of a training set of labeled examples. In many domains, however, labeling the examples is a costly process that requires either intensive computation or manual labor. In such cases, it may be beneficial for the learner to be *active* by intelligent selection of examples for labeling with the goal of reducing the labeling cost.

In this paper we propose a lookahead algorithm for selective sampling of examples for nearestneighbors classifiers. The algorithm attempts to find the example with the highest utility considering its effect on the resulting classifier. Computing the expected utility of an example requires estimating the probability of the possible labels. We propose to use the *random field* model for this estimation.

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

1 Introduction

Most existing inductive learning algorithms assume the availability of a training set of labeled examples. In many domains, however, labeling the examples is a costly process that requires either intensive computation or manual labor. Consider, for example, training a classifier for a character recognition problem. In this case, the character images are easily available, while their classification is a costly process requiring a human operator. Another example may be a game-playing program that infers a resource allocation strategy by inducing the class of positions that requires extra computation (Markovitch & Sella, 1996). Labeling examples for this purpose requires a costly computation of deep alpha-beta search. Yet another example is a robot that learns about objects in an unknown environment by performing experiments on them (Scott & Markovitch, 1993).

In such domains it is desirable to reduce the number of training examples while maintaining the quality of the resulting classifier. A possible solution to this problem is to provide the learning algorithm with 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 the selection of informative examples from a large set of unlabeled data (Lewis & Catlett, 1994; Freund, Seung, Shamir & Tishbi, 1997; Dagan & Engelson, 1995). Note that the membership query paradigm may pose a practical problem by requesting a label for a nonsense example, e.g. asking to appraise an illegal chess-board position, or asking to recognize an arbitrary image as a character.

One recent approach to selective sampling in a general setting, which is accompanied by strong theoretical justification, is *Query by Committee*. There, a committee of hypotheses consistent with the labeled data is specified and an unlabeled example on which the committee members most disagree is chosen (Seung, Opper & Sompolinsky, 1992; Freund, Seung, Shamir & Tishbi, 1997; Hasenjager & Ritter, 1996; RayChaudhuri & Hamey, 1995). Techniques from the field of optimal experiment design

(Fedorov, 1972) have been applied to active learning of real-valued functions (MacKay, 1992; Cohn, Ghahramani & Jordan, 1995). Various selective sampling methods have been developed for specific 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 Hidden Markov Models (Dagan & Engelson, 1995).

The goal of this paper is to develop a selective sampling methodology for nearest-neighbor classification learning algorithms. The nearest-neighbor algorithm (Cover & Hart, 1967; Aha, Kibler & Albert, 1991) is a non-parametric classification method, useful especially when little information is available about the structure of the distribution, implying that parametric classifiers are harder to construct. The problem of active learning for nearest-neighbor classifiers in the context of membership queries paradigm was considered by Hasenjager and Ritter (1998). They proposed querying in points which are the farthest (in the instance space) from previously sampled examples, that is, in the vertices of the Voronoi diagram of the points labeled so far.

This paper introduces a lookahead approach to selective sampling which is suitable for nearestneighbor classification. The lookahead-based method for selective sampling chooses the next example (or sequence of examples) in order to maximize the expected utility of the resulting classifier. The major components of this framework are a utility function for appraising classifiers and a posteriori class probability estimates for unlabeled points in the instance space. We derive a Bayesian utility function and propose a *random field* model for the feature space classification structure, which serves as a basis for class probability estimation.

The merit of our approach is empirically demonstrated on real and artificial problems. The algorithm was compared with a number of known selective sampling methods adopted for use with nearest-neighbor classifier. The lookahead selective sampling method outperformed the other methods and exhibited a much more stable performance on the wide range of real and artificial problems.

This article is organized in the following way. We start by formalizing the problem of selective sampling and by presenting intuitive considerations for selective sampling methods (Section 2). In Section 3 we describe the lookahead-based framework and propose methods for estimating the utility of classifiers and for estimating the class probabilities. In Section 4 we develop a random field model for the feature space classification structure, and in Section 5 we show how this methodology can be implemented. Section 6 describes the experimental evaluation of our algorithm and Section 7 concludes.

2 Selective Sampling for Nearest Neighbor Classifiers

In order to address the problem of selective sampling for the nearest neighbor classifier, we must first understand and formalize what do we mean under the concept of selective sampling. In particular, what information a selective sampling algorithm can and should use, and what is the goal of the selective sampling process. These questions are addressed in the follow subsections.

2.1 The selective sampling process

We consider here the following selective sampling paradigm. Let \mathcal{X} be a set of objects described by a finite collection of attributes (*features*). Let $f : \mathcal{X} \to \{0, 1\}$ be a *teacher* (also called an expert) which labels instances by 0 or 1. A *learning algorithm* takes a set of *labeled examples*, $\{\langle x_1, f(x_1) \rangle, \ldots, \langle x_n, f(x_n) \rangle\}$, and returns a hypothesis $h : \mathcal{X} \to \{0, 1\}$. 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 \mathfrak{p} . Let $D = \{\langle x_i, f(x_i) \rangle : x_i \in X, i = 1, ..., n\}$ be a set of labeled examples from X. A selective sampling algorithm S_L , with respect to learning algorithm L, takes X and D, and returns an unlabeled element of X. The architecture of an active learning system that uses selective sampling for obtaining training examples is illustrated in Figure 1.



Figure 1. Active Learning System

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, $D \leftarrow D \bigcup \{ \langle x, \omega \rangle \}$
- (d) Update the classifier, $h \leftarrow L(D)$

4. Return classifier h

Figure 2. Learning with Selective Sampling

The process of learning with selective sampling can be described as an iterative procedure where at each iteration the selective sampling procedure is called to obtain an unlabeled example and the teacher is called to label that example. The labeled example is added to the set of currently available labeled examples and the updated set is given to the learning procedure which induce a new classifier. This sequence repeats until the stop criterion is satisfied. The pseudo-code for this algorithm is described in Figure 2. The stop criterion may be a resource bound M on the number of examples that the teacher is willing to label, or a lower bound on the desired class accuracy. Here we will assume 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.

2.2 Nearest neighbor classification

In this paper we will be particularly interested in selective sampling for the nearest-neighbor classifier (Cover & Hart, 1967). The nearest-neighbor classifier accumulates the labeled examples received as input. An unlabeled instance is then classified according to the label of its nearest labeled neighbor. Variations of this scheme include k-nearest neighbor classifiers (Duda & Hart, 1973), which use the vote of the k nearest labeled neighbors and selective classifiers that store and utilize the labeled examples selectively (Aha, Kibler & Albert, 1991).

Formally speaking, given a set of labeled examples pairs $\{\langle x_1, \omega_1 \rangle, \ldots, \langle x_n, \omega_n \rangle\}$, where the x_i 's are objects from a set \mathcal{X} with the metric d, and the ω_i 's take values from the set $\{0, 1\}$, the nearest-



Figure 3: Voronoi diagram and nearest-neighbor classification: (a) Voronoi diagram for a set of 10 points. Each query point for nearest-neighbor classifier will be classified according to the label of the center point of the Voronoi cell it belongs to. (b) The decision boundary of the nearest-neighbor classifier based on the labeled points of (a).

neighbor rule decides that $x \in \mathcal{X}$ belongs to the category ω_j of its nearest neighbor x_j , where $j = \arg\min_{i=1,\ldots,n} d(x_i, x)$.

The concept of nearest-neighbor classification is closely related to the concept of Voronoi diagram for a set of points. Voronoi diagram of n points $X = \{x_1, \ldots, x_n\}$ from a metric space \mathcal{X} is a partition of \mathcal{X} into regions $\{V(x_1), \ldots, V(x_n)\}$, such that $V(x_i) = \{x \in \mathcal{X} : \forall x_j \in X, d(x, x_i) \leq d(x, x_j)\}$. In this way the Voronoi diagram actually defines the space classification for the nearest-neighbor rule based on the same points, see Figure 3.

2.3 Some intuitive considerations

What should be considered to be a good example? In this subsection we discuss some of the intuitive considerations for selective sampling. The underlying principle is that the uncertainty of classification should be reduced. One possible strategy for achieving such a reduction is to assign classification uncertainty to each unlabeled example and to select the unlabeled instance associated with the highest uncertainty. For the nearest-neighbor classifier the most uncertain points will lie near the classification boundary. Such points have two nearest neighbors with similar distances but conflicting labeling. This idea can be used to develop a selective sampling algorithm for nearest-neighbor classifier. Looking carefully at the sampling style of this method, however, we can discover a number of intrinsic problems.

Consider various point configurations shown in Figure 4. These configurations are much simpler than the typical ones that may arise in practice, but they provide test cases for analyzing the behavior of selective sampling algorithms.

The boundary sampling will obviously prefer to sample point (a) over point (b) in configuration (1). This decision fits our intuition, since point (b) seems to be is less important because of its proximity to the labeled point. Configuration (2), however, shows a weakness of the naive uncertainty sampling. In this configuration, the boundary method will prefer sampling in the border (point (c)). If point (c) will



Figure 4: Various point configurations that may arise in the instance space. Unlabeled points are marked by circles. (1) Single point near the border. (2) Group of points near the border. (3) Compact group of points far from labeled examples. (4) The figure of eight, very far from labeled examples.

be labeled as '-', the uncertainty associated with the classification of cluster (d) will remain high. On the other hand, sampling in any point in cluster (d) will yield classification with high certainty for all its points. Thus, in configuration (2), sampling in cluster (d) is preferable over the sampling point (c).

These two examples imply that selective sampling algorithms should consider not only the uncertainty of the candidate sample point, but also the effect of its classification on the remaining unlabeled points. Thus, sampling in dense regions may be preferred over sampling in an isolated point. Even when a compact group of unlabeled points is surrounded by the instances of the same classification, as illustrated in configuration (3), in make sense to sample in it, because the resulting label influences many points.

Where should we sample if the instance space contains both configurations (2) and (3)? Sampling in either cluster will be especially beneficial if the true label of the cluster is '-'. In this case, the sample in cluster (d) is preferable, since intuitively cluster (e), being surrounded by points of class '+', is less likely to be of different labeling.

What if we can sample two or more instances in a row? Although we will select these points one by one (according to the framework described in Section 2.1) the very knowledge of the fact that we can sample more than one point can change our priorities in example selection. For example, in configuration (4), the two clusters may be of different classes. If we are allowed to sample only one instance, then selecting a point between the two clusters may be a best strategy. On the other hand, if we know that we will be able to ask for the label of another instance, we may be better of sampling in the center of one cluster (and sampling in the center of the other cluster afterwards).

The above examples show that one must consider not only the uncertainty of the particular point in the instance space, but mostly the effect that the labeling of this point may have on its neighborhood. The *lookahead* selective sampling algorithm described in next section takes these considerations into account.

3 Lookahead Algorithms for Selective Sampling

In this section we develop a lookahead framework for selective sampling algorithms, that chooses the next example (or sequence of examples) in order to maximize the expected utility of the resulting classifier. The framework requires a method for estimating class probabilities of unlabeled instances and a utility function for appraising classifiers. We derive an optimal utility function in the Bayesian sense, and describe a number of possibilities for class probability estimation methods.



Figure 5: Lookahead, part of the tree of depth 2: (a) - without considering the teacher response, (b) - considering the teacher response.

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

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

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

$$U_L^*(X, D, D', k) = \begin{cases} U_L(D', D) & k = 0\\ \max_x E_\omega[U_L^*(X, D, 0, 0] \\ D' \cup \{\langle x, \omega \rangle\}, k - 1\}] & \text{otherwise} \end{cases}$$

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

Figure 6. k-deep lookahead algorithm.

3.1 The lookahead framework for selective sampling

Using an intuition developed in the previous section, we may introduce the *lookahead* algorithm for selective sampling, which considers sampling sequences of length k and selects an example that leads to the best sequence (as illustrated in Figure 5a). The merit of the sequence is determined by estimating the utility of selected points as a training set for the classifier. For example, one may prefer sequences that uniformly sample the instance space (according to its distribution).

One problem with this approach is that it does not take into account the possible responses of the teacher. An alternative approach views 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" such as the one shown in Figure 5b.

We use this tree representation in order to develop a lookahead algorithm for selective sampling. Let $U_L(D', D \subseteq D')$ be a *utility function* that estimates the merit of adding labeled instances $D' \setminus D$ to the set D as training examples for learning algorithm L. Let $P(f(x) = \omega | D)$ denote the conditional class probabilities of x for a given labeled set D. The k-deep lookahead algorithm for selective sampling with respect to learning algorithm L selects the example that leads to the learning sequence with the highest expected utility. This algorithm is presented in Figure 6.

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 example, pessimistic and consider a *minimax* approach.

 $\mathbf{S}_{\mathbf{L}}^{1}(\mathbf{X}, \mathbf{D})$: Select $x \in X$ with maximal expected utility, $E_{\omega \in \{0,1\}}[U_{L}(D \cup \{\langle x, \omega \rangle\}, D)]$, which is equal to:

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

Figure 7. One-step lookahead algorithm.

In our implementation we use a one-step lookahead algorithm which assumes $\omega \in \{0, 1\}$. This algorithm is illustrated in Figure 7.

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

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

These choices are considered in the next sections.

3.2 Accuracy-based utility functions

Taking a Bayesian approach, we specify the utility of the classifier as its expected accuracy relative to the distribution of consistent target functions. First, consider a specific target f and a hypothesis h. Let $I_{f,h}$ be a binary indicator function, where $I_{f,h}(x) = 1$ if and only if f(x) = h(x), and let $\alpha_f(h) = \int_{x \in \mathbb{R}^d} I_{f,h}(x) \mathfrak{p}(x) dx$ denote the accuracy of hypothesis h relative to f. Recall that $\mathfrak{p}(x)$ is the probability density function specifying the instance distribution over \mathbb{R}^d (Section 2.1). Let $\mathcal{A}_L(D)$ denote the expected accuracy of a hypothesis produced by learning algorithm L on data D:

$$\mathcal{A}_{L}(D) = E_{f|D}[\alpha_{f}(h = L(D))]$$

= $E_{f|D}[\int_{x \in \mathbb{R}^{d}} I_{f,h}(x)\mathfrak{p}(x)dx] = \int_{x \in \mathbb{R}^{d}} P(f(x) = h(x)|D)\mathfrak{p}(x)dx$ (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) and P(f(x) = 1|D) gives also the accuracy estimate (from Equation 1):

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

We assume that X is large but finite, otherwise, if X is infinite, we can draw a sufficiently large and finite set $X' \subset X$ and work with it instead of X. Equation 2 translates the problem of evaluating the utility measure as a classifier accuracy into the problem of estimating the class probabilities. Assuming that the probability calculation model is correct, the Bayesian selective sampling strategy is one that uses $U_L^{acc}(D', D) \triangleq \mathcal{A}_L(D')$ as the utility function.

If we assume that the learning process is "monotonic", i.e. that additional examples only increase the accuracy of the resulting classifier, than we can devise an alternative utility function based on this assumption. The proposed function evaluates the increase in accuracy between the classifier based on the current data (D) and the classifier based on the data together with the additional examples (D'). This accuracy gain is evaluated from the point of view of the resulting classifier (based on D').

Assume we have labeled data D and we wish to evaluate the merit of sampling points $D' \setminus D$. Let h be a nearest-neighbor classifier based on D and let h' be a nearest-neighbor classifier based on D'. Given

a target function f, the increase in accuracy is:

$$\begin{aligned} f(h',h) &\triangleq \alpha_f(h') - \alpha_f(h) \\ &= \int_{x \in \mathbb{R}^d} I_{f,h'}(x) \mathfrak{p}(x) dx - \int_{x \in \mathbb{R}^d} I_{f,h}(x) \mathfrak{p}(x) dx \\ &= \int_{x \in \mathbb{R}^d, h(x) \neq h'(x)} (I_{h',f}(x) - I_{h,f}(x)) \mathfrak{p}(x) dx \end{aligned}$$
(3)

We define the utility function to be the expected value of $\epsilon_f(h', h)$ given D':

$$U_{L}^{ch}(D',D) = E_{f|D'}[\epsilon_{f}(h' = L(D'), h = L(D))]$$

= $\int_{x \in \mathbb{R}^{d}, h(x) \neq h'(x)} (P(f(x) = h'(x)|D') - P(f(x) = h(x)|D')) \mathfrak{p}(x)dx$ (4)
= $\int_{x \in \mathbb{R}^{d}, h(x) \neq h'(x)} (2P(f(x) = h'(x)|D') - 1) \mathfrak{p}(x)dx$

Since X was drawn randomly from \mathbb{R}^d according to \mathfrak{p} , we can approximate U_L^{ch} by:

$$U_L^{ch}(D',D) \approx \sum_{x \in X, h(x) \neq h'(x)} (2P(f(x) = h'(x)|D') - 1)/|X|$$
(5)

The value of $U_L^{ch}(D', D)$ is positive, because h' should be a Bayesian classification under the specified probability model.

The presented utility function tends to sample points that have the potential of radically changing the current hypothesis. The first proposed utility function, U_L^{acc} , is more conservative in this sense. This difference may lead to faster learning when using U_L^{ch} in domains that have the "monotonicity" property.

In this section we have solved the problem of estimating the utility function, by moving the focus to determining the correct class probabilities from the labeled data. Once these probabilities are estimated, one should have no problem to construct the utility function for a lookahead selective sampler.

3.3 Probability estimation methods

The utility propagation method used in the lookahead selective sampling algorithm needs the same conditional probabilities as the utility function. Denote by $P_0(x)$, $P_1(x)$ the approximation of the probability that x is labeled 0 or 1 for a given data D, i.e. $P_0(x) \approx P(f(x) = 0|D)$ and $P_1(x) \approx P(f(x) = 1|D)$.

Some possible options for the estimation of conditional class probabilities are described below:

• The simplest approach to estimate these conditional probabilities is to assume them equal to a priori class probabilities, P_0 and P_1 . These a priori probabilities can be estimated from labeled data by averaging the number of points of each class:

$$P_{0} = \frac{|\{\langle x, \omega \rangle \in D : \omega = 0\}|}{|D|}, \ P_{1} = 1 - P_{0}.$$
(6)

This approach assumes constant class probabilities for all regions in the feature space. This assumption is too rough and does not lead to an effective example selective method.

• A common heuristic is to specify class probabilities by a distance ratio to the nearest labeled examples of different classes:

$$P_{0}(x) = \frac{d_{1}}{d_{0}+d_{1}}, P_{1}(x) = 1 - P_{0}(x),$$

$$d_{0} = \min_{\{s,0\} \in D} d(x,s),$$

$$d_{1} = \min_{\{s,1\} \in D} d(x,s).$$
(7)

• A more rigorous approach, which converges (in a limit) to the true probability (Duda & Hart, 1973), is to set the probabilities by voting, or weighted voting, between the k nearest neighbors:

$$P_{0}(x) = \frac{\sum_{(s,0) \in D_{x,k}} w(d(x,s))}{\sum_{(s,\omega) \in D_{x,k}} w(d(x,s))}, P_{1}(x) = 1 - P_{0}(x),$$

$$D_{x,k} \triangleq k \text{ points from } D \text{ closest to } x.$$
(8)

where w(d) is a monotonically non-increasing weight (influence) function. For ordinary vote procedure, $w(d) \equiv 1$.

- Technion Computer Science Department Technical Report CIS9906 1999
- Estimation of class probabilities can be also done by using a committee of consistent hypotheses (This method is inspired by *Query by Committee* paradigm (Seung, Opper & Sompolinsky, 1992; Freund, Seung, Shamir & Tishbi, 1997)). The first step is to choose some reasonable class of possible target functions, For example, For an be the class of nearest-neighbor classifiers based on *m* labeled points, *NN_m*. Then the probabilities can be estimated by:

$$P_0(x) = \frac{|\{h : h \in H, h(x) = 0\}|}{|H|}, P_1(x) = 1 - P_0(x)$$
(9)

where $H = \{h \in \mathfrak{F} : \forall \langle x, \omega \rangle \in D, h(x) = \omega\}$. The exact computation of H may be intractable, and a random finite subset of consistent hypotheses $H' \subset H$ may be used instead.

The complexity of \mathfrak{F} may be adapted to the expected number of allowed examples M (see Section 2.1), e.g. for the class of nearest-neighbor classifier, NN_m , m may increase with M.

A theoretical issue is whether such conditional probability estimates are consistent with some target function family \mathfrak{F} , i.e. $P_0(x) \approx P(f(x) = 0|D)$ for any finite $D \subset \mathbb{R}^d$ and f drawn randomly from \mathfrak{F} . This is clearly the case for the last method, but setting function space to consist of nearest-neighbor classifiers with limited number of base point seems to significantly restrict the possibilities for target function f, and such restriction may be unjustified. In this paper, we choose another approach, and a realistic model of feature space classification structure will be proposed in the next section.

4 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 instances 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.

Thus, we assume that the classification of a feature space is a sample function of a binary valued homogeneous isotropic random field (Wong & Hajek, 1985) characterized by a covariance function decreasing with distance (see (Eldar, Lindenbaum, Porat & Zeevi, 1997), where a similar method was used for progressive image sampling). That is: let x_0, x_1 be points in \mathcal{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. $E[\theta_0] = E[\theta_1] = \overline{\theta}$, and that the covariance between θ_0 and θ_1 is specified only by the distance between x_0 and x_1 :

$$Cov[\theta_0, \theta_1] = E[(\theta_0 - \overline{\theta})(\theta_1 - \overline{\theta})] \triangleq \gamma(d(x_0, x_1))$$
(10)

where $\gamma : \mathbb{R}^+ \to (-1, 1)$ is a covariance function with $\gamma(0) = Var[\theta] = E[(\theta - \overline{\theta})^2] = P_0P_1$, 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 $\lim_{r\to\infty} \gamma(r) = 0$. This idea is illustrated in Figure 8. Note that while specifying the covariance does not uniquely determine the random field and the distribution of target functions, it limits them considerably and makes them "similar" in a sense. If, for example, the covariance is substantial for close points and decreases with distance (as we assume), then the labels of close points are expected to be identical.

We shall now describe several ways of calculating estimates of these conditional probabilities, using the underlying random field model.



Figure 8: The labels of every pair of points, such as x_0 and x_1 , are correlated with covariance decreasing with the distance between the points. Thus points in the region A are more likely to have the same classification as x_0 than points in region B.

4.1 Calculating the probabilities from correlation data by mean-square estimation of the conditional mean

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

Having only two classes implies that 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, common 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_0 be the binary r.v. associated with some unlabeled feature vector, x_0 , and let $\theta_1, \ldots, \theta_n$ be the (known) r.v. associated with the feature vectors, x_1, \ldots, x_n , that were already labeled. A linear estimator of the unknown label θ_0 is:

$$\hat{\theta}_0 = \alpha_0 + \sum_{i=1}^n \alpha_i \theta_i.$$
(11)

The estimate uses the known labels and relies on the coefficients α_i , i = 1, ..., n. The construction of such estimators is a common statistical procedure (Papoulis, 1991). The optimal linear estimator, minimizing the MSE $\epsilon_{mse} = E[(\hat{\theta} - \theta_0)^2]$ is

$$\hat{\theta} = E[\theta_0] + \vec{\mathbf{a}} \cdot (\vec{\theta} - E[\vec{\theta}])^t$$
(12)

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

$$\vec{\mathbf{a}} = \mathbf{R}^{-1} \cdot \vec{\mathbf{r}},$$

$$R_{ij} = E \left[(\theta_i - E[\theta])(\theta_j - E[\theta]) \right],$$

$$r_i = E \left[(\theta_0 - E[\theta])(\theta_i - E[\theta]) \right].$$
(13)

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

$$R_{ij} = \gamma(d(x_i, x_j)),$$

$$r_i = \gamma(d(x_0, x_i)).$$
(14)

The procedure is straightforward and easy to implement. In practice only reasonably close labeled points are used to construct the estimate, because far points are not correlated and therefore cannot contribute anything to the estimate implying that their corresponding coefficients become zero. Therefore this estimation procedure is also fast.

One problem with the linear estimation of label probability is that the range of the estimated values is not limited and may lie outside the [0, 1] interval. With the interpretation of $\hat{\theta}$ as probability such values are clearly not valid. Fortunately, this is rarely the case. When such invalid values occur, they are clipped to either 0 or 1, resulting in a legal value (and a more accurate estimate also in the MSE sense).

Another problem with linear estimators is that they use only the information on second-order statistics (covariance matrix), and higher order statistics cannot be used.

4.2 Direct calculation of conditional probabilities

The conditional distribution of θ_0 given the $\theta_1, \ldots, \theta_n$ can also be found from the joint distribution of $\theta_0, \ldots, \theta_n$, since

$$P(\theta_0|\theta_1,\dots,\theta_n) = p(\theta_0,\theta_1,\dots,\theta_n)/p(\theta_1,\dots,\theta_n)$$

= $p(\theta_0,\theta_1,\dots,\theta_n)/\sum_{\omega \in \{0,1\}} p(\theta_0 = \omega,\theta_1,\dots,\theta_n).$ (15)

Let $\vec{\omega} \triangleq (\omega_0, \omega_1, \dots, \omega_n)$, $\vec{\omega} \in \{0, 1\}^{n+1}$ denote the vector of values of $\theta_0, \theta_1, \dots, \theta_n$ and $p_{\vec{\omega}} \triangleq P(\theta_0 = \omega_0, \dots, \theta_n = \omega_n)$ denote the probability of $\theta_0, \dots, \theta_n$ to obtain these values. We can find the joint distribution of $\theta_0, \dots, \theta_n$ from the moment data:

$$\sum_{\vec{\omega} \in \{0,1\}^{n+1}} p_{\vec{\omega}} = 1$$
 sum of probabilities is one.

$$\sum_{\vec{\omega} \in \{0,1\}^{n+1}} \omega_i p_{\vec{\omega}} = P_1 \text{ (for } i=0,\dots,n) \text{ a priori probabilities.}$$
(16)

$$\sum_{\vec{\omega} \in \{0,1\}^{n+1}} (\omega_i - P_1)(\omega_j - P_1) p_{\vec{\omega}} = \gamma(d(x_i, x_j)) \text{ for } i \neq j: i, j \in \{0,\dots,n\}.$$

There are 2^{n+1} unknown variables and only $1 + (n+1) + \frac{n(n+1)}{2}$ equations, so the unique solution can not be obtained for *n* larger than 1. This could have been expected, since the second-order statistics does not specify an arbitrary joint probability function uniquely. Additional constrains must be introduced in order to obtain the unique solution.

Knowing a number of higher moments, $\{\zeta\}_{3}^{n+1}$, gives additional constraints:

ū

$$\sum_{\mathbf{x}\in\{0,1\}^{n+1}} \left[\prod_{k\in I} (\omega_k - P_1) \right] p_{\vec{\omega}} = \zeta_{|I|}(\mathbf{x}_{i\in I})$$
(17)

for $I \subseteq \{0, \ldots, n\}, 3 \leq |I| \leq n + 1, (n \geq 2)$. Note that the constraints specified in Equation 16 are the lower $\zeta_0, \zeta_1, \zeta_2$ moments. This linear system (Equations 16 and 17), implied by all moments consists of $1 + (n + 1) + \frac{n(n+1)}{2} + \sum_{i=3}^{n+1} {n+1 \choose i} = \sum_{i=0}^{n+1} {n+1 \choose i} = 2^{n+1}$ equations and 2^{n+1} unknown variables denoting the joint probabilities of $\theta_0, \ldots, \theta_n$, and thus can be uniquely solved. Note, however, that the intuition on which the nearest-neighbor classifier is based, implies that the second-order moments are positive and decreasing with distance. Making further assumptions on higher order moments constraints the distribution beyond the basic nearest-neighbor rationale, and may be unjustified. Note also that arbitrarily chosen moments may be inconsistent with any binary r.v. distribution.

It may be natural, in a certain context, to assume that both class probabilities are equal, $P_0 = P_1 = \frac{1}{2}$, and the distribution is "symmetric". That is,

$$\forall \vec{\omega} \in \{0, 1\}^{n+1}, p_{\vec{\omega}} = p_{\neg \vec{\omega}} \tag{18}$$

This readily implies that all odd moments are zero, $\zeta_{2k+1} \equiv 0, k \in \mathbb{N}$. Interestingly, the symmetry property (18) holds if and only if $P_0 = P_1$ and $\zeta_{2k+1} \equiv 0, \forall k \in \mathbb{N}, k \leq \lfloor \frac{n+1}{2} \rfloor - 1$.

This assumption is much stronger than just assuming that $P_0 = P_1$, but if it is believed to hold, it provides the additional third order moments, and allows to compute the joint distribution of three random variables from Equations 16 and 18 alone.

In contrast to the linear estimation approach which relies only on the second-order statistics of the distribution and is therefore non-optimal for non-Gaussian distributions, the direct method described above tries to calculate the exact joint distribution. Therefore, it requires more information, in the form of higher order statistical moments. On the other hand, it guarantees that the estimated probabilities lie in the legal [0, 1] interval (provided that all moments that are used, are indeed correct).

Both methods are associated with some theoretical deficiency, to be explained and solved below.

4.3 A modified random field model: Excursion set

As mentioned above, there are some deficiencies in the probability estimation methods that we described. The MSE approximation of the conditional mean can yield an invalid estimate of the mean, outside the [0, 1] range. The direct calculation does not have this problem but requires the use of a set of higher order moments which are not justified by the nearest-neighbor rationale.

The major theoretical deficiency, as we see it, is that an arbitrarily specified covariance function, $\gamma(d)$, may not correspond to a *binary* random field. That is, there is no guarantee that there indeed exists some random field with binary outcomes and with the chosen covariance function. One way to solve this problem is to specify the labels indirectly, using an additional, *real*-valued random field. Now, from the distribution of this "hidden" random field, we can infer the covariance of the binary field.

Define a Gaussian random field as a collection of random variables $Y(x), x \in \mathbb{R}^d$, such that all finite dimensional distributions of variables Y(x) are multivariate Gaussians (Adler, 1981). For simplicity, we also assume that this random field is homogeneous and isotropic, i.e. $E[Y(x)] \equiv \bar{Y}$ and the covariance between $Y(x_1)$ and $Y(x_2)$ depends only on the distance between x_1 and x_2 , $Cov[Y(x_1), Y(x_2)] \triangleq \gamma^*(d(x_1, x_2))$ (similarly to Equation 10). Note that all finite dimensional distributions of Y's are uniquely defined by γ^* and \bar{Y} .

Now we can model the (random) *concept class* (the set of all points of class 1 in \mathbb{R}^d), to be the (random) *excursion set* (Adler, 1981) above the level 0:

$$A_0(\mathbf{Y}, \mathbb{R}^d) = \{ x \in \mathbb{R}^d : \mathbf{Y}(x) \ge 0 \}$$

$$\tag{19}$$

of the real-valued random field $Y(x), x \in \mathbb{R}^{d}$. Then, the binary class label is:

$$\theta(x) = \begin{cases} 0, & Y(x) < 0\\ 1, & Y(x) \ge 0. \end{cases}$$
(20)

For any value assignment, the joint probability $p(\theta_0 = \omega_0, \theta_1 = \omega_1, \dots, \theta_n = \omega_n)$ may be calculated by (numerically) integrating the multivariate Gaussian distribution of Y_0, \dots, Y_n :

$$p(\theta_0 = \omega_0, \dots, \theta_n = \omega_n) = \int_{a_0}^{b_0} \dots \int_{a_n}^{b_n} \frac{1}{(2\pi)^{\frac{n+1}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(\vec{Y} - \mu_{\vec{Y}})\Sigma^{-1}(\vec{Y} - \mu_{\vec{Y}})^t} d\vec{Y}.$$
 (21)

where $a_i = \begin{cases} -\infty, & \omega_i = 0 \\ 0, & \omega_i = 1 \end{cases}$, $b_i = \begin{cases} 0, & \omega_i = 0 \\ \infty, & \omega_i = 1 \end{cases}$ and $\Sigma, \ \mu_{\bar{Y}}$ are defined by the model, that is $\Sigma[i,j] = \gamma^*(d(x_i, x_j))$ and $\mu_{\bar{Y}}[i] = \bar{Y}$.

The γ^* function must be *non-negative definite*, that is for any finite collection of $x_1, x_2, \ldots, x_k \in \mathbb{R}^d$, the covariance matrix C, where $C_{ij} = \gamma^*(d(x_i, x_j))$, is non-negative definite. Knowing the prior probabilities specifies the Gaussian distribution so that the following relation,

$$P_1 = P(\theta = 1) = \int_0^\infty (2\pi\gamma^*(0))^{-\frac{1}{2}} e^{-\frac{1}{2}\frac{(t-\bar{\Upsilon})^2}{\gamma^*(0)}} dt$$
(22)

is satisfied. This relation may be considered as a condition defining the mean when the prior probability is known and the variance is specified. (This framework resembles the Gaussian Process Modeling (MacKay, 1998; Williams & Barber, 1998), although we use a different transformation to get binary r.v.)



Figure 9: The log-plot for the covariance function of binary r.v., $\gamma(d)$, computed numerically from covariance function of the underlying Gaussian random field, $\gamma^*(d) = e^{-d}$ ($\alpha = 1$). Except for a very small values of d the plot is almost linear indicating that $\gamma(d)$ can be approximated by an exponential function.

Note that the nearest-neighbor method rationale may be satisfied also by specifying the covariance (associated with the "hidden" Gaussian process) as a decreasing function of the distance between the points. Such a specification induces (another) covariance associated with the binary field, which is also a decreasing function of the distance. Therefore, the intuitive properties of the field are satisfied. We show below that these covariance functions are not necessarily very different. The main purpose of introducing the excursion set is to make the specification of the covariance legal. A secondary benefit associated with the choice of the Gaussian distribution is that for such choice the random field is specified by its second-order statistics, namely by the covariance function associated with the hidden r.v., and no higher order statistics are required.

The excursion set model may be used directly to produce predictions on the conditional probabilities of the labels. It is, however, associated with high computational cost. Another way to use it would be to derive the covariance function of the binary random field from it, thus guaranteeing its validity.

Consider the situation when $P_0 = P_1 = \frac{1}{2}$ ($\bar{Y} = 0$) and the exponential covariance function of the hidden, real-valued, random field is $\gamma^*(d) = e^{-\alpha d}$. Using the equality $Cov[\theta_1, \theta_2] = P(\theta_1 = 1, \theta_2 = 1) - 0.25$, it is possible to (numerically) calculate the covariance function of the binary random field $\gamma(d)$. As we can see in Figure 9, the resulting function is very close to exponential. Thus, if we choose an exponential covariance function for a binary valued random field in an experimental implementation of our algorithm (see Section 5), it is at least a very good approximation to a valid covariance, which is justified theoretically. This is the method adopted in our experiments.

5 Lookahead Selective Sampling Using a Random Field Model

The previous chapters describe a general framework for lookahead-based selective sampling as well as a spectrum of methods for probability estimation. In this chapter the proposed algorithm is described in details including its instantiation.

With the probability estimation methods described above, every sampled point influences the estimated probability. Such long-range influence is non-intuitive and is also computationally expensive. Technion - Computer Science Department - Technical Report CIS9906 - 1999

Therefore, in practice, we neglect the influence of all except the two closest neighbors. Such choice gives a higher probability to the nearest-neighbor class and is therefore consistent with 1-NN classification.

Using only two closest labeled neighbors to estimate the conditional probabilities of unknown points we allow the problem to be solved by the methods described in Sections 4.1 and 4.2. Using Equations 12-14 for the estimation of θ_0 from θ_1, θ_2 we get:

$$\vec{\mathbf{r}} = [\gamma(d_{01}), \gamma(d_{02})], \mathbf{R} = \begin{bmatrix} \gamma(d_{11}) & \gamma(d_{12}) \\ \gamma(d_{21}) & \gamma(d_{22}) \end{bmatrix} = \begin{bmatrix} P_0 P_1 & \gamma(d_{12}) \\ \gamma(d_{12}) & P_0 P_1 \end{bmatrix}$$

$$\hat{\theta}_0 = P_1 + \begin{bmatrix} \frac{\gamma(d_{02})\gamma(d_{12}) - \gamma(d_{01})P_0 P_1}{\gamma(d_{12})^2 - P_0^2 P_1^2}, \frac{\gamma(d_{01})\gamma(d_{12}) - \gamma(d_{02})P_0 P_1}{\gamma(d_{12})^2 - P_0^2 P_1^2} \end{bmatrix} \cdot \left(\begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} - \begin{bmatrix} P_1 \\ P_1 \end{bmatrix} \right)$$
(23)

where $d_{ij} = d(x_i, x_j)$ and $\langle x_1, \omega_1 \rangle, \langle x_2, \omega_2 \rangle \in D$ are the first and second nearest neighbors of $x_0 \in X$. The $\hat{\theta}_0$ value is an approximation for $P(\theta_0 = 1 | \theta_1, \theta_2)$ (see Section 4.1).

For the implementation of our algorithm, we assume equal a priori class probabilities which are justified at the initial stages of the learning process, when a priori class probabilities can not be estimated. Simplifying and substituting $P_0 = P_1 = \frac{1}{2}$ we get the conditional probabilities:

$$P(\theta_{0} = 1 | \theta_{1} = 0, \theta_{2} = 0) = \frac{1}{2} + \frac{-\gamma(d_{01}) - \gamma(d_{02})}{\frac{1}{2} + 2\gamma(d_{12})}$$

$$P(\theta_{0} = 1 | \theta_{1} = 0, \theta_{2} = 1) = \frac{1}{2} + \frac{-\gamma(d_{01}) + \gamma(d_{02})}{\frac{1}{2} - 2\gamma(d_{12})}$$

$$P(\theta_{0} = 1 | \theta_{1} = 1, \theta_{2} = 0) = \frac{1}{2} + \frac{\gamma(d_{01}) - \gamma(d_{02})}{\frac{1}{2} - 2\gamma(d_{12})}$$

$$P(\theta_{0} = 1 | \theta_{1} = 1, \theta_{2} = 1) = \frac{1}{2} + \frac{\gamma(d_{01}) + \gamma(d_{02})}{\frac{1}{2} + 2\gamma(d_{12})}.$$
(24)

These values are equal to the values we get by "direct" probability calculations (Equations 16 and 17) by assuming $\zeta_3 \equiv 0$ and $P_0 = P_1 = \frac{1}{2}$ (or assuming the "symmetry" property, see Equation 18). This is due to the fact that setting these conditions actually means that we are incorporating only first and second order statistics into probability calculations. In this way we are making an intrinsic assumption about distribution of $\theta_0, \theta_1, \theta_2$ being Gaussian, thus allowing the mean of conditional distribution of θ_0 for given θ_1, θ_2 to behave as a linear function of θ_1 and θ_2 .

We choose an exponentially decreasing covariance function $\gamma(d) = 0.25e^{-d/\sigma}$. Such function is a good approximation for a covariance function, which can be computed numerically from the excursion set model (Section 4.3), thus providing our approach with theoretical justification. Given $P_0 = P_1 = \frac{1}{2}$ and $\gamma(d) = 0.25e^{-d/\sigma}$ the Bayesian classification of x_0 is defined by the label of x_1 , i.e. $P(\theta_0 = 1|\theta_1, \theta_2) > \frac{1}{2} \Leftrightarrow \theta_1 = 1, (d_{01} \neq d_{02})$, thus being consistent with the nearest-neighbor classification.

Assuming $\gamma(d)$ to approximate the "correct" covariance function derived from an excursion set model does not guarantee that $P(\theta_0|\theta_1, \theta_2)$ will be in valid range of [0, 1]. To show the correctness of the choice of γ we need to show that

$$\forall x_0, x_1, x_2 \in \mathbb{R}^d: \ 0 \le P(\theta_0 | \theta_1, \theta_2) \le 1.$$

$$(25)$$

The main step is to prove that $\frac{\gamma(d_{01})-\gamma(d_{02})}{\frac{1}{2}-2\gamma(d_{12})} \leq \frac{1}{2}$ (the rest is similar or trivial, keeping in mind that $d_{01} \leq d_{02}$ and d_{01}, d_{02}, d_{12} are subject to triangular inequality). After substituting $\gamma(d) = \frac{1}{4}e^{-d}$ (omitting σ as a scaling parameter) and simplifying, it must be shown that: $e^{-d_{01}} - e^{-d_{02}} + e^{-d_{12}} \leq 1$. Using the triangular inequality $d_{02} \leq d_{01} + d_{12}$ and the fact that $e^{-d_{ij}} \leq 1$ (for $d_{ij} \geq 0$), we get: $e^{-d_{01}} - e^{-d_{02}} + e^{-d_{12}} \leq e^{-d_{12}} + e^{-d_{01}} - e^{-d_{01}}e^{-d_{12}} \leq 1 \square$. The common forms of the resulting probability distributions are illustrated in Figure 10.

For the experiments, we implemented the one-step lookahead selective sampling algorithm, which uses the accuracy utility function and the above method of estimating the conditional class probabilities. The algorithm consists of two parts: the initialization, illustrated in Figure 11, which is used to define the covariance function, and a selective sampling procedure, described in Figure 12, which uses the lookahead selective sampling method.

The algorithm needs a specification of the σ parameter, which we set depending on the average pairdistance in the instance space and on the a priori scale parameter \mathfrak{D} , which is set to four. The effect of the choice of \mathfrak{D} is considered in the separate set of experiments (Section 6.3).



Figure 10: The conditional probabilities of class 1 as a function of location, in one dimension, $\gamma(d) = 0.25e^{-d}$ ($\sigma = 1$): (a) Point of class 1 at x = 5 and point of class 0 at x = 7. (b) Two points labeled as class 1 at x = 5 and x = 7.

Initialize-Selective-Sampling-Algorithm(X):

1. Compute average pair-wise distance in X.

- 2. $\sigma \leftarrow \text{average-pair-distance} / \mathfrak{D}$
- 3. Set $\gamma(d)$ equal to $0.25e^{-d/\sigma}$.

Figure 11: Initialization of the lookahead selective sampling algorithm. The only parameter is the scale of the covariance function, \mathfrak{D} , which is set to be the quarter of the average-pair-distance. The experiments show the stability of algorithm upon wide range of \mathfrak{D} parameter.

The time complexity of the described algorithm is $O(|X|^2)$ for the straightforward implementation. This time complexity, however, may be reduced by selecting a random subset $X' \subseteq X$ and working with it instead of X, thus managing the time/performance trade-off of this algorithm.

6 Experimental Evaluation

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

6.1 Experimental methodology

The lookahead algorithm (with $\mathfrak{D} = 4$) was compared with the following three selective sampling algorithms representing the most common choices found in the literature (see Section 1):

- 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. This is one of the most common choices for selective sampling (Section 1). No specific uncertainty sampling method, however, was proposed for the nearest-neighbor classifiers. We defined the uncertainty for each example as a probability of its misclassification, which is estimated

- 1. If D is empty, return random point from X.
- 2. Otherwise, set $U_{max} \leftarrow 0$.
- 3. For each $x \in X$ do:
 - (a) $D' \leftarrow D \cup \{\langle x, 0 \rangle\}.$
 - (b) Compute class probabilities for all points in X based on data D', using Equation 24.
 - (c) Compute utility by approximating the accuracy of the classifier based on data D' using computed class probabilities and Equation 2, $U_0(x) \leftarrow \mathcal{A}(D')$.
 - (d) Repeat the above steps for $D' \leftarrow D \cup \{\langle x, 1 \rangle\}$ and get $U_1(x)$.
 - (e) Compute class probabilities for x based on data D, using Equation 24.
 - (f) $U(x) \leftarrow P(f(x) = 0|D) \cdot U_0(x) + P(f(x) = 1|D) \cdot U_1(x)$.
 - (g) If $U_{max} < U(x)$ then $U_{max} \leftarrow U(x)$, $x_{best} \leftarrow x$.

4. Return x_{best} .

Figure 12: The body of the lookahead selective sampling algorithm. The algorithm uses a covariance function defined at the initialization stage.

by the ratio between the distances to the closest labeled neighbors of different classes (see Equation 7). This method tends to sample on the existing border, without considering exploration in other regions, and while for some decision boundaries this may be beneficial, for others it may be a source of a serious failure (as will be shown in Section 6.2.6).

• *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 which 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 experiments were conducted on seven datasets. Among them there were three natural datasets: Pima Indians Diabetes dataset, Ionosphere dataset and Image Segmentation dataset, one synthetic dataset: Letters dataset and three artificial problems: Two-Spirals problem, Two-Gaussians problem and Multi-Gaussian problem. The Gaussian problems were included so that we can control the geometry of the instance space in order to illustrate the benefits of the lookahead selective sampling. The source of each dataset is indicated in the corresponding section.

The basic quantity measured in the experiments was the average error rate of the classifier based on the training points selected by the selective sampling algorithm. For the real datasets the following procedure was applied:

- The training set and the test sets were obtained from the data. All natural datasets which we used were already divided into training and test sets according to the past usages. The training set was used as an instance space, X, for a selective sampling algorithm. The test set was used only for the evaluation of error rates of the resulting classifiers.
- 2. The selective sampling algorithms were applied to the training set, X. After selecting each example, the error rate of the current hypothesis, h (which is the nearest-neighbor classifier), was calculated

based on 20 selected examples (10% of the training set). Statistics based on 100 runs.				
Selective sampling method	Average error rate	Standard deviation		
Random	30.2%	$\pm 4.4\%$		
Uncertainty	30.3%	$\pm 7.8\%$		
Maximal Distance	29.8%	$\pm 5.3\%$		
Lookahead	26.9%	$\pm 1.7\%$		

Table 1: Pima Indians Diabetes dataset: Average error rates and their standard deviations for classifiers based on 20 selected examples (10% of the training set). Statistics based on 100 runs.

using the test set of examples put aside.

3. Steps 1, 2 were performed 100 times and the average error rate was calculated.

Note that, while the training and test sets for natural datasets remain the same, we are still interested in the average performance, since all non-trivial selective sampling methods are naturally randomized by selection of the first example.

For the artificial datasets (for which we were able to generate an unlimited number of examples) we conducted 100 independent runs, with randomly drawn separate training and test sets of size 1000 elements each, in a manner similar to the one described above.

6.2 The performance of the lookahead selective sampling

6.2.1 Pima Indians Diabetes dataset

This dataset was widely used in the past and it is available from the UCI Machine Learning Repository (Blake, Keogh & Merz, 1998). The data, consisting of 7 dimensional vectors, has already been split into training and test sets of sizes 200 and 332 respectively, which we used in the experiments. In addition, we normalized the data attributes to fit into the [0, 20] interval. In this dataset an error rate of a nearest-neighbor classifier based on the training set was 31%.

The experimental results of comparison between the selective sampling methods are shown in Table 1 and the learning curves are shown in Figure 13. Interestingly, learning only 10% of the training set with the proposed lookahead selective sampling method yields a better classifier than the nearest-neighbor classifier trained on all available points.

6.2.2 Ionosphere dataset

This dataset is available from UCI Machine Learning Repository (Blake, Keogh & Merz, 1998). This data set consists of 34 dimensional vectors belonging to two classes. Similarly to the past usages we used first 200 examples as a training set and the last 151 as a test. The data was normalized with all features transformed to the same range. In this dataset, an error rate of a nearest-neighbor classifier based on the training set was 7.9%.

The results of the selective sampling methods comparison are shown in Table 2, together with learning curves presented in Figure 14. The lookahead selective sampling not only achieves a much lower error rate on this dataset, but is also the most stable one, in terms of its standard deviation. Note, that the lookahead selective sampling needs only 8% of the training set in order to achieve the same average accuracy as the nearest-neighbor classifier based on all training points.

6.2.3 Image Segmentation dataset

The data we used was taken from UCI Machine Learning Repository (Blake, Keogh & Merz, 1998). This dataset originally contained feature vectors with 19 numerical valued attributes which belonged to



Figure 13: Pima Indians Diabetes dataset: Comparison of the selective sampling methods. Learning all points in the training set gives an error of $\approx 31\%$.

Table 2: Ionosphere dataset: Average error rates and their standard deviations for classifiers based on 20 selected examples (10% of the training set). Statistics based on 100 runs.

Selective sampling method	Average error rate	Standard deviation
Random	18.5%	$\pm 8.9\%$
Uncertainty	30.3%	$\pm 21.7\%$
Maximal Distance	19.8%	$\pm 9.2\%$
Lookahead	7.6%	$\pm 2.2\%$

7 classes. We dropped one attribute which was constant for all feature vectors, normalized the attributes to fit into [0, 20] interval and transformed the data to be the binary classification problem by assigning label 0 to original classes of BRICKFACE, SKY, FOLIAGE and label 1 to the classes of CEMENT, WINDOW, PATH, GRASS. This dataset consists of pre-defined training and test sets of sizes 210 and 2100, which we used in our experiments. An error rate of a nearest-neighbor classifier based on the training set was 4.7%.

The average accuracy for a classifiers based on the selected 10% of the learning space is shown on Table 3. The results of comparison of the selective sampling methods learning curves are shown in in Figure 15. Again we observe the superiority of the lookahead selective sampling method in terms of an average error rate and standard deviation. Note that the best non-lookahead selective sampling method, *uncertainty*, needs almost twice the number of examples the *lookahead* method needs to achieve an error rate of 15%.

6.2.4 Letters dataset

This synthetic dataset, which was contributed to the UCI Machine Learning Repository (Blake, Keogh & Merz, 1998) by David Slate (1991), consists of 20000 feature vectors belonging to 26 classes representing the capital letters of Latin alphabet. The features of these characters were summarized in 16 numerical



Figure 14: Ionosphere dataset: Comparison of the selective sampling methods. Learning all points in the training set gives an error of $\approx 7.9\%$.

Table 3: Image Segmentation dataset: Average error rates and their standard deviations for classifiers based on 21 selected examples (10% of the training set). Statistics based on 100 runs.

Selective sampling method	Average error rate	Standard deviation		
Random	16.9%	$\pm 5.5\%$		
Uncertainty	14.8%	$\pm 5.0\%$		
Maximal Distance	26.6%	$\pm 9.3\%$		
Lookahead	10.9%	$\pm 1.4\%$		

attributes that can receive values from the range [0, 15]. We modified the data by changing all letters from 'a' to 'm' to 0 and all the letters from 'n' to 'z' to 1. This dataset was randomly divided into 10 separate pairs of training and test sets (1000 examples each), and 10 experiment runs were conducted on each pair, resulting in total 100 experimental runs for which the average error rate and its standard deviation were reported. The average error rate of a nearest-neighbor classifier based on all points in the training set was $\approx 11.6\%$.

The error rates of the various selective sampling methods using only a 10% of the training set are reported in Table 4, along with the learning curves shown in Figure 16. The lookahead selective sampling algorithm outperforms other selective sampling methods in this particularly hard domain, where every class consists of many different subclasses (associated with the different letters).

6.2.5 Two Spirals problem

We decided to test our method on this particular problem for a comparison, since this dataset was used in the work of Hasenjager and Ritter (1998). This is an artificial problem where the task is to distinguish between two spirals in XY-plane, as shown in Figure 17. The examples were randomly generated from



Figure 15: Image Segmentation dataset: Comparison of the selective sampling methods. Learning all the points in the training set gives an error of ≈ 4.7 .

Table 4: Letters dataset: Average error rates and their standard deviations for classifiers based on 100 selected examples (10% of the training set). Statistics based on 100 runs.

Selective sampling method	Average error rate	Standard deviation
Random	30.8%	$\pm 1.7\%$
Uncertainty	28.6%	$\pm 2.3\%$
Maximal Distance	29.6%	$\pm 4.0\%$
Lookahead	28.2%	$\pm 2.0\%$

a uniform distribution over $[-7,7] \times [-7,7]$ and classified off-line¹. The feature space of this problem is illustrated in Figure 17. The Bayes error of such classification is zero since the classes are perfectly separable and the average error of a nearest-neighbor classifier based on 1000 random labeled points is $\approx 9.1\%$.

The error rates of the various selective sampling methods using only 10% of a training set are shown in Table 5. The learning curves of the selective sampling methods are illustrated in Figure 18. All three non-random methods exhibited comparable performance, with *Maximal-Distance* method being slightly better than others. On the other hand, the *Maximal-Distance* method is the most unstable method. In the next experiment we 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.

6.2.6 Two Gaussians problem

This dataset was specially constructed to demonstrate the advantage of the lookahead selective sampling method in the domains which consist of more than one region of the same classification. The feature

¹We used a code available from [http://www.boltz.cs.cmu.edu/benchmarks/two-spirals.html], and (Lang & Witbrock, 1988) as a basis for our "two spirals" data generation program.



Figure 16: Letters dataset: Comparison of the selective sampling methods. Learning all the feature space gives a classifier with an average error of 11.6%.

Table 5: Two Spirals problem: Average error rates and their standard deviations for classifiers based on 100 selected examples (10% of the training set). Statistics based on 100 independent runs.

Selective sampling method	Average error rates	Standard deviation
Random	27.9%	$\pm 2.1\%$
Uncertainty	26.1%	$\pm 2.5\%$
Maximal Distance	24.5%	$\pm 5.1\%$
Lookahead	26.3%	$\pm 2.0\%$

space of Two Gaussians problem 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 $\Sigma = 2^2 I$, normalized to fit the $[0, 10] \times [0, 10]$ and $[10, 20] \times [10, 20]$ windows. The data is illustrated in Figure 19. The Bayes decision boundary can be derived analytically, and the Bayes error is 18.2%, while the expected error of a nearest neighbor classifier based on 1000 randomly drawn training points is $\approx 18.2\%$.

The average errors of a nearest-neighbor classifiers based on 10% of the training set selected by various selective sampling methods are summarized in Table 6. The learning curves of the various selective sampling methods are shown in Figure 20. 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. The variance of the lookahead selective sampling method is much lower due to the same reason - the lookahead selective sampling algorithm *always* detects both regions, while the uncertainty and the maximal-distance methods discover it only occasionally creating greater variance in the quality of resulting classifiers. The uncertainty and the maximal-distance selective sampling algorithm experience failure due to the fact that these methods consider sampling only at the existing boundary, while in this domain *exploration* is essential for good performance.



Figure 17. The feature space of Two Spirals problem.



Figure 18: Two Spirals problem: Comparison of the selective sampling methods. Learning all the points in the training set gives a classifier with average error 9.1%.



Figure 19. The feature space of Two Gaussians problem.

Table 6: Two Gaussians problem: Average error rates and their standard deviations for classifiers based on 100 selected examples (10% of the training set). Statistics based on 100 independent runs.

Selective sampling method	ng method Average error rate	
Random	27.3%	$\pm 2.6\%$
Uncertainty	32.7%	$\pm 6.0\%$
Maximal Distance	31.4%	$\pm 6.5\%$
Lookahead	24.9%	$\pm 2.1\%$

6.2.7 Multi-Gaussian problem

This artificial problem is similar to the Two Gaussians dataset, but the number of Gaussians is increased, in order to further investigate the sampling style of the lookahead selective sampling algorithm. Similarly to Two Gaussians data, we have 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 nine symmetric Gaussians, with means in the grid (3.5, 10, 16.5) × (3.5, 10, 16.5) and covariance matrix $\Sigma = 1^2 I$. The data is illustrated in Figure 21. The Bayes decision boundary can be approximated analytically, and the Bayes error is 20.8%, while the expected error of a nearest-neighbor classifier based on 1000 points is $\approx 29.9\%$.

The learning curves of the various selective sampling methods are shown in Figure 22 and the numerical results are summarized in Table 7. Again, it can be seen that the uncertainty and the maximaldistance selective sampling methods fail to detect some of the Gaussians, resulting in higher error rates.

The lookahead selective sampling seems to perform bad at the initial stages of the learning. By analyzing the errors in each class (Figure 23) we can understand that such behavior is the result of oversampling in the clusters of class 0, arising from a higher density in these regions.

6.3 The effect of the covariance function on the performance

We carried out additional experiments to determine how much our algorithm depends on the choice of covariance function, in particular, how much it depends on the choice of the scaling parameter \mathfrak{D} . In the experiments reported in the previous section, \mathfrak{D} was set to 4. In this section the scale parameter was set to be twice and four times larger and smaller than those in the original experiments, i.e. $\mathfrak{D} = 1, 2, 8$ and 16.

The dependence of the average error rate on parameter $\mathfrak D$ is shown in Figure 24. The results for



Figure 20: Two Gaussians problem: Comparison of the selective sampling methods. Learning all the points in the training set gives an error of 27.3% and the Bayes error is 18.2%.

Table 7: Multi-Gaussian problem: Average error rates and their standard deviations for classifiers based on 100 selected examples (10% of the training set). Statistics based on 100 independent runs.

Selective sampling method	Average error rate	Standard deviation
Random	35.5%	$\pm 2.9\%$
Uncertainty	39.4%	$\pm 3.1\%$
Maximal Distance	38.5%	$\pm 4.6\%$
Lookahead	33.6%	$\pm 2.6\%$

Ionosphere datasets, typical for the rest of data, are summarized in Table 8. These results demonstrate the stability of the lookahead algorithm upon a wide range of the scale parameter. In addition, we can see that generally the variance of the error rates of the resulting classifiers decreases with \mathfrak{D} . This could have been expected, since increasing \mathfrak{D} means the decreasing of the actual influence range (see initialization algorithm on Figure 11) and thus the higher \mathfrak{D} the more local, and the less depending on the initial random example, is the sampling strategy.

6.4 The effect of the utility function on the performance

In Section 3.2 we propose an alternative exploratory utility function, U_L^{ch} . We performed an experiment to compare this function to our standard utility function. The results are shown in Table 9. For most datasets the exploratory utility function yields the slight improvement over the accuracy-based utility function. Noticeable exception is the noisy Pima Indians Diabetes dataset, where the more conservative accuracy based utility function has an advantage.

6.5 Summary of experimental results

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



Figure 21. The feature space for Multi-Gaussian problem. Bayes decision boundaries are shown.



Figure 22: Multi-Gaussian problem: Comparison of the selective sampling methods. The Bayes error is $\approx 20.8\%$ and learning all the points in the learning space gives an average error of $\approx 29.9\%$.





Figure 23: Multi-Gaussian problem: (a) errors in class 0, (b) errors in class 1. This Figure supports the claim that the lookahead selective sampling algorithm tends to sample first in the dense clusters (class 0).



Figure 24: Average error rate (in percent) as a function of the scale parameter \mathfrak{D} . Statistics based on 100 runs, sampling 10% of the instance space in each run.

space contains more than one region of some class. Then, the selective sampling algorithm must consider not only the examples from the hypothesis boundary, but also explore large unsampled regions. The lack of an "exploration" element in uncertainty and maximal-distance sampling methods often results in the failure of these methods.

The advantage of a lookahead selective sampling method can be seen by comparing the number of examples needed to reach some pre-defined accuracy level. Figure 25 shows a number of examples needed for various selective sampling algorithms to reach the average error level, equal to that achieved by the lookahead selective sampling using only 5% of the instance space. Note that in the Pima Indians Diabetics domain, other selective sampling methods never achieve accuracy of the lookahead sampling (due to noise).

The experiments indicate that the proposed selective sampling approach is more stable than the alternatives in the following two aspects:

Stability within a domain As apparent from the variance of the error rates of the resulting classifiers,

Table 8: Variations of \mathfrak{D} parameter, Ionosphere dataset: Average error rates and their standard deviations for classifiers based on 20 selected examples (10% of the training set). Statistics based on 100 runs.

Selective sampling method	Average error rate	Standard deviation
Lookahead, $\mathfrak{D} = 1$	10.6%	$\pm 6.4\%$
Lookahead, $\mathfrak{D} = 2$	7.4%	$\pm 3.3\%$
Lookahead, $\mathfrak{D} = 4$	7.6%	$\pm 2.2\%$
Lookahead, $\mathfrak{D} = 8$	7.2%	$\pm 0.6\%$
Lookahead, $\mathfrak{D} = 16$	8.4%	$\pm 0.6\%$

Table 9: Average error rates for the lookahead selective sampling algorithm using the two alternative utility functions.

Utility	Pima	Ionosphere	Image	Letters	Two	Two	Multi
	Indians		Segm.		Spirals	Gaussians	Gauss
accuracy	26.9%	7.6%	10.9%	28.2%	26.3%	24.9%	33.6%
exp-change	30.0%	6.7%	9.5%	27.5%	25.0%	24.2%	34.0%

the lookahead selective sampling algorithm is the most stable one with regard to different training sets within a domain and to the different choices of the first example.

Stability over domains The results, described in Tables 1-7, show that while the second best selective sampling method changes, the lookahead selective sampling algorithm remains the best and the most stable among the compared methods. This can also be seen in Figure 26, which brings the results of different experiments together.

7 Discussion

In many real-world domains unlabeled examples are available in large quantities, while it is expensive to label a large number of examples for training. A possible solution of this problem, investigated in this paper, is to provide the learning algorithm with an ability to automatically select the training examples from an unlabeled instance space. In the context of the nearest-neighbor classification, this paradigm can be viewed as an example filtering in addition to the *selective utilization* filtering implemented in the IB3 algorithm (Aha, Kibler & Albert, 1991).

This paper proposes a *lookahead* framework for selective sampling, which selects an optimal example in the Bayesian sense. This framework and a novel, *random field* model of the instance space labeling structure, are the major contributions of this research to the field of machine learning.

The random field model which we use was inspired by the rationale for the nearest-neighbor classification. Nearest-neighbor classifiers are often used when little or no information is available about the feature space structure. In this case the loose, minimalistic specification of the feature space labeling structure implied by the distance-based random field model, seems to be appropriate. We also observe that large changes in the covariance function had no significant effect on the classification performance.

Our algorithm, however, has a number of deficiencies, which can be addressed in future research. Considering the classification of one point (including finding 1 or 2 labeled neighbors) a basic operation, the uncertainty and the maximal-distance methods have time complexity of O(|X|) while the straightforward implementation of the 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



Figure 25: Comparing a number of examples needed for the average learning error of various methods to achieve the level equal to one that lookahead algorithm achieves at 5% of the learning sets. In each group from left to right: random, uncertainty, maximal distance and lookahead sampling methods. Note that in "Pima Indians" domain, other selective sampling methods never achieve accuracy of the lookahead sampling algorithm (due to noise).

resources to save time for a human expert whose role is to label examples.

Another deficiency, and a possible direction of further research, lies in the application of the random field model as a probability estimation tool for a nearest neighbor classification. As it was mentioned in Section 5, this model is consistent with 1-NN classification only if we can neglect the influence of all labeled neighbors except the two closest to the current point of interest. In some cases, however, taking only two neighbors into account is not an optimal strategy.

Consider, for example, n labeled points distributed uniformly within a unit ball of d dimensions, and suppose the point of interest lies in the center of the ball. The higher the dimensionality, the larger part of the ball weight tends to be on the outer sphere; hence the difference in distances from the point of interest to closest labeled points in the sphere decreases with dimensionality. This example supports the claim that in order to define the classification of the point of interest in higher dimensions one should consider a larger number of nearest neighbors.

A natural extension of this research is the application of the lookahead sampling to other models of feature space classification structure or to another classification algorithm. The issue of reducing the time complexity of a lookahead sampling should also be further investigated, where one possibility is to define the cost model, i.e. an amount of resources/time one can allow in order to select the next example.

We believe that the research presented in this paper is a significant contribution to the emerging field of selective sampling becoming especially relevant in modern data mining applications.

8 Acknowledgments

The authors would like to thank Neri Merhav for the helpful discussions.

References

Adler, R. J. (1981). The Geometry of Random Fields. Wiley series in probability and mathematical statistics. John Wiley & Sons.



Figure 26: Average error rates (in percent) for compared selective sampling methods. Results after sampling 10% of the instance space.

- Aha, D. W., Kibler, D., & 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., & Merz, C. (1998). UCI Repository of machine learning databases [http://www.ics.uci.edu/~mlearn/MLRepository.html] University of California, Irvine, Dept. of Information and Computer Sciences.
- Cohn, D. A., Atlas, L., & Lander, R. (1994). Improving generalization with active learning. Machine Learning, 15(2), 201-21.
- Cohn, D. A., Ghahramani, Z., & Jordan, M. I. (1995). Active learning with statistical models. In Advances in Neural Information Processing Systems, volume 7, (pp. 705-12). MIT Press.
- Cover, T. M. & Hart, P. E. (1967). Nearest neighbor pattern classification. IEEE Transactions on Information Theory, 13(1), 21-7.
- Dagan, I. & Engelson, S. P. (1995). Committee-based sampling for training probabilistic classifiers. In Machine Learning - International Workshop then Conference - 1995; conf 12, (pp. 150-7). Morgan Kaufmann.
- Davis, D. T. & Hwang, J.-N. (1992). Attentional focus training by boundary region data selection. In *IJCNN* International Joint Conference on Neural Networks, volume 1, (pp. 676-81). IEEE, IEEE.
- Duda, R. O. & Hart, P. E. (1973). Pattern Classification and Scene Analysis. Wiley-Interscience.
- Eldar, Y., Lindenbaum, M., Porat, M., & Zeevi, Y. Y. (1997). The farthest point strategy for progressive image sampling. *IEEE Transactions on Image Processing*, 6(9), 1305-15.
- Fedorov, V. V. (1972). Theory of optimal experiments, volume 12 of Probability and mathematical statistics. New York: Academic Press. translation of Teoriia optimalnogo eksperimenta.
- Freund, Y., Seung, H. S., Shamir, E., & Tishbi, N. (1997). Selective sampling using the query by committeee algorithm. *Machine Learning*, 28(2-3), 133-68.

- Frey, P. W. & Slate, D. J. (1991). Letter recognition using holland-style adaptive classifiers. Machine Learning, 6(2), 161-82.
- Hasenjager, M. & Ritter, H. (1996). Active learning of the generalized high-low-game. In Artificial Neural Networks, International Conference Proceedings, (pp. xxv+922, 501-6). Springer-Verlag.
- Hasenjager, M. & Ritter, H. (1998). Active learning with local models. Neural Processing Letters, 7(2), 107-17.
- Lang, K. J. & Witbrock, M. J. (1988). Learning to tell two spirals apart. In Proceedings of the Connectionist Models Summer School, (pp. 52-9). Morgan Kaufmann.
- Lewis, D. D. & Catlett, J. (1994). Heterogeneous uncertainty sampling for supervised learning. In Machine Learning - International Workshop then Conference - 1994; conf 11, (pp. 148-56).
- MacKay, D. J. (1998). Introduction to gaussian processes. NATO ASI series. Series F, Computer and system sciences., 168, 133.
- MacKay, D. J. C. (1992). Information-based objective functions for active data selection. Neural Computation, 4(4), 590-604.
- Markovitch, S. & Sella, Y. (1996). Learning of resource allocation strategies for game playing. Computational Intelligence, 12(1), 88-105.
- Papoulis, A. (1991). Probability, Random Variables, and Stohastic Processes (3rd ed.). McGraw-Hill series in electrical engineering, Communications and signal processing. McGraw-Hill, Inc.
- RayChaudhuri, T. & Hamey, L. (1995). Minimisation of data collection by active learning. In IEEE International Conference on Neural Networks Proceedings, volume 3, (pp. 6 vol. 1+3219, 1338-41). IEEE, IEEE.
- Scott, P. D. & Markovitch, S. (1993). Experience selection and problem choice in an exploratory learning system. Machine Learning, 12, 49-67.
- Seung, H. S., Opper, M., & Sompolinsky, H. (1992). Query by committee. In Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory, (pp. v+452, 287-94). ACM; New York, NY, USA.
- Williams, C. K. I. & Barber, D. (1998). Bayesian classification with gaussian processes. IEEE Transactions on Pattern Analysis and Machine Intelligence, 20(12), 1342.

Wong, E. & Hajek, B. (1985). Stohastic Processes in Engineering Systems. Springer-Verlag.