Completely Lazy Learning

E. K. Garcia, S. Feldman, M. R. Gupta, and S. Srivastava

Abstract—Local classifiers are sometimes called lazy learners because they do not train a classifier until presented with a test sample. However, such methods are generally not completely lazy, because the neighborhood size k (or other locality parameter) is usually chosen by cross-validation on the training set, which can require significant preprocessing and risks overfitting. We propose a simple alternative to cross-validation of the neighborhood size that requires no pre-processing: instead of committing to one neighborhood size, average the discriminants for multiple neighborhoods. We show that this forms an expected estimated posterior that minimizes the expected Bregman loss with respect to the uncertainty about the neighborhood choice. We analyze this approach for six standard and state-of-the-art local classifiers, including discriminative adaptive metric kNN (DANN), a local support vector machine (SVM-KNN), hyperplane distance nearest-neighbor (HKNN) and a new local Bayesian quadratic discriminant analysis (local BDA). The empirical effectiveness of this technique vs. cross-validation is confirmed with experiments on seven benchmark datasets, showing that similar classification performance can be attained without any training.

Index Terms—lazy learning, Bayesian estimation, cross-validation, local learning, quadratic discriminant analysis

1 Introduction

Local classifiers such as kNN are sometimes called *lazy learners* [1] because they wait to see the test sample before learning a classifier. Achieving competitive error rates on practical problems [2], [3], [4], [5], Local classifiers can trivially adapt to evolving training data and are suitable for problems where one cannot assume that training and test samples are drawn from the same distribution. For instance, consider the problem of recommending a webpage to a user given this user's feedback about previously visited webpages: there is a growing training set, the user's preferences are evolving, and the training and test data are almost certainly not drawn iid from some distribution. For such a problem, classifying a new webpage based only on training samples similar to the test sample (local learning) is a compelling choice. Although our focus here will be on classification, much of this treatment applies analogously to local regression and other local learning tasks.

Local learning requires a definition of locality that is usually chosen to be a cross-validated number of neighbors, as in kNN. Many algorithms exist for finding nearest-neighbors efficiently [6], [7], [8], even for evolving data [9], uncertain training data [10], or non-metric spaces [11], [12]. However, training the neighborhood size k for local classifiers requires added system complexity and costs. For example, given an evolving database with around one million training samples, continuously re-cross-validating or re-training the choice of k is burdensome on the system and will add complexity and fragility to the codebase.

- *E. K. Garcia, S. Feldman, and M. R. Gupta are with the Department of Electrical Engineering, University of Washington, Seattle WA 98195. E-mail: gupta@ee.washington.edu.*
- *S. Srivastava is with the Fred Hutchinson Cancer Research Center.*

The main contribution of this paper is demonstrating that simply averaging local probabilities/discriminants over a reasonable set of neighborhoods performs similarly to cross-validating one neighborhood size for local classifiers but without the costs or complexity of cross-validation. This is verified for six different local classifiers on seven benchmark datasets. Further, we show how this can be interpreted as a Bayesian approach to estimating the neighborhood size, and refer to it as *Bayesian neighborhoods*. A secondary contribution is presenting a local version of the Bayesian quadratic discriminant analysis classifier [13], [14]. As with classical quadratic discriminant analysis (QDA) [5] the global version of this classifier has significant model bias that can be greatly reduced by applying it locally. Combined with the proposed Bayesian neighborhoods, the *local Bayesian quadratic discriminant analysis* (local BDA) proposed herein approximately minimizes expected misclassification error with respect to uncertainty in both the neighborhood and the locally modeled class-conditional Gaussians and is a state-of-the-art classifier that is also completely lazy.

First, we review related approaches to finding neighborhoods for local learning. Then in Section 3, we motivate and define the proposed Bayesian neighborhoods and discuss computational complexity. In Section 4, we analyze how Bayesian neighborhoods affect different categories of local classifiers, and introduce the local BDA classifier. In Section 5, we present experiments that compare Bayesian neighborhoods to cross-validation on benchmark datasets for six standard and state-of-the-art local classifiers: kNN, a local linear SVM (SVM-KNN) [2], nearest-hyperplane kNN (HKNN) [15], discriminant adaptive nearest-neighbor (DANN) [16], local ridge regression classification (local ridge), and local BDA. The paper concludes in Section 6 with a summary of results and open questions.

2 Related Work

Cross-validation is by far the most common way to choose the neighborhood size for local classifiers. In M-fold cross-validation [5], the test error is approximated by the classification error on held-out portions of the training data. Given a set of potential neighborhood sizes $\{k_1, k_2, \ldots, k_\kappa\}$, the training set is partitioned into M roughly equal-sized parts and each training sample is classified using the k_i nearest points outside its own part. For each k_i, the resulting error is averaged to produce the sequence of *cross-validation errors* $\{CV(k_1), CV(k_2), \ldots, CV(k_\kappa)\}$, the minimizer of which is chosen as the "optimal" neighborhood size. Though successful in practice, cross-validation suffers from many drawbacks, including: a) prohibitively long training time on large datasets, b) the unjustified assumption that there is a single optimal value of k for the entire feature space, and c) a lack of a rigorous method for choosing the set of potential neighborhood sizes. The theory behind cross-validation assumes that the training and test datasets are iid which is often unrealistic in practice. For some applications, such as speech processing, learning from the web, or trying to predict behavior of an evolving pathogen, the iid assumption fails because the distribution of the data evolves over time. Similarly, there may be significant biases in how the training data is collected versus the distribution of the test data; for instance the volunteers who choose to participate in a study may be statistically different from the larger population for which the study hopes to make predictions. By not cross-validating the neighborhood size or other learning parameters, one avoids overfitting to the training distribution in such cases. In contrast to cross-validating one neighborhood size for a learning problem, Bayesian neighborhoods only makes locally iid assumptions.

There have been previous efforts to define neighborhoods for local learning methods that do not require cross-validation. For example, a small set of experiments showed that using the relative-neighborhood-graph neighbors of the test point yields generally lower error than k nearest neighbors for classification [17]. For local linear interpolation and local linear regression, significant error reductions have been achieved with the test point's natural neighbors and the enclosing kNN neighbors [18], [19], which attempt to enclose a test point in the convex hull of its neighbors. Although such spatially adaptive neighborhoods have worked well for low-dimensional learning problems, they tend to be computationally challenging or ill-suited for general classification problems where the training data is not sufficiently dense in the sample space [19].

Another set of related methods are kNN committee classifiers that combine multiple classifiers aiming to alleviate individual weaknesses [5, ch. 8]. The proposed Bayesian neighborhoods falls into this category because it takes an average of the predicted local probabilities (or local discriminants) weighted by a prior probability on neighborhood size. Other researchers have proposed methods that form weighted averages over neighborhoods of kNN classifiers but have focused on learning an appropriate weight for each neighborhood size. Ghosh et al. [20] and Paik and Yang [21] have each proposed weighting schemes that are trained by cross-validation, and although not explicitly shown, they may be applied to other local classifiers as well as kNN. However, these are shown to offer only modest gains in performance over cross-validating a single fixed classifier while offering no savings in computation. Like we do in this work, Holmes and Adams [22] treated the neighborhood size k as a random variable and assume a prior on k. For their Bayesian kNN classifier, they assumed a logistic form for the probability of training sample labels given a k and the training sample feature vectors, and estimated the class posterior for a test sample by marginalizing out uncertainty in k. Their method is completely lazy, as it does not require training, but it cannot be applied to arbitrary local classifiers. Furthermore, it is computationally expensive as it requires Markov chain Monte Carlo sampling to numerically solve the integral needed to classify a test sample.

Other prior work in ensemble methods for kNN have used component classifiers that are not based on different neighborhoods but are not completely lazy. Bay [23] took a committee of kNN classifiers where each acted on a random subset of features. Hall and Samworth [24] have analyzed bagged nearest neighbor classifiers (though at least one empirical study suggests bagging kNN has little effect [25]). Other researchers built component kNN classifiers on different condensed subsets of the training samples [26], [27]. Masip and J. Vitrià [28] considered kNN classifiers on different linear combinations of features and use boosting to find an optimal feature set.

3 Bayesian Neighborhoods

In this section, we introduce Bayesian neighborhoods and show that it minimizes expected misclassification costs for generative classifiers. This is followed by a comparison of the computational complexity of this approach to that of cross-validation.

3.1 Neighborhood Size as a Random Variable

Given the true joint probability distribution of features and labels $p_{X,Y}$ and a prior distribution over class labels P_Y, a test sample $x \in \mathbb{R}^d$ can be assigned the label $y^* \in \{1, 2, \ldots, G\}$ that minimizes the expected misclassification cost (with respect to $p_{X,Y}$). Then y^* solves,

$$\underset{g \in \{1,2,\ldots,G\}}{\arg\min} \sum_{h=1}^{G} C(g,h) P_{Y|X}(h|x), \tag{1}$$

where $C(g,h)$ is the cost of labeling a sample as class g when the true label is class h and we define $P_{Y|X}(h|x) \triangleq P(Y = h \,|\, X = x)$. In practice, it is common to substitute

an estimate of the posterior $\hat{P}_{Y|X}(h|x)$ into (1), where the estimate depends on the training data sample pairs (x_i, y_i), $i \in \{1, 2, \ldots, n\}$, $x_i \in \mathbb{R}^d$, $y_i \in \{1, 2, \ldots, G\}$.

Estimating $\hat{P}_{Y|X}(h|x)$ works best when the training samples and test samples are drawn iid from the same distribution. It is less assumptive to suppose that only a subset of the n training samples are drawn iid from the same class-conditional distribution as the test sample x, and we use the k nearest training samples as an intuitive subset. Let $\hat{P}_{Y|X,K}(h|x,k)$ denote the local posterior distribution estimated from the k nearest neighbors of x. Then we treat the neighborhood size as a random variable K and choose the class that minimizes the expected estimated misclassification costs, where the expectation is with respect to the random K:

$$\begin{aligned} &\underset{g\in\{1,2,\ldots,G\}}{\arg\min}\ E_K\left[\sum_{h=1}^{G} C(g,h)\hat{P}_{Y|X,K}(h|x,K)\right] \\ \equiv\ &\underset{g\in\{1,2,\ldots,G\}}{\arg\min}\ \sum_{h=1}^{G} C(g,h)E_K\left[\hat{P}_{Y|X,K}(h|x,K)\right]. \quad (2) \end{aligned}$$

We refer to this approach as using *Bayesian neighborhoods* because the estimated posterior distribution in (2) is the Bayesian estimate that minimizes any expected Bregman divergence with respect to uncertainty in the neighborhood size (for more on Bayesian estimation, see for example [29, ch. 4]). That is, the probability mass function over classes $E_K\left[\hat{P}_{Y|X,K}(h|x,K)\right]$ that appears in (2) solves

$$\underset{r\in[0,1]^G,\ \sum_{h=1}^{G} r(h)=1}{\arg\min}\ E_K\left[L\left(r, \hat{P}_{Y|X,K}(h|x,K)\right)\right],$$

where L is any Bregman divergence (such as squared ℓ_2 distance). This is a special case of the general result that the expectation minimizes the expected Bregman divergence [30, Theorem 1], [14].

Treating the neighborhood size K as a random variable requires a prior distribution P_K over possible neighborhood sizes, so that the average posterior can be calculated as a simple weighted average of different local posteriors:

$$E_K\left[\hat{P}_{Y|X,K}(h|x,K)\right] = \sum_k P_K(k)\hat{P}_{Y|X,K}(h|x,k).$$

We note that setting the prior P_K is similar to the problem of specifying the set of possible parameter choices for cross-validation but the impact is subtly different. In cross-validation, if one offers too many or too-closely spaced options for k, there is a risk of overfitting to the training data while with Bayesian neighborhoods, placing a heavy prior on large values of k risks bias.

In general, for Bayesian estimation, it is well-known that the choice of prior is extremely important. We designed a prior P_K based on our past experiences of reasonable neighborhood sizes for cross-validated local learning methods [3], [19], [31], [32], and on the following design goals: First, we desire a simple formula for P_K that could be expected to work well with any local classifier. Second, we hope to lessen bias by not placing too much prior probability on large k. Third, we suggest using only a subset of k for computational efficiency. Fourth, because of curse-of-dimensionality issues, we believe the number of nearest-neighbors should generally be larger if there are more feature dimensions. Fifth, we enable satisfaction of Stone's conditions for consistency [33] by ensuring that the largest k in the support of P_K grows as $n \to \infty$, but grows relatively slowly (see Section 4.1 for a further discussion of P_K and consistency).

Given these design goals and past experiences with cross-validated local learning, we propose that in the absence of other prior knowledge, P_K be a sampled log-uniform prior over the set $\mathcal{K} = \{2^1, 2^2, \ldots, 2^\gamma\}$, such that $P_K(k) = 1/|\mathcal{K}|$ for $k \in \mathcal{K}$ and $\gamma = \min(\lfloor \log_2(d \log_2 n) \rfloor, \lfloor \log_2 n \rfloor)$. For example, for the Vowel dataset with $n = 528$ training samples and $d = 10$ features, $k \in \{2, 4, 8, 16, 32, 64\}$. Some model-based local classifiers use k neighbors from each class. In those cases, let $\bar{n}$ be the average number of neighbors per class, and let the prior probability on the gth class's neighborhood size k_g be $P_K(k_g) = 1/|\mathcal{K}|$ for $k_g \in \mathcal{K}$ and $\gamma = \min(\lfloor \log_2(d \log_2 \bar{n}) \rfloor, \lfloor \log_2 \bar{n} \rfloor, n_g)$.

Using this sparse log-uniform prior on the neighborhood, a classification of test samples incurs only an additional $O(\log n)$ cost, where n is the size of the training set. This is a significant reduction compared to the added $O(n)$ cost of the *completely* uniform prior, and in trial examples we observed little difference in the classification performance of these two priors.

Many classifiers produce estimates of the likelihood density $\hat{p}_{X|Y}(x|h)$ for each class h. Given the standard decision rule $\hat{y} = \arg\max_h \hat{P}_{Y|X}(h|x)$ with estimated prior $\hat{P}_Y(h)$, the standard decision rule can be written in terms of the likelihood: $\hat{y} = \arg\max_h \hat{p}_{X|Y}(x|h)\hat{P}_Y(h)$ because $p_X(x)$ does not change the decision. However, in the decision rule given by (2), $p_X(x)$ cannot be ignored. By Bayes' rule, (2) can be written,

$$\underset{g\in\{1,2,\ldots,G\}}{\arg\min} \sum_{h=1}^{G} C(g,h)E_K\left[\frac{\hat{p}_{X|Y,K}(x|h,K)\hat{P}_{Y|K}(h|K)}{\hat{p}_{X|K}(x|K)}\right]. \quad (3)$$

Throughout this paper, we assume that class priors are independent of neighborhood size such that $\hat{P}_{Y|K}(h) = \hat{P}_Y(h)$. Furthermore, for classifiers that produce an estimate of the likelihood, we estimate each $p_{X|K}(x|k)$ after estimating the class likelihoods to ensure its role as a normalizer such that (3) becomes

$$\underset{g\in\{1,2,\ldots,G\}}{\arg\min} \sum_{h=1}^{G} C(g,h)\, E_K\left[\frac{\hat{p}_{X|Y,K}(x|h,K)}{\sum_{j=1}^{G} \hat{p}_{X|Y,K}(x|j,K)\frac{\hat{P}_Y(j)}{\hat{P}_Y(h)}}\right]. \quad (4)$$

We additionally assume uniform class priors, $\hat{P}_Y(h) =$

$1/G$ for all h, simplifying (4) to

$$\arg\min_{g\in\{1,2,\ldots,G\}} \sum_{h=1}^{G} C(g,h) E_K \left[\frac{\hat{p}_{X|Y,K}(x|h,K)}{\sum_{j=1}^{G} \hat{p}_{X|Y,K}(x|j,K)} \right]. \quad (5)$$

Note that in the case of unbalanced data, one would expect increased performance by instead estimating the class priors $\hat{P}_Y(h)$ in (4). However, the choice of how to estimate this quantity is subjective and we chose instead to investigate how well one could do with the simplest (albeit pessimistic) approach.

3.2 Computational Cost

When learning from a dynamic set of data, the cost of training a classifier is no longer a one-time expense. To adapt to the evolving statistics of the data, this cost will be incurred periodically through re-training the classifier. Thus, one might benefit from a completely lazy approach that requires no training whatsoever.

Consider a local classifier with Q parameters (other than neighborhood size). Let S_q be the size of the cross-validation set for the qth parameter and let $S = \prod_{q=1}^{Q} S_q$ be the total number of parameter configurations that will be searched using cross-validation. For Bayesian neighborhoods, let these parameters be fixed to default values, as we do in the experiments in this paper, so that the resulting classifiers are completely lazy. Let Ψ be the cost of evaluating the classifier for a single test point under a single configuration of its parameters.

For neighborhood size, assume that P_K has support for only κ neighborhood sizes such that $\sum_k I_{(P_K(k)\neq 0)} = \kappa$ and that the set of neighborhood sizes considered by cross-validation also has κ choices; again this is consistent with our treatment in the experiments.

Given resource constraints, the set of training data cannot be unbounded. As such, we will restrict our attention to a steady-state analysis in which new labeled samples acquired and old labeled samples are discarded at equal rates, yielding an evolving training set of fixed size n. For any fixed time interval, assume that the cross-validated classifier is re-trained R times in this interval. Then, the evaluation of m test samples in this same interval will incur a cost of $(nRS\kappa + m)\Psi$ using cross-validation and $m\kappa\Psi$ using Bayesian neighborhoods. Thus, Bayesian neighborhoods will be more efficient when $nRS > m$. For instance, Bayesian neighborhoods has a computational advantage when the size of the evolving training data or the rate of re-training (or both) are larger than the rate of test evaluations. However, for very high test throughput, cross-validation may be more efficient.

4 Bayesian Neigbhorhoods for Specific Local Classifiers

In the next subsections we consider the effect of using Bayesian neighborhoods with a number of standard and state-of-the-art local classifiers, and propose a local Bayesian quadratic discriminant analysis (QDA) that is also motivated by minimizing expected Bregman loss.

4.1 Bayesian Neigbhorhoods and kNN

The posterior distribution estimate of the kNN classifier is given by

$$\hat{P}_{Y|X,K}(h|x,k) = \frac{1}{k}\sum_{j=1}^{k} I_{(y_j=h)}, \quad (6)$$

where I is the indicator function, and y_j is the label of the jth neighbor of x.

The hth posterior distribution for kNN with Bayesian neighborhoods is given by,

$$\begin{aligned} &E_K[\hat{P}_{Y|X,K}(h|x,K)] \\ &\quad = \sum_{k=1}^{n} P_K(k) \left(\frac{1}{k}\sum_{j=1}^{k} I_{(y_j=h)} \right) \\ &\quad = \sum_{k=1}^{n} P_K(k) \left(\sum_{j=1}^{n} \left(\frac{1}{k} I_{(j\leq k)} \right) I_{(y_j=h)} \right) \\ &\quad = \sum_{j=1}^{n} \left(\sum_{k=1}^{n} \frac{P_K(k)}{k} I_{(j\leq k)} \right) I_{(y_j=h)}. \quad (7) \end{aligned}$$

From (7), one sees that kNN with Bayesian neighborhoods is a weighted nearest neighbor classifier that applies weights that depend on the distance to the test point. For example, if we take a uniform prior $P_K(k) = \frac{1}{M}$ for $k \in \{1,2,\ldots,M\}$, then the weight for the $(M+1)$th nearest neighbor is $w_{M+1} = 0$, $w_M = \frac{1}{M^2}$, $w_{M-1} = \frac{1}{M^2} + \frac{1}{M(M-1)}$, and so on. This effectively creates a weighting function (kernel) that adapts to the spread of the data and decreases with distance (as long as P_K is decreasing faster than linearly in k). This kernel effect also applies to standard weighted kNN classifiers that employ a fixed weighting kernel.

The Bayesian neighborhoods kNN classifier is consistent if the prior P_K produces a weight vector w on the training samples that satisfies Stone's conditions [33]. The conditions can be met by a prior P_K that is non-increasing in k, and that has support on $\{1,\ldots,M(n)\}$ such that $M(n) \to \infty$ as $n \to \infty$ but slowly such that $M(n)/n \to 0$. Lastly, P_K must have high enough entropy that $\max_j(w_j) \to 0$ as $M(n) \to \infty$. The proposed sampled log-uniform prior described in Section 2 meets Stone's conditions, and thus forms a consistent classifier when used with kNN.

4.2 Locally Linear Classifiers

Locally linear classifiers fit a hyperplane to the neighboring training samples, then classify the test point based on the resulting linear discriminant(s). Two state-of-the-art locally linear classifiers are SVM-KNN [2] and local ridge regression. The SVM-KNN applies a linear kernel SVM to the test samples' k nearest neighbors, and has a regularization parameter C; both k and C are recommended to be chosen by cross-validation [2]. Though a classical regression technique, the use of local linear regression for classification dates back as far as

1977 [33]. In our experiments, we use least-squares fits with a ridge regularization penalty on the hyperplane slope coefficients and a fixed regularization parameter $\kappa = 1$ [34], [5], ensuring numerical stability.

Let $f_{k,g}(x)$ be a local discriminant for class g learned from the k nearest neighbors of x. Then for the Bayesian neighborhood approach we classify based on the expected discriminants $\{E_K[f_{K,g}(x)]\}$ for $g = 1, \ldots, G$. Like kNN, using a locally linear classifier with a Bayesian neighborhood results in the nearer-samples having a greater contribution. However, because the contributions can be positive, zero, or negative, the precise effect of using Bayesian neighborhoods is less predictable for these methods than for kNN.

4.3 Locally Gaussian Classifiers

A standard approach to classification is to assume that each class-conditional distribution is Gaussian, fit the class-conditional distribution to the training data, then classify a test point by choosing the class whose model maximizes the posterior probability of the test point [5]. Depending on whether each class covariance is estimated separately or not, this is called linear discriminant analysis (Fisher discriminant analysis) or quadratic discriminant analysis (QDA). However, these classifiers can perform poorly due to the large model bias of assuming that the class-conditional is modeled by only one Gaussian. We contend that modeling each class-conditional distribution instead as only locally Gaussian could significantly reduce the model bias.

In the first two subsections, we show that two recent local classifiers can be interpreted as locally modeling each class posterior as Gaussian. Then, in Section 4.3.3 we propose to classify by explicitly modeling each local class-conditional distribution as a Gaussian, and adapt a recent Bayesian estimation approach to do so, forming the local BDA classifer.

In each of the three subsections, we consider how Bayesian neighborhoods affects such locally Gaussian classifiers, with the distinction from the previous section that each Gaussian uses the k nearest neighbors from its class for a total of $k \times G$ neighbors.

4.3.1 Local Nearest Means (Local NM)

The local nearest means classifier calculates the mean of each class in a neighborhood of the test sample, and classifies the test sample depending on which local class mean is nearest [35], [36]. Compared to the standard nearest-means classifier, local nearest-means drastically reduces the potentially large bias inherent in modeling each class as being characterized by its mean feature vector. Compared to standard kNN, the classification variance due to outlying samples is reduced. Local nearest-means can be equivalently expressed as a generative classifier where each class is locally modeled as being drawn from a Gaussian distribution with identity covariance. Experimentally, we found that local nearest means generally improves on standard k-NN, but can have a significant model bias problem that keeps it from being competitive with state-of-the-art local classifiers.

4.3.2 K-local Hyperplane Distance Nearest Neighbor Algorithm

Vincent and Bengio [15] proposed a local classifier they termed k-local hyperplane distance nearest neighbor algorithm (HKNN), which we show is equivalent to a local Mahalanobis nearest-means classifier, that is, it is approximately a locally Gaussian model. They motivated HKNN as classifying a test point x by drawing k nearest neighbors from each class, projecting x to the linear span of each set of k points (the affine hull) and choosing the class with minimal projection distance. Given the standard assumption that the training samples are in general position, such a classifier would be indeterminant if k_h exceeds the dimensionality of the data for each h, because each class's neighbors' affine hull would span the entire feature space. They mitigate this problem by regularizing the projection weights, such that the HKNN classification rule is to classify x as the class h that has minimum discriminant $d_k(x, \mu_h)$, where

$$d_k(x, \mu_h)^2 = \min_{\alpha \in \mathbb{R}^k} \|(x - \mu_h) - X_h \alpha\|_2^2 + \lambda \|\alpha\|_2^2, \quad (8)$$

where X_h is a $d \times k$ matrix of the k nearest training samples of class h demeaned by the class mean μ_h, and the regularization parameter λ is trained by cross-validation.

Next, we show that HKNN is equivalent to a local Mahalanobis-distance nearest-means classifier, which is the same as a local regularized QDA classifier except for the class-uncertainty penalty terms:

Proposition 1:
The HKNN class h discriminant (8) can be equivalently expressed as,

$$d_k(x, \mu_h)^2 = (x - \mu_h)^T \left(I + \lambda^{-1} X_h {X_h}^T\right)^{-1} (x - \mu_h). \quad (9)$$

The proof is given in the appendix.

From (9), one sees that the HKNN discriminant is the log-likelihood of x given a Gaussian distribution with regularized covariance $I + \lambda^{-1} X_h X_h^T$, just as it appears in regularized QDA [37]. However, the HKNN discriminant does not include the Gaussian's normalization term, which would add an additional factor of $\ln |I + \lambda^{-1} X_h {X_h}^T|$. In regularized QDA, this normalization term penalizes classes that are less predictable in terms of their preferred feature vectors.

4.3.3 Local Bayesian QDA (Local BDA)

Motivated by the state-of-the-art performance of HKNN [15] and our idea of modeling class-conditional distributions as locally Gaussian, we propose to explicitly model the class-conditionals as locally Gaussian and use a recent data-dependent Bayesian approach [13] to

TABLE 1
Closed-form likelihood for proposed local BDA classifier.

$$\begin{aligned}\hat{p}_{X|Y,K}(x|h,k) &= E_{N_{h,k}}[N_{h,k}(x)] \\ &= \frac{\left(\frac{2k}{k+1}\right)^{\frac{d}{2}}\Gamma\left(\frac{k+d+4}{2}\right)\left|\sum_{i=1}^{n}(x_i-\bar{x}_h)(x_i-\bar{x}_h)^T I_{(y_i=h)}+B_h\right|^{\frac{k+d+3}{2}}}{\Gamma\left(\frac{k+4}{2}\right)\left|\sum_{i=1}^{n}(x_i-\bar{x}_h)(x_i-\bar{x}_h)^T I_{(y_i=h)}+\frac{k(x-\bar{x}_h)(x-\bar{x}_h)^T}{k+1}+B_h\right|^{\frac{k+d+4}{2}}},\end{aligned} \tag{10}$$

estimate the class-conditional distributions from the local training data. We term the resulting classifier *local BDA*.

For local BDA, the estimated class-conditional likelihood is the expectation over all possible Gaussian models: $\hat{p}_{X|Y,K}(x|h,k) = E_{N_{h,k}}[N_{h,k}(x)]$, where the $N_{h,k}$ are independent random Gaussians drawn from $p_{N_{h,k}|\mathcal{T}_x(h,k)}$, and $\mathcal{T}_x(h,k)$ are the k training sample pairs from class h nearest to the test sample x. As in HKNN and nearest means, applying QDA locally reduces its model bias. But here we have the added advantage of estimating the Gaussians with a data-dependent Bayesian approach that reduces the estimation variance as well.

Bayesian QDA classifiers were first proposed in the 1960's [38], [39], but were not found to perform well; in particular, they were found to have too much bias [40]. Recently, Srivastava et al. [13] demonstrated that using a data-dependent prior and the Fisher information measure leads to a Bayesian QDA classifier they termed BDA. It was shown to perform as well or better than other state-of-the-art approaches to QDA, including regularized QDA [37] and eigenvalue-decomposition discriminant analysis [41]. It has been shown that the mean Gaussian with respect to the posterior minimizes the expected functional Bregman risk between the estimated pdf and the possible Gaussians that could have generated the training samples [13], [14]. This data-dependent Bayesian QDA classifier requires cross-validating hyperparameters of the inverted Wishart prior: a scale parameter q and a seed matrix B_h for the hth class for $h = 1, \ldots, G$. For our local BDA classifier, we fix q and each B_h without cross-validation. First, we set $q = d+3$, which makes the prior the standard inverted gamma distribution if $d = 1$. Like Srivastava et al. [13], we note that B_h/q is the maximum of the inverted Wishart prior, and that by setting B_h/q to be a rough estimate of the covariance we can form a data-dependent prior that is tuned to the scale of the data. Specifically, we would like to use the diagonal of the maximum likelihood estimate as a rough estimate of the covariance, so that $B_h/q = \text{diag}\left(\hat{\Sigma}_{ML,h}\right)$, but for $k < d$ this could leave B_h ill-posed and uninvertible. To avoid that possibility and regularize the choice of the maximum of the prior, we set

$$B_h = (1-\lambda)q\text{diag}\left(\hat{\Sigma}_{ML,h}\right) + \lambda I,$$

where $\lambda \in [0,1]$ and we assume that each feature is standard-normalized (as is standard) in each dimension before classification. We contend that the exact choice of λ should matter little, though preferably be small as its role is simply to ensure that B_h is well-posed. For our experiments we chose $\lambda = .05$ without preliminary experiments (so as not to be tempted to overfit the choice of λ to these particular datasets). After running the experiments we analyzed whether a different small choice of λ would have mattered. As we expected, we found that classification performance was quite robust to the choice of λ. For example, for the Vowel dataset and $\lambda \in \{.0001, .001, .01, .02, .03, \ldots, .1\}$, the classification performance on the standard train/test partitions only ranged from 43.7 to 45.0 when the neighborhood size was selected by cross-validation, and only ranged from 33.6% to 34.6% when local BDA was combined with Bayesian neighborhoods (see Section 4 for complete experimental details). On the randomized train/test partitions, the Vowel classification performance for the same test range of λ ranged from 5.9% to 6.2% with cross-validation and from 6.0% to 6.1% with Bayesian neighborhoods.

Following from [13, Theorem 1], the proposed local BDA classifier estimates the hth local class-conditional likelihood as given in Table 1, where $\Gamma(\cdot)$ is the standard gamma function, and $\bar{x}_h$ is the average of the k nearest training feature vectors from class h.

Combined with the proposed Bayesian neighborhoods, local BDA produces the decision rule:

$$\underset{g\in\{1,2,\ldots,G\}}{\arg\min}\sum_{h=1}^{G} C(g,h)E_K\left[\frac{E_{N_{h,K}}[N_{h,K}(x)]}{\sum_{j=1}^{G}E_{N_{j,K}}[N_{j,K}(x)]}\right].$$

The estimated class-conditional distributions $E_{N_{h,k}}[N_{h,k}(x)]$ given in (10) are not Gaussians, but in fact the local BDA decision boundary is locally quadratic:

Proposition 2:
For fixed k, the local BDA decision boundary is piecewise quadratic. The proof is in the appendix.

4.3.4 Discriminant Adaptive Nearest Neighbors

The discriminant adaptive nearest neighbors (DANN) method [16] also uses a locally Gaussian assumption. DANN models each local class likelihood as a Gaussian with the same covariance matrix. Then, DANN uses the

Gaussian assumption to imply a local distance metric, which is then used to find the k nearest neighbors to the test point x. It is dissimilar to the previous three algorithms in that the final classifier used is a weighted kNN classifer. The Gaussian assumption is used merely to adapt the metric, and not to classify x.

5 Experiments and Results

In the introduction we discussed some potential shortcomings of cross-validation. In this section we provide experimental results to show that the Bayesian neighborhoods with the proposed log-sampled prior works as well or better in practice as cross-validation, and does so without training. Experiments are performed using seven datasets with standard training/test partitions. Six of these datasets are from the UCI machine learning repository (http://www.ics.uci.edu): Vowel, Image Segmentation, Optical Digits, Letter Recognition, Pen Digits and Isolet. The last is the USPS dataset available from http://www.kernel-machines.org/data. Details for all of the datasets are given in Table 2.

First we motivate using Bayesian neighborhoods rather than cross-validation on a case study of the Vowel dataset. Then we compare kNN and the proposed Bayesian neighborhood on six classifiers. The experimental details are given in Sec. 5.2, followed by results using standard train/test partitions in Sec. 5.3 and random train/test partitions in Sec. 5.4.

TABLE 2
Information About the Benchmark datasets

	# of Classes	# of Features	Total Samples	# in Standard Train/Test
Vowel	11	10	990	528/462
Image Seg.	7	19	2,310	210/2,100
Opt. Digits	10	64	5,620	3,823/1,797
Letter Rec.	26	16	20,000	16,000/4,000
Pen Digits	10	16	10,992	7,494/3,498
USPS	10	256	9,298	7,291/2,007
Isolet	26	617	7,797	6,238/1,559

5.1 A Case Study of the Vowel Data Set

In this section we aim to explore the effects of cross-validation in detail by focusing on the Vowel dataset (see Table 2). This dataset is of practical interest because the training and test sets consist of vocalizations generated by two separate sets of individuals, thus the training and test samples are not identically distributed.

5.1.1 Classification over k

To demonstrate that classification using one fixed value for k might be sub-optimal, Fig. 1 shows how 45 randomly chosen test samples from the standard test set of Vowel are classified using the HKNN classifier for neighborhood sizes $k = 2, 3, \ldots, 32$. In this figure, white indicates a correct classification and black indicates an incorrect classification. One sees that for many test points the classification is quite sensitive to the choice of k. There is no range of values for k which produce correct classifications consistently across the samples (i.e. no broad white columns); we found that this aspect of Fig. 1 is representative, regardless of the classifier used.

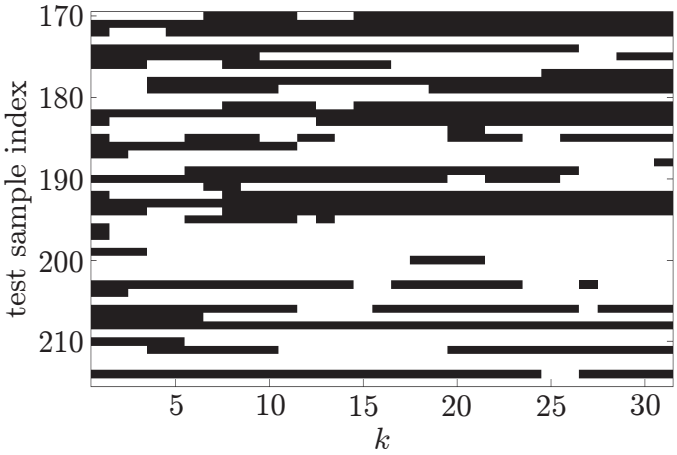

Fig. 1. Rows are test samples; columns are neighborhood sizes. White indicates a correct classification and black indicates a misclassification using HKNN

5.1.2 Discrimination over k

Looking a bit closer, we examine the discriminant values (posteriors for probabilistic classifiers) for two test samples on a subset of the classifiers investigated. The test samples were chosen to be representative of easy to classify and hard to classify test samples, respectively. In Figures 2 and 3 we plot the eleven class discriminants for four local classifiers, with the discriminant of the correct class marked in bold. Note that for a fixed value of k, the cross-validated classifier chooses the class corresponding to the largest discriminant. Depending on the classifier, k either denotes the total number of neighbors (kNN, SVM-KNN), or the number of neighbors from each class (local BDA and HKNN).

One can see that the classifiers each have a different sensitivity to the choice of neighborhood. While one cannot extrapolate from the two test samples shown here, we found over a larger set that the most sensitive classifier was indeed SVM-KNN, as reflected in these plots. In general, the discriminant values appear to be a "noisy" function of k, which motivates averaging over the discriminant values as done with Bayesian neighborhoods. Note that although in this example the discriminants are being plotted for a dense range of k values, in practice, the discriminants are computed only for a sparse sampling of k. For our experiments there were never more than 12 values of k for any dataset.

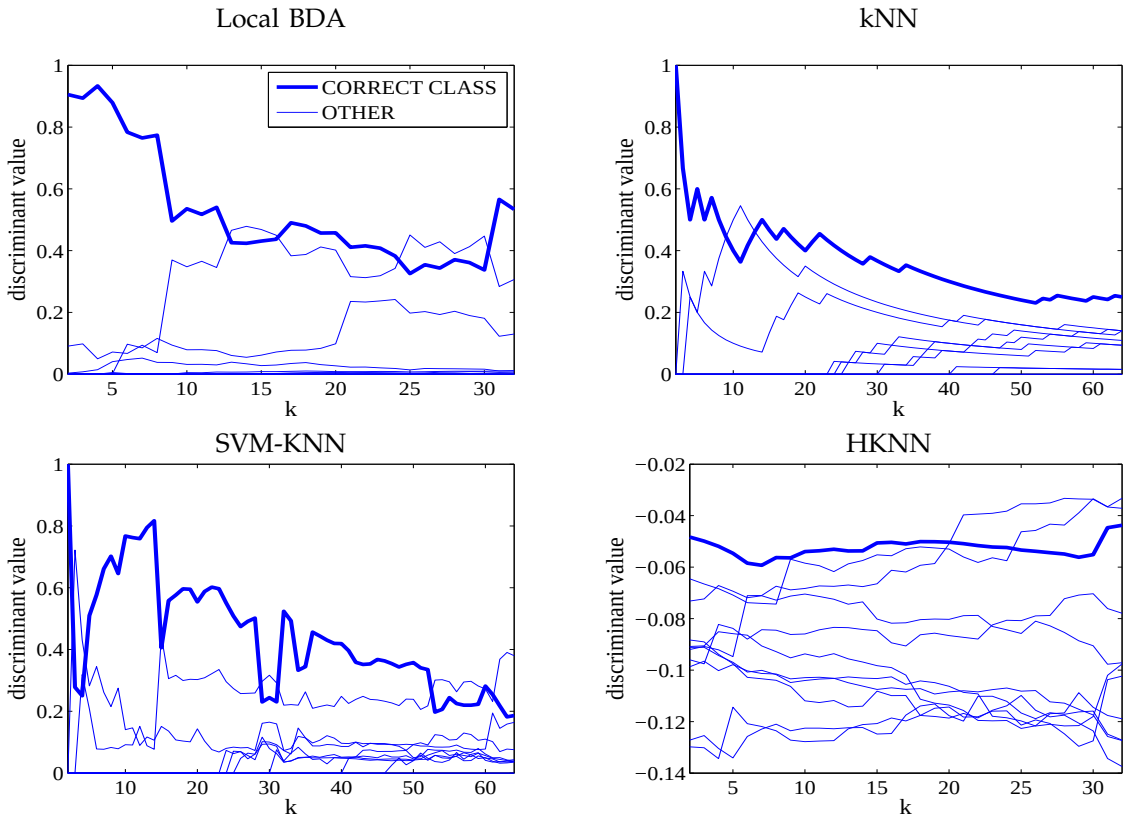

Fig. 2. Discriminant vs neighborhood size for example test point 284 from Vowel.

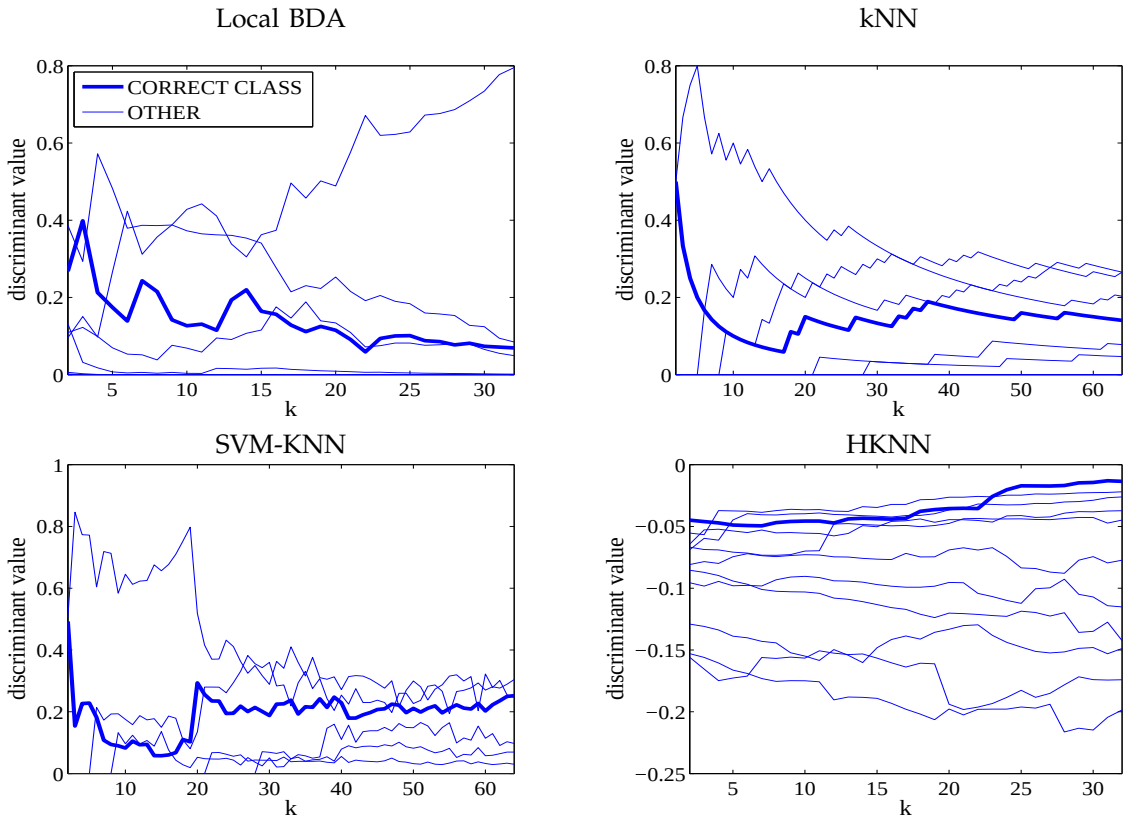

Fig. 3. Discriminant vs neighborhood size for example test point 55 from Vowel.

5.1.3 Letting cross-validation "cheat"

Next, we performed an experiment using the standard training/test partitions for the Vowel dataset. Here we compare the performance of Bayesian neighborhoods to cross-validation while allowing the cross-validated classifier to train on the test data. It is important to note that here each cross-validated classifier chooses the fixed k that achieves the lowest possible error on the test set (not possible in practice). The results are shown in Table 3.

TABLE 3
% test error for Vowel using the standard test set

	Local BDA	HKNN	KNN	DANN	Pinv	SVM-KNN
BN	**34.0**	40.3	48.1	39.8	42.6	44.2
CV on test	35.9	**39.0**	48.1	**38.5**	**38.5**	**37.7**

We used the sampled log-uniform prior proposed in section 3 for the Bayesian neighborhood to generate $k \in$ {2,4,8,16,32} neighbors from each class for HKNN and local BDA, and $k \in$ {2,4,8,16,32,64} total neighbors for kNN, DANN, local ridge, and SVM-KNN. The cross-validated k were chosen from $k \in$ {2,3,..., 32} neighbors from each class for HKNN, and local BDA, and $k \in$ {2,3,..., 64} total neighbors for kNN, DANN, local ridge, and SVM-KNN. Table 3 shows that the Bayesian neighborhood for Vowel actually performs better than the best possible fixed k for local BDA, and kNN, and only slightly worse for HKNN and DANN. Thus even when cross-validation is allowed to "cheat" by cross-validating on the test data, Bayesian neighborhoods can be competitive.

5.2 Training/Test Experimental Details

We performed experiments comparing cross-validation and the proposed Bayesian neighborhoods on seven standard benchmark datasets. Each dataset was used twice, once with standard train/test partitions for reproducibility and once with random train/test partitions to evaluate statistical significance. These datasets were chosen because all features are real-valued, there are no missing data, and there are standard training/test partitions. For each standard/random partition, the training data were normalized to have a mean of 0 and standard deviation of 1, and the test data were then normalized by the same values. Features constant over all training samples were removed.[1]

For the experiments on random partitions detailed in Sec. 5.4, the test and training data were combined, and then ten random splits were made independently and identically to form ten new randomized 50-50 splits of training and test. Due to the large size/dimensionality of the USPS and Isolet datasets, they were not included in the randomized-partition experiments.

All classifier parameters were set by 10-fold cross-validation and in the case of cross-validation error ties, the smallest tied-parameter was chosen. The HKNN algorithm requires a regularization parameter λ as well as a neighborhood size k. For the cross-validation runs, the HKNN λ was cross-validated as $\lambda \in \{1, 5, 10, 20, 30, 50\}$ as recommended by the HKNN authors. To form a completely lazy learning approach using the Bayesian neighborhood, we fixed the HKNN regularization parameter to be $\lambda = 1$, the minimal amount of regularization recommended by the HKNN authors (a better approach would probably be to set λ to be a decreasing function of k such that more regularization is used with lower k, but the results show that even the naive non-k-adaptive choice of $\lambda = 1$ works well).

The SVM-KNN algorithm also requires a regularization parameter C in addition to the neighborhood size k. For the cross-validation runs, the SVM-KNN C was cross-validated as $C \in \{.001, .01, .1, 1, 10, 100, 1000\}$, based on standard practice. To form a completely lazy learning approach using the Bayesian neighborhood, we fixed the SVM-KNN regularization parameter to be the default value of $C = 1$ (as with HKNN, a better approach would probably be to regularize harder for lower k, but the results show that even the naive non-k-adaptive choice of $C = 1$ works well). [2] For multi-class datasets we implemented $\binom{n}{2}$ one-against-one classifiers.

In all cases, the support of the prior P_K and the set of choices of k for cross-validation were the same. For kNN, DANN, SVM-KNN, and local ridge: $k \in \{2^1, \ldots, 2^\gamma\}$ with $\gamma = \min(\lfloor \log_2(d \log_2 n) \rfloor, \lfloor \log_2 n \rfloor)$, where n denotes the number of training samples available in the cross-validation (for 10-fold cross-validation, n is 90% of the total training samples). For HKNN, and local BDA, and for each class g, we set

$$\gamma(g) = \min(\lfloor \log_2(d \log_2 \bar{n}) \rfloor, \lfloor \log_2 \bar{n} \rfloor, n_g),$$

where $\bar{n}$ denotes the average number of training samples for each class available in the cross-validation and n_g is the number of training samples available in the cross-validation for that class. The prior P_K was taken as in Section 2 to be uniform over the sampled log-uniform set of possible values for k.

Throughout the experiments, we assume the class prior probabilities P_Y are uniform over the set of possible classes.

5.3 Standard-Partition Benchmark Data Set Results

Table 4 shows the classification errors using the standard partitions of the benchmark datasets. For each dataset, the best performance is marked in bold, and for six of the seven datasets this is achieved with the Bayesian neighborhood method. We note that some of the algorithms

1. MATLAB code is available at http://idl.ee.washington.edu.

2. SVM was implemented with LIBSVM [42].

experience a drastic reduction in error with Bayesian neighborhoods, for example the 34% error achieved with local BDA on the Vowel dataset is, to the best of our knowledge, the lowest recorded error for this dataset, and a 23% improvement over cross-validation. Over the seven datasets, the local BDA score is only worse than cross-validation on PenDigits, and by less than 5%. Similarly, local ridge error increases slightly with Bayesian neighborhoods on Vowel, ImageSeg and USPS, but decreases by 32% for OptDigits, 33% for LetterRec, 19% for PenDigits, and 32% for Isolet.

The best performances are achieved by Bayesian neighborhoods for six of the seven datasets, three times by local BDA and three times by local ridge. Over the datasets, local BDA performs consistently well and has the lowest total error with Bayesian neighborhoods, followed by cross-validated local BDA. Local ridge, SVM-KNN, and local HKNN are the next top performers, and each has total lower error with Bayesian neighborhoods than with cross-validation. This is surprising because to make the Bayesian neighborhoods completely lazy, we used fixed regularization parameters for HKNN ($\lambda = 1$) and SVM-KNN ($C = 1$) for the results in the column marked BN (and cross-validated both parameters for the cross-validation option). Table 6 shows the chosen cross-validation parameters.

TABLE 6
Cross-validated Parameter Choices Given Standard Training/Test Partitions

	Vowel	Image Seg	Opt Digits	Letter Rec	Pen Digits	USPS	Isolet
kNN k	2	4	4	4	2	4	16
DANN k	2	2	4	4	2	4	4
Local BDA k	2	8	16	8	32	32	256
HKNN k	8	4	64	8	4	32	256
HKNN λ	1	1	50	1	1	50	50
Local ridge k	8	8	256	8	32	32	128
SVM-KNN k	2	128	128	64	32	256	256
SVM-KNN C	0.001	10	0.1	10	10	0.01	0.1

5.4 Random-Partition Benchmark Data Set Results

Table 5 shows the mean misclassification rate averaged over the 10 randomized train/test splits. For each dataset the lowest mean score is in bold, as well as any results for which the lowest mean score classifier was not statistically significantly better, according to one-sided Wilcoxon nonparametric signed rank tests with a significance value of $p = .05$. With these iid partitions and averaged over ten randomizations, one sees less dramatic differences between the cross-validation and Bayesian neighborhood error rates. Comparing the different local classifiers, local BDA again performs consistently well and achieves the lowest total average error for Table 5 with Bayesian neighborhoods. Cross-validated local BDA and HKNN are the second best performers in terms of total average error, with SVM-KNN and local ridge the next top performers.

Table 7 shows for each dataset and classifier whether the classification results were statistically significantly better using cross-validation or Bayesian neighborhoods according to one-sided Wilcoxon nonparametric signed rank tests with a significance value of $p = .05$; the mark $-$ denotes that neither was significantly better than the other. The results vary by algorithm. Again, we are surprised that not cross-validating the regularization parameters for HKNN and SVM-KNN for Bayesian neighborhoods seems to have only a small impact.

TABLE 7
Statistically Significantly Better Performance: Bayesian Neighborhoods vs. Cross-validation

	kNN	DANN	Local BDA	HKNN	Local Ridge	SVM-KNN
Vowel	CV	CV	-	BN	-	CV
Image Seg.	CV	-	-	BN	BN	CV
Opt. Digits	-	CV	-	CV	BN	-
Letter Rec.	BN	BN	BN	CV	BN	-
Pen Digits	CV	CV	-	BN	-	-

6 Conclusions and Open Questions

In this paper, we proposed a simple averaging alternative to neighborhood selection for local classifiers that is optimal in the sense that the recovered posterior minimizes the expected Bregman divergence to the true posterior distribution. We showed that this Bayesian neighborhoods approach achieves error rates that are similar to those given by a cross-validated neighborhood size with six different local classifiers, but without the pre-processing. For learning problems with large or evolving training sets, this can offer significant computational savings. Coupling Bayesian neighborhoods with the proposed local BDA classifier takes expectations with respect to both the uncertain posterior and neighborhood size, and performed strongly across the set of experiments compared to the six other local classifiers. The next best performance overall is given by another closed-form classifier, the local ridge classifier.

While not suitable for all applications, lazy learning is an effective approach for a wide variety of tasks, in particular those characterized by an evolving training distribution where frequent re-training is impractical, and those tasks where the training set is too large to train global classifiers. More generally, because lazy learning makes strictly looser assumptions than globally-trained classifiers about training and test samples being iid, we hypothesize that effective completely lazy learning methods can perform better than globally-trained classifiers for applications where the iid assumption is not valid, though this remains an open question.

TABLE 4
% Test Error Given Standard Training/Test Partitions

	Vowel		ImageSeg		OptDigits		LetterRec		PenDigits		USPS		Isolet	
	CV	BN	CV	BN	CV	BN	CV	BN	CV	BN	CV	BN	CV	BN
kNN	52.4	48.1	12.9	12.1	3.5	3.5	5.3	5.2	2.7	3.1	5.7	7.6	8.7	6.9
DANN	39.0	39.8	7.5	7.5	4.0	4.3	5.1	4.6	2.8	3.1	8.2	8.1	8.6	8.4
Local BDA	44.2	**34.0**	6.9	**6.3**	2.2	1.9	3.0	2.9	2.1	2.2	5.4	5.4	3.3	**3.1**
HKNN	43.9	40.3	9.4	9.1	2.5	2.9	4.2	4.4	2.3	2.3	**4.6**	5.9	4.9	3.7
Local ridge	40.9	42.6	8.2	8.4	2.5	**1.7**	4.2	**2.8**	2.1	**1.7**	5.4	6.2	7.4	5.0
SVM-KNN	49.4	44.2	7.9	9.7	2.3	2.2	3.9	3.4	2.1	2.1	4.7	**4.6**	3.7	**3.1**

Bold: Best result on the dataset.

TABLE 5
% Test Error Averaged Over 10 Random Training/Test Partitions

	Vowel		ImageSeg		OptDigits		LetterRec		PenDigits	
	CV	BN	CV	BN	CV	BN	CV	BN	CV	BN
kNN	12.2	13.4	6.1	6.8	2.9	3.0	7.6	6.9	0.9	1.4
DANN	8.5	9.3	4.3	4.2	2.3	2.6	6.7	6.3	1.2	1.5
Local BDA	5.9	6.0	**4.0**	**3.8**	**1.4**	**1.3**	4.1	4.0	0.6	0.6
HKNN	5.1	**4.3**	5.1	4.2	1.5	1.7	5.0	5.4	0.6	**0.4**
Local ridge	7.3	6.5	4.6	**4.1**	1.8	**1.4**	5.8	**3.8**	**0.5**	**0.5**
SVM-KNN	6.2	8.2	**3.8**	4.6	**1.3**	**1.4**	4.9	4.9	0.6	0.6

Bold: Best mean result for each dataset, and results that are not statistically significantly worse.

Although we used a sampled log-uniform prior P_K that we believe is a reasonable choice given no other information, prior probabilities can have a strong effect on Bayesian estimation, and how to choose an optimal or effective data-dependent prior over the neighborhood sizes is an open question.

APPENDIX

Proof of Proposition 1:
For notational simplicity, we denote X_h and μ_h by X and μ in this proof. The α which solves the minimization in (8) has the closed-form solution,

$$\alpha = (X^T X + \lambda I)^{-1} X^T (x - \mu).$$

With this α, the HKNN discriminant becomes,

$$\begin{aligned} d(x,\mu)^2 &= \|(x-\mu) - X(X^TX+\lambda I)^{-1}X^T(x-\mu)\|_2^2 \\ &\quad +\lambda\|(X^TX+\lambda I)^{-1}X^T(x-\mu)\|_2^2 \\ &= \|(I - X(X^TX+\lambda I)^{-1}X^T)(x-\mu)\|_2^2 \\ &\quad +\lambda\|(X^TX+\lambda I)^{-1}X^T(x-\mu)\|_2^2 \\ &\overset{(a)}{=} \|\lambda(\lambda I + XX^T)^{-1}(x-\mu)\|_2^2 \\ &\quad +\lambda\|X^T(\lambda I+XX^T)^{-1}(x-\mu)\|_2^2 \\ &= \lambda(x-\mu)^T(\lambda I+XX^T)^{-1} \\ &\quad (\lambda I+XX^T)(\lambda I+XX^T)^{-1}(x-\mu) \\ &= \lambda(x-\mu)^T\left(\lambda I+XX^T\right)^{-1}(x-\mu), \end{aligned}$$

where (a) follows by the matrix identities $I - A(I+BA)^{-1}B = (I+AB)^{-1}$ and $(I+AB)^{-1}A = A(I+BA)^{-1}$[43].

Proof of Proposition 2:
The decision boundary between class 1 and 2 is defined by the set of x such that $E_{N_{1,k}}[N_{1,k}(x)] = E_{N_{2,k}}[N_{2,k}(x)]$. From (10), without loss of generality the decision boundary between class 1 and 2 is,

$$\begin{aligned} &[1 + \frac{k}{k+1}(x-\bar{x}_1)^T D_1^{-1}(x-\bar{x}_1)]^{\frac{k+d+4}{2}} \\ &= \gamma_{db}[1 + \frac{k}{k+1}(x-\bar{x}_2)^T D_2^{-1}(x-\bar{x}_2)]^{\frac{k+d+4}{2}}, \qquad (11) \\ &\text{where } D_{h,k} = B_{h,k} + \sum_{i=1}^{k}(x_i-\bar{x}_h)(x_i-\bar{x}_h)^T I_{(y_i=h)}, \end{aligned}$$

and γ_{db} is a constant that depends on the training samples and the number of training samples, but does not depend on the test sample x. Because the exponentiated terms must always be real and positive, raising both sides of (11) to the power $2/(k+d+4)$, gives the following quadratic decision boundary:

$$\begin{aligned} &\left(1 + \frac{k}{k+1}(x-\bar{x}_1)^T D_1^{-1}(x-\bar{x}_1)\right) \\ &= \tilde{\gamma}_{db}\left(1 + \frac{k}{k+1}(x-\bar{x}_2)^T D_2^{-1}(x-\bar{x}_2)\right), \end{aligned}$$

where $\tilde{\gamma}_{db} = \gamma_{db}^{\frac{2}{k+d+4}}$.

REFERENCES

[1] D. Aha, *Lazy Learning*, Springer, 1997.

[2] H. Zhang, A. C. Berg, M. Maire, and J. Malik, "SVM-KNN: discriminative nearest neighbor classification for visual category recognition," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, vol. 2, pp. 2126–2136, 2006.

[3] M. R. Gupta, R. Gray, and R. Olshen, "Nonparametric supervised learning by linear interpolation with maximum entropy," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 28, no. 5, pp. 766–781, May 2006.

[4] W. Lam, C. Keung, and D. Liu, "Discovering useful concept prototypes for classification based on filtering and abstraction," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 8, pp. 1075–1090, August 2002.

[5] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer-Verlag, New York, 2001.

[6] C. Böhm, S. Berchtold, and D. Keim, "Searching in high-dimensional spaces- index structures for improving the performance of multimedia databases," *ACM Computing Surveys*, vol. 33, no. 3, pp. 322–373, September 2001.

[7] D. Cantone, A. Ferro, A. Pulvirenti, D. Reforgiato, and D. Shasha, "Antipole indexing to support range search and k-nearest neighbor on metric spaces," *IEEE Trans. on Knowledge and Data Engineering*, vol. 17, no. 4, pp. 535–550, 2005.

[8] T. Liu, A. W. Moore, and A. Gray, "New algorithms for efficient high-dimensional nonparametric classification," *Journal Machine Learning Research*, vol. 7, pp. 1135–1158, 2006.

[9] X. Liu and H. Ferhatosmano, "Efficient k-NN search on streaming data series," *Lecture Notes on Computer Science*, vol. 2750, pp. 83–101, 2003.

[10] K. Yi, F. Li, G. Kolios, and D. Srivastava, "Efficient processing of top-k queries in uncertain databases with x-Relations," *IEEE Trans. on Knowledge and Data Engineering*, vol. 20, no. 12, pp. 1669–1682, 2008.

[11] Y. Lifshits and S. Zhang, "Combinatorial algorihtms for nearest neighbors, near-duplicates, and small world design," *Proc. SODA*, 2009.

[12] L. Chen and X. Lian, "Efficient similarity search in nonmetric spaces with local constant embedding," *IEEE Trans. on Knowledge and Data Engineering*, vol. 20, no. 3, pp. 321–336, 2008.

[13] S. Srivastava, M. R. Gupta, and B. A. Frigyik, "Bayesian quadratic discriminant analysis," *Journal of Machine Learning Research*, vol. 8, pp. 1287–1314, 2007.

[14] B. A. Frigyik, S. Srivastava, and M. R. Gupta, "Functional Bregman divergence and Bayesian estimation of distributions," *IEEE Trans. Information Theory*, vol. 54, no. 3, pp. 5130–5139, November 2008.

[15] P. Vincent and Y. Bengio, "K-local hyperplane and convex distance nearest neighbor algorithms," *NIPS*, pp. 985–992, 2001.

[16] T. Hastie and R. Tibshirani, "Discriminative adaptive nearest neighbour classification," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 18, no. 6, pp. 607–615, 1996.

[17] J. S. Sánchez, F. Pla, and F. J. Ferri, "On the use of neighbourhood-based non-parametric classifiers," *Pattern Recognition Letters*, pp. 1179–1186, 1997.

[18] R. Sibson, *Interpreting multivariate data*, chapter : A brief description of natural neighbour interpolation, pp. 21–36, John Wiley, 1981.

[19] M. R. Gupta, E. K. Garcia, and E. M. Chin, "Adaptive local linear regression with application to printer color management," *IEEE Trans. Image Processing*, vol. 17, no. 6, pp. 936–945, 2008.

[20] A. K. Ghosh, P. Chaudhuri, and C. A. Murthy, "On visualization and aggregation of nearest neighbor classifiers," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1592–1602, October 2005.

[21] M. Paik and Y. Yang, "Combining nearest neighbor classifiers versus cross-validation selection," *Statistical Applications in Genetics and Molecular Biology*, vol. 3, no. 1, 2004.

[22] C. C. Holmes and N. M. Adams, "A probabilistic nearest neighbour method for statistical pattern recognition," *Journal Royal Statistical Society B*, vol. 64, pp. 295–306, 2002.

[23] S. Bay, "Combining nearest neighbor classifiers through multiple feature subsets," *Proc. Intl. Conf. Machine Learning (ICML)*, pp. 37–45, 1998.

[24] P. Hall and R. J. Samworth, "Properties of bagged nearest neighbour classifiers," *Journal Royal Statistical Society B*, vol. 67, pp. 363–379, 2005.

[25] T. G. Speed, *Statistical Analysis of Gene Expression Microarray Data*, CRC Press, 2003.

[26] D. B. Skalak, *Prototype Selection for Composite Nearest Neighbor Classification*, Ph.D. thesis, Univ. of Massachusetts, 1996.

[27] E. Alpaydin, "Voting over multiple condensed nearest neighbors," *Artificial Intelligence Research*, , no. 11, pp. 115–132, 1997.

[28] D. Masip and J. Vitrià, "Boosted discriminant projections for nearest neighbor classifiers," *Pattern Recognition*, vol. 39, pp. 164–170, 2006.

[29] E. L. Lehmann and G. Casella, *Theory of Point Estimation*, Springer, New York, 1998.

[30] A. Banerjee, X. Guo, and H. Wang, "On the optimality of conditional expectation as a Bregman predictor," *IEEE Trans. on Information Theory*, vol. 51, no. 7, pp. 2664–2669, 2005.

[31] M. R. Gupta and W. H. Mortensen, "Weighted nearest neighbor classifiers and first-order error," *(in review)*, 2008.

[32] M. R. Gupta, S. Srivastava, and L. Cazzanti, "Minimum expected risk estimation for near-neighbor classification," *UWEE Tech Report Series*, , no. 2006-0006, 2006.

[33] C. J. Stone, "Consistent nonparametric regression," *The Annals of Statistics*, vol. 5, no. 4, pp. 595–645, 1977.

[34] A. E. Hoerl and R. Kennard, "Ridge regression: biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, pp. 55–67, 1970.

[35] Y. Mitani and Y. Hamamoto, "Classifier design based on the use of nearest neighbor samples," *Proc. Intl. Conf. on Pattern Recognition*, pp. 769–772, 2000.

[36] Y. Mitani and Y. Hamamoto, "A local mean-based nonparametric classifier," *Pattern Recognition Letters*, vol. 27, pp. 1151–1159, 2006.

[37] J. H. Friedman, "Regularized discriminant analysis," *Journal American Statistical Association*, vol. 84, no. 405, pp. 165–175, 1989.

[38] S. Geisser, "Posterior odds for multivariate normal distributions," *Journal Royal Society Series B Methodological*, vol. 26, pp. 69–76, 1964.

[39] D. G. Keehn, "A note on learning for Gaussian properties," *IEEE Trans. on Information Theory*, vol. 11, pp. 126–132, 1965.

[40] B. Ripley, *Pattern recognition and neural nets*, Cambridge University Press, Cambridge, 2001.

[41] H. Bensmail and G. Celeux, "Regularized Gaussian discriminant analysis through eigenvalue decomposition," *Journal American Statistical Association*, vol. 91, pp. 1743–1748, 1996.

[42] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001.

[43] K. B. Petersen and M. S. Pedersen, "The matrix cookbook," Tech. Rep., Technical University of Denmark, 2005.