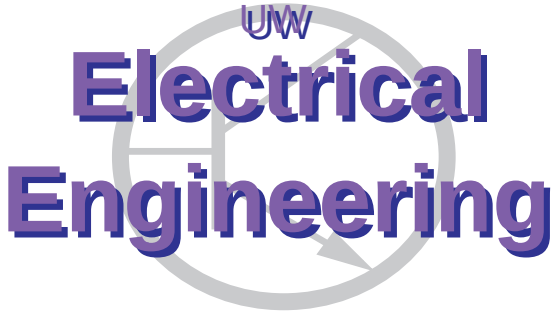
Similarity-based Classification: Concepts and Algorithms

Yihua Chen, Eric K. Garcia, Maya R. Gupta
{yhchen,garciaer,gupta}@ee.washington.edu
Dept. of EE, University of Washington
Seattle, WA 98195

Luca Cazzanti
luca@apl.washington.edu
Applied Physics Lab, University of Washington
Seattle, WA 98195

Ali Rahimi
ali.rahimi@intel.com
Intel Research
Seattle, WA 98195



UWEE Technical Report
Number UWEETR-2008-0005
December 2008

Department of Electrical Engineering
University of Washington
Box 352500
Seattle, Washington 98195-2500
PHN: (206) 543-2150
FAX: (206) 543-3842
URL: http://www.ee.washington.edu

Similarity-based Classification: Concepts and Algorithms

Yihua Chen, Eric K. Garcia, Maya R. Gupta
{yhchen,garciaer,gupta}@ee.washington.edu
Dept. of EE, University of Washington
Seattle, WA 98195

Luca Cazzanti
luca@apl.washington.edu
Applied Physics Lab, University of Washington
Seattle, WA 98195

Ali Rahimi
ali.rahimi@intel.com
Intel Research
Seattle, WA 98195

University of Washington, Dept. of EE, UWEETR-2008-0005
December 2008

Abstract

This report reviews and extends the field of similarity-based classification, presenting new analyses, algorithms, data sets, and the most comprehensive set of experimental results to date. Specifically, the generalizability of using similarities as features is analyzed, design goals and methods for weighting nearest-neighbors for similarity-based learning are proposed, and different methods for consistently converting similarities into kernels are compared. Experiments on eight real data sets compare eight approaches and their variants to similarity-based learning.

1 Introduction

Similarity-based classifiers estimate the class label of a test sample based on the similarities between the test sample and a set of labeled training samples, and also the pairwise similarities between the training samples. Like others, we use the term *similarity-based classification* whether the pairwise relationship is a similarity or dissimilarity. Similarity-based classification does not require direct access to the features of the samples, and thus the sample space can be any set, not necessarily a Euclidean space, as long as the similarity function is well defined for any pair of samples. Let Ω be the sample space and $\mathcal{G}$ be the finite set of class labels. Let $\psi : \Omega \times \Omega \rightarrow \mathbb{R}$ be the similarity function. We assume that the pairwise similarities between n training samples are given as an $n \times n$ similarity matrix S whose (i, j)-entry is $\psi(x_i, x_j)$, where $x_i \in \Omega$, $i = 1, \ldots, n$, denotes the ith training sample, and $y_i \in \mathcal{G}$, $i = 1, \ldots, n$ the corresponding ith class label. The problem is to estimate the class label $\hat{y}$ for a test sample x based on its similarities to the training samples $\psi(x, x_i)$, $i = 1, \ldots, n$ and its self-similarity $\psi(x, x)$.

Similarity-based classification is useful for problems in computer vision, bioinformatics, information retrieval, natural language processing, and a broad range of other fields, especially those involving or trying to imitate human judgement. Similarity functions may not be symmetric or satisfy the other mathematical properties required for metrics or inner products [1]. Some simple example similarity functions are: travel time from one place to another, compressability of one random process given a code built for another, and the minimum number of steps to convert one sequence into another (edit distance). Computer vision researchers use many similarities, such as the tangent distance [2], earth mover's distance (EMD) [3], shape matching distance [4], and pyramid match kernel [5] to measure the

similarity or dissimilarity between images in order to do image retrieval and object recognition. In bioinformatics, the Smith-Waterman algorithm [6], the FASTA algorithm [7] and the BLAST algorithm [8] are popular methods to compute the similarity between different amino acid sequences for protein classification. The cosine similarity between term frequency-inverse document frequency (tf-idf) vectors is widely used in information retrieval and text mining for document classification.

Notions of similarity appear to play a fundamental role in human learning, and thus psychologists have done extensive research to model human similarity judgement. Tversky's *contrast model* and *ratio model* [9] represent an important class of similarity functions. In these two models, each sample is represented by a set of features, and the similarity function is an increasing function of set overlap but a decreasing function of set differences. Tversky's set-theoretic similarity models have been successful in explaining human judgement in various similarity assessment tasks, and are consistent with the observations made by psychologists that metrics do not account for cognitive judgement of similarity in complex situations [9–11]. Therefore, similarity-based classification may be useful for imitating or understanding how humans categorize.

The main contributions of this report are: (1) we distill and analyze concepts and issues specific to similarity-based learning, including the generalizability of using similarities as features, (2) we propose similarity-based nearest-neighbor design goals and methods, and (3) present the largest-to-date set of experimental results for eight similarity-based learning problems and eight different similarity-based classification approaches and their variants. First, we discuss the idea of similarities as inner products in Section 2, then the concept of treating similarities as features in Section 3. In Section 4, we propose design goals and solutions for similarity-based weighted nearest-neighbor learning. Generative similarity-based classifiers are discussed in Section 5. Then in Section 6 we describe eight similarity-based classification problems, detail our experimental set-up, and discuss the results. The paper concludes with some open questions in Section 7. For the reader's reference, key notation is summarized in Table 1.

Table 1: Key Notation

Ω	sample space	S	$n \times n$ matrix with (i,j)-entry $\psi(x_i, x_j)$
$\mathcal{G}$	set of class labels	s_i	$n \times 1$ vector with jth element $\psi(x_i, x_j)$
n	number of training samples	s	$n \times 1$ vector with jth element $\psi(x, x_j)$
$x_i \in \Omega$	ith training sample	$\mathbf{1}$	column vector of 1's
$x \in \Omega$	test sample	I	identity matrix
$y_i \in \mathcal{G}$	class label of ith training sample	I_A	indicator function of event A
$y \in \mathcal{G}^n$	$n \times 1$ vector with ith element y_i	K	kernel matrix or kernel function
$\hat{y} \in \mathcal{G}$	estimated class label for x	k	neighborhood size
$\mathcal{D}$	n training sample pairs $\{(x_i, y_i)\}$	L	hinge loss function
$\psi : \Omega \times \Omega \to \mathbb{R}$	similarity function	$\mathrm{diag}(a)$	diagonal matrix with diagonal given by a

2 Similarities as Inner Products

A popular approach to similarity-based classification is to treat the given similarities as inner products in some Hilbert space or to treat dissimilarities as distances in some Euclidean space. This approach can be roughly divided into two categories: one is to explicitly embed the samples in a Euclidean space according to the given (dis)similarities using multidimensional scaling (see [12] for further reading); the other is to modify the similarities to be kernels and apply kernel classifiers. We discuss different methods for modifying similarities into kernels in Subsection 2.1. An important technicality is how to handle test samples, which is addressed in Subsection 2.2.

2.1 Modify Similarities into Kernels

The power of kernel methods lies in the implicit use of a reproducing kernel Hilbert space (RKHS) induced by a positive semidefinite (PSD) kernel [13]. Although the mathematical meaning of a kernel is the inner product in some Hilbert space, a standard interpretation of a kernel is the pairwise similarity between different samples. Conversely, many researchers have suggested treating similarities as kernels, and applying any classification algorithm that only

depends on inner products. Using similarities as kernels eliminates the need to explicitly embed the samples in a Euclidean space.

Here we focus on the support vector machine (SVM), which is a well-known representative of kernel methods, and thus appears to be a natural approach to similarity-based learning. All the SVM algorithms that we discuss in this paper are for binary classification[1] such that $y_i \in \{\pm 1\}$. Let y be the $n \times 1$ vector whose ith element is y_i. The SVM dual problem can be written as

$$\begin{aligned} \underset{\alpha}{\text{minimize}} \quad & \frac{1}{2}\alpha^T \operatorname{diag}(y) K \operatorname{diag}(y) \alpha - \mathbf{1}^T \alpha \\ \text{subject to} \quad & 0 \preceq \alpha \preceq C\mathbf{1}, \quad y^T \alpha = 0, \end{aligned} \tag{1}$$

with variable $\alpha \in \mathbb{R}^n$, where $C > 0$, K is a PSD kernel matrix whose (i, j)-entry is $K(x_i, x_j)$, $\mathbf{1}$ is the column vector with all entries one, and $\succeq$ denotes component-wise inequality for vectors. The corresponding decision function is [13]

$$\hat{y} = \operatorname{sgn}\left(\sum_{i=1}^{n} \alpha_i y_i K(x, x_i) + b \right),$$

where

$$b = y_i - \sum_{j=1}^{n} \alpha_j y_j K(x_i, x_j)$$

for any i that satisfies $0 < \alpha_i < C$. The theory of RKHS requires the kernel to satisfy Mercer's condition, and thus the corresponding kernel matrix K must be PSD. However, many similarity functions do not satisfy the properties of an inner product, and thus the similarity matrix S can be indefinite. In the following subsections we discuss several methods to modify similarities into kernels; a previous review can be found in [15]. Unless mentioned otherwise, in the following subsections we assume that S is symmetric. If not, we use its symmetric part $\frac{1}{2}\left(S + S^T\right)$ instead. Notice that the symmetrization does not affect the SVM objective function since $\alpha^T \frac{1}{2}\left(S + S^T\right)\alpha = \frac{1}{2}\alpha^T S\alpha + \frac{1}{2}\alpha^T S^T \alpha = \alpha^T S\alpha$.

2.1.1 Indefinite Kernels

One approach is to simply replace K with S, and ignore the fact that S is indefinite. For example, although the SVM problem given by (1) is no longer convex when S is indefinite, it has been shown that the sequential minimal optimization (SMO) [16] algorithm will still converge with a simple modification to the original algorithm [17], but the solution is a stationary point instead of a global minimum. Ong et al. [18] interpret this as finding the stationary point in a reproducing kernel Kreĭn space (RKKS), while Haasdonk [19] shows that this is equivalent to minimizing the distance between reduced convex hulls in a pseudo-Euclidean space. A Kreĭn space, denoted by $\mathcal{K}$, is defined to be the direct sum of two disjoint Hilbert spaces, denoted by $\mathcal{H}_+$ and $\mathcal{H}_-$, respectively. So for any $a, b \in \mathcal{K} = \mathcal{H}_+ \oplus \mathcal{H}_-$, there are unique $a_+, b_+ \in \mathcal{H}_+$ and unique $a_-, b_- \in \mathcal{H}_-$ such that $a = a_+ + a_-$ and $b = b_+ + b_-$. The "inner product" on $\mathcal{K}$ is defined as

$$\langle a, b \rangle_{\mathcal{K}} = \langle a_+, b_+ \rangle_{\mathcal{H}_+} - \langle a_-, b_- \rangle_{\mathcal{H}_-},$$

which no longer has the property of positive definiteness. Pseudo-Euclidean space is a special case of Kreĭn space where $\mathcal{H}_+$ and $\mathcal{H}_-$ are two Euclidean spaces. Ong et al. [18] provide a representer theorem for RKKS that poses learning in RKKS as a problem of finding a stationary point of the risk functional, in contrast to minimizing a risk functional in RKHS. Because this can lead to a saddle point solution and thus does not ensure minimizing the risk functional, this approach does not guarantee learning in the sense of a good function approximation. Also, the nonconvexity of the problem may require intensive computation.

2.1.2 Spectrum Clip

Since S is assumed to be symmetric, it has an eigenvalue decomposition $S = U^T \Lambda U$, where U is an orthogonal matrix and Λ is a diagonal matrix of real eigenvalues, that is, $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$. Spectrum clip makes S PSD by clipping

[1] We refer the reader to [14] for multiclass SVM.

all the negative eigenvalues to zero. Some researchers assume that the negative eigenvalues of the similarity matrix are caused by noise and view spectrum clip as a denoising step [15]. Let

$$\tilde{\Lambda}_{\mathrm{c}} = \mathrm{diag}\left(\max\{\lambda_1, 0\}, \ldots, \max\{\lambda_n, 0\}\right),$$

and the modified PSD similarity matrix be $\tilde{S}_{\mathrm{c}} = U^T \tilde{\Lambda}_{\mathrm{c}} U$. Let u_i denote the ith column vector of U. Using $\tilde{S}_{\mathrm{c}}$ as a kernel matrix for training the SVM is equivalent to implicitly using $x_i = \tilde{\Lambda}_{\mathrm{c}}^{1/2} u_i$ as the representation of the ith training sample since $\langle x_i, x_j \rangle$ is equal to the (i,j)-entry of $\tilde{S}_{\mathrm{c}}$. A mathematical justification for spectrum clip is that $\tilde{S}_{\mathrm{c}}$ is the nearest PSD matrix to S in terms of the Frobenius norm, that is,

$$\tilde{S}_{\mathrm{c}} = \arg\min_{K \succeq 0} \|K - S\|_F,$$

where $\succeq$ denotes the generalized inequality with respect to the PSD cone (see [20] for a proof).

Recently, a robust extension of SVM has been proposed for indefinite kernels [21]. Instead of only considering $\tilde{S}_{\mathrm{c}}$, they consider all the PSD matrices within distance β to S, i.e., $\{K \succeq 0 \,|\, \|K - S\|_F \leq \beta\}$, where $\beta > \min_{K \succeq 0} \|K - S\|_F$, and propose to minimize the worst case of the SVM dual objective among these matrices:

$$\begin{aligned} \underset{\alpha}{\text{minimize}} \quad & \max_{K \succeq 0, \|K - S\|_F \leq \beta} \left(\frac{1}{2} \alpha^T \,\mathrm{diag}(y) K \,\mathrm{diag}(y) \alpha - \mathbf{1}^T \alpha \right) \\ \text{subject to} \quad & 0 \preceq \alpha \preceq C\mathbf{1}, \quad y^T \alpha = 0. \end{aligned}$$

They further replace the hard constraint $\|K - S\|_F \leq \beta$ with a penalty term and propose the following problem as the robust SVM for S:

$$\begin{aligned} \underset{\alpha}{\text{minimize}} \quad & \max_{K \succeq 0} \left(\frac{1}{2} \alpha^T \,\mathrm{diag}(y) K \,\mathrm{diag}(y) \alpha - \mathbf{1}^T \alpha - \rho \|K - S\|_F^2 \right) \\ \text{subject to} \quad & 0 \preceq \alpha \preceq C\mathbf{1}, \quad y^T \alpha = 0, \end{aligned}$$

where $\rho > 0$ is the parameter to control the trade-off. They point out that the inner problem, which, by the trick of Schur complement, can be formulated into a semidefinite program (SDP), has an analytic solution, and that the outer problem is convex because its objective is a pointwise maximum of a set of convex quadratic functions of α and thus convex.

2.1.3 Spectrum Flip

In contrast to the interpretation that negative eigenvalues are caused by noise, it has also been shown that the negative eigenvalues of some similarity data can code useful information about object features or categories [22, 23], which agrees with some fundamental psychological studies [10, 24]. In order to use the negative eigenvalues, Graepel et al. [25] propose an SVM in pseudo-Euclidean space, and Pekalska et al. [26] also consider a generalized nearest mean classifier and Fisher linear discriminant classifier in the same space. Following the notation in 2.1.1, they assume that the samples lie in a Kreĭn space $\mathcal{K} = \mathcal{H}_+ \oplus \mathcal{H}_-$ with similarities given by $\psi(a,b) = \langle a_+, b_+ \rangle_{\mathcal{H}_+} - \langle a_-, b_- \rangle_{\mathcal{H}_-}$. Such classifiers are their standard versions in the Hilbert space $\mathcal{H} = \mathcal{H}_+ \oplus \mathcal{H}_-$ with associated inner product $\langle a, b \rangle_{\mathcal{H}} = \langle a_+, b_+ \rangle_{\mathcal{H}_+} + \langle a_-, b_- \rangle_{\mathcal{H}_-}$. This is equivalent to flipping the sign of the negative eigenvalues of the similarity matrix S: let $\tilde{\Lambda}_{\mathrm{f}} = \mathrm{diag}\left(|\lambda_1|, \ldots, |\lambda_n|\right)$, and then the similarity matrix after spectrum flip is $\tilde{S}_{\mathrm{f}} = U^T \tilde{\Lambda}_{\mathrm{f}} U$. It has been noted that this is the same as replacing the original eigenvalues of S with its singular values [15].

2.1.4 Spectrum Shift

Spectrum shift is another popular approach to modifying a similarity matrix into a kernel matrix: since $S + \lambda I = U^T(\Lambda + \lambda I)U$, any indefinite similarity matrix can be made PSD by shifting its spectrum by at least the absolute value of its minimum eigenvalue $|\lambda_{\min}(S)|$. Let $\tilde{\Lambda}_{\mathrm{s}} = S + |\min\{\lambda_{\min}(S), 0\}|\, I$, which is used to form the modified similarity matrix $\tilde{S}_{\mathrm{s}} = U^T \tilde{\Lambda}_{\mathrm{s}} U$. Compared with spectrum clip and flip, spectrum shift only enhances all the self-similarities by the amount of $|\lambda_{\min}(S)|$ and does not change the similarity between any two different samples. Roth et

al. [27] propose spectrum shift for clustering nonmetric proximity data and show that $\tilde{S}_{\mathrm{s}}$ preserves the group structure of the original data represented by S. Specifically, they consider minimizing the clustering cost function[2]

$$f(\{\mathcal{X}_\ell\}) = -\sum_{\ell=1}^{N} \left(\sum_{\substack{i,j \in \mathcal{X}_\ell \\ i \neq j}} \frac{\psi(x_i, x_j)}{|\mathcal{X}_\ell|} \right), \tag{2}$$

where $\mathcal{X}$ is the set of samples to cluster, $\{\mathcal{X}_\ell \mid \ell = 1, \ldots, N\}$ is a partition of $\mathcal{X}$ with N elements, and $|\mathcal{X}_\ell|$ denotes the cardinality of set $\mathcal{X}_\ell$. Then (2) is invariant under spectrum shift.

Recently, Zhang et al. [28] proposed training an SVM only on the k-nearest neighbors of each test sample, called SVM-KNN. They used spectrum shift to produce a kernel from the similarity data. Their experimental results on image classification demonstrated that SVM-KNN performs comparably to a standard SVM classifier, but with significant reduction in training time.

2.1.5 Spectrum Square

The fact that $SS^T \succeq 0$ for any $S \in \mathbb{R}^{n \times n}$ led us to consider using SS^T as a kernel, which is valid even when S is not symmetric. For symmetric S, this is equivalent to squaring its spectrum since $SS^T = U^T \Lambda^2 U$. It is also true that using SS^T is the same as defining a new similarity function $\tilde{\psi}$ for any $a, b \in \Omega$ as

$$\tilde{\psi}(a, b) = \sum_{i=1}^{n} \psi(a, x_i)\psi(x_i, b).$$

We note that for symmetric S, treating SS^T as a kernel matrix is equivalent to representing each x_i by its similarity feature vector $s_i = \begin{bmatrix} \psi(x_i, x_1) & \ldots & \psi(x_i, x_n) \end{bmatrix}^T$ since $K(x_i, x_j) = \tilde{\psi}(x_i, x_j) = \langle s_i, s_j \rangle$. The concept of treating similarities as features is discussed in more detail in Section 3.

2.2 Consistent Treatment of Training and Test Samples

Consider a test sample x that is the same as a training sample x_i. Then if one uses an empirical risk minimization (ERM) classifier trained with modified similarities $\tilde{S}$, but uses the unmodified test similarities, represented by vector $s = \begin{bmatrix} \psi(x, x_1) & \ldots & \psi(x, x_n) \end{bmatrix}^T$, the same sample will be treated inconsistently. In general, one would like to modify the training and test similarities in a consistent fashion, that is, to modify the underlying similarity function rather than only modifying the S. In this context, given S and $\tilde{S}$, we term a transformation T on test samples *consistent* if $T(s_i)$ is equal to the ith row of $\tilde{S}$ for $i = 1, \ldots, n$.

One solution is to modify the training and test samples all at once. However, when test samples are not known beforehand, this may not be possible. For such cases, Wu et al. [15] proposed to first modify S and train the classifier once using the modified $n \times n$ similarity matrix $\tilde{S}$, and then for each test sample modify its s in an effort to be consistent with the modified similarities used to train the model. Their approach is to re-compute the modification on the augmented $(n + 1) \times (n + 1)$ similarity matrix

$$S' = \begin{bmatrix} S & s \\ s^T & \psi(x, x) \end{bmatrix}$$

to form $\tilde{S}'$, and then let the modified test similarities $\tilde{s}$ be the first n elements of the last column of $\tilde{S}'$. The classifier that was trained on $\tilde{S}$ is then applied on $\tilde{s}$. To implement this approach, they also propose a fast algorithm to do eigenvalue decomposition of S' by using the results of the eigenvalue decomposition of S. However, this approach does not guarantee consistency.

To attain consistency, we note that both the spectrum clip and flip modifications can be represented by linear transformations, that is, $\tilde{S} = PS$, where P is the corresponding transformation matrix, and we propose to apply the

[2] They originally use dissimilarities in their cost function, and we reformulate it into similarities with the assumption that the relationship between dissimilarities and similarities is affine.

same linear transformation P on s, such that $\tilde{s} = Ps$. For spectrum flip, the linear transformation is $P_{\mathrm{f}} = U^T M_{\mathrm{f}} U$, where

$$M_{\mathrm{f}} = \mathrm{diag}\left(\mathrm{sgn}(\lambda_1), \ldots, \mathrm{sgn}(\lambda_n)\right).$$

For spectrum clip, the linear transformation is $P_{\mathrm{c}} = U^T M_{\mathrm{c}} U$, where

$$M_{\mathrm{c}} = \mathrm{diag}\left(I_{\{\lambda_1 \geq 0\}}, \ldots, I_{\{\lambda_n \geq 0\}}\right),$$

and I_A is the indicator function of event A. Recall that using $\tilde{S}$ implies embedding the training samples in a Euclidean space. For spectrum clip, this linear transformation is equivalent to embedding the test sample as a feature vector into the same space of the embedded training samples:

Proposition 1 *Let $\tilde{S}_{\mathrm{c}}$ be the Gram matrix of the column vectors of $\tilde{X}_{\mathrm{c}} \in \mathbb{R}^{m \times n}$, where $\mathrm{rank}(\tilde{X}_{\mathrm{c}}) = m$. For a given s, let $\tilde{x}_{\mathrm{c}} = \arg\min_{x \in \mathbb{R}^m} \|\tilde{X}_{\mathrm{c}}^T x - s\|_2$, then $\tilde{X}_{\mathrm{c}}^T \tilde{x}_{\mathrm{c}} = P_{\mathrm{c}} s$.*

The proof is in the appendix.

On the other hand, there is no linear transformation to ensure consistency for spectrum shift. For our experiments using spectrum shift, we adopt the approach of [15], which for this case is to let $\tilde{s} = s$, because spectrum shift only affects self-similarities.

3 Similarities as Features

Similarity-based classification problems can be formulated into standard learning problems in Euclidean space by treating the similarities between a sample x and the n training samples as features [25, 26, 29–31]. That is, represent sample x by the similarity feature vector s.

Graepel et al. [25] consider applying an SVM with a linear kernel on similarity feature vectors by solving the following problem:

$$\underset{w,b}{\text{minimize}} \quad \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{n} L(w^T s_i + b, y_i) \tag{3}$$

with variables $w \in \mathbb{R}^n$ and $b \in \mathbb{R}$, where $C > 0$ and $L(\alpha, \beta) \triangleq \max(1 - \alpha\beta, 0)$ is the hinge loss function. Liao et al. [31] also propose to apply an SVM on similarity feature vectors; they use a Gaussian radial basis function (RBF) kernel.

In order to make the solution w sparser, which helps ease the computation of the discriminant function $f(s) = w^T s + b$, Graepel et al. [29] substitute the ℓ_1-norm regularization for the squared ℓ_2-norm regularization in (3), and propose a linear programming (LP) machine:

$$\underset{w,b}{\text{minimize}} \quad \|w\|_1 + C\sum_{i=1}^{n} L(w^T s_i + b, y_i). \tag{4}$$

Another approach is the potential support vector machine (P-SVM) [32, 33], which solves

$$\begin{array}{ll} \underset{\alpha}{\text{minimize}} & \frac{1}{2}\|y - S\alpha\|_2^2 + \epsilon\|\alpha\|_1 \\ \text{subject to} & \|\alpha\|_\infty \leq C, \end{array} \tag{5}$$

where $C > 0$ and $\epsilon > 0$. We note that by strong duality (5) is equivalent to

$$\underset{\alpha}{\text{minimize}} \quad \frac{1}{2}\|y - S\alpha\|_2^2 + \epsilon\|\alpha\|_1 + \gamma\|\alpha\|_\infty \tag{6}$$

for some $\gamma > 0$. One can see from (6) that P-SVM is equivalent to the lasso regression [34] with an extra ℓ_∞-norm regularization term. The use of multiple regularization terms in P-SVM is similar to the elastic net [35], which uses ℓ_1 and squared ℓ_2 regularization together.

The algorithms above minimize the empirical risk with regularization. In addition, Pekalska et al. consider generative classifiers for similarity feature vectors; they propose a regularized Fisher linear discriminant classifier [26] and a regularized quadratic discriminant classifier [30].

3.1 Generalization Bounds of Similarity SVM Classifiers

In the formulation above, we attempt to learn $n+1$ parameters using n training samples, this casts doubt on the classifier's ability to generalize. To investigate this, we analyze two forms of the familiar SVM classifier: using similarity as a kernel as discussed in Section 2, and a linear SVM using the similarities as features. When similarities are used as features, we show that good generalization performance can be achieved by training the SVM on a small subset of $m < n$ randomly selected training examples, and we compare this to the established analysis for the kernelized SVM.

The SVM learns a discriminant function f by minimizing a regularization term and the empirical risk

$$\hat{R}_{\mathcal{D}}(f, L) = \frac{1}{n}\sum_{i=1}^{n} L(f(x_i), y_i),$$

where f is the discriminant function to be learned. The form of the regularization used plays a key role in proving generalization bounds.

Observation 1 (SVM representation) *The SVM using either similarity as a kernel or similarity as features learns a linear discriminant function $f(s) = w^T s + b$ by minimizing empirical risk subject to some smoothness constraint $\mathcal{N}$, that is,*

$$\underset{f}{\text{minimize}} \quad \hat{R}_{\mathcal{D}}(f, L) + \lambda_n \mathcal{N}(f),$$

where $\lambda_n = \frac{1}{2nC}$. Using a positive definite similarity as a kernel corresponds to setting $\mathcal{N}(f) = w^T S w$, while using (arbitrary) similarity as features corresponds to setting $\mathcal{N}(f) = w^T w$.

The proof is in the appendix.

To simplify the following analysis, we do not directly investigate the SVM classifier as presented in Observation 1; instead, as is standard in SVM learning theory, we investigate the following constrained version of the problem:

$$\begin{aligned} \underset{f}{\text{minimize}} \quad & \hat{R}_{\mathcal{D}}(f, L_t) \\ \text{subject to} \quad & \mathcal{N}(f) \leq \beta^2, \end{aligned} \tag{7}$$

with truncated hinge loss $L_t \triangleq \min(L, 1) \in [0, 1]$ and $f(s) = w^T s$ stripped of the intercept b.

The following generalization bound for the SVM follows directly from the results in [36].

Theorem 1 (Generalization Bounds of Similarity as Kernel) *Suppose that (x, y) and the elements of $\mathcal{D}$ are drawn i.i.d. from a distribution on $\Omega \times \{\pm 1\}$. Let F_S be the set of real-valued functions $\left\{f(s) = w^T s \mid w^T S w \leq \beta^2\right\}$ for some finite β, and let ψ be positive definite with $\psi(a, a) \leq \kappa^2$ for all $a \in \Omega$ and some finite κ. Then with probability at least $1 - \delta$ with respect to $\mathcal{D}$, every function f in F_S satisfies*

$$P\left(yf(s) \leq 0\right) \leq \hat{R}_{\mathcal{D}}(f, L_t) + 4\beta\kappa\sqrt{\frac{1}{n}} + \sqrt{\frac{\ln(2/\delta)}{2n}}.$$

The proof is in the appendix.

That is, with high probability as $n \to \infty$, the misclassification rate is tightly bounded by the empirical risk $\hat{R}_{\mathcal{D}}(f, L_t)$, implying that a discriminant function trained by (7) with $\mathcal{N}(f) = w^T S w$ generalizes well to unseen data.

We state a weaker result in Theorem 2 for the SVM using similarity as features. Let the features be the similarities to $m < n$ randomly chosen prototypes $\{(\tilde{x}_1, \tilde{y}_1), \ldots, (\tilde{x}_m, \tilde{y}_m)\} \subseteq \mathcal{D}$ so that $\tilde{s}$ is the $m \times 1$ vector with ith element $\psi(x, \tilde{x}_i)$. Results will be obtained on the remaining $n - m$ training data $\widetilde{\mathcal{D}} = \mathcal{D} \backslash \{(\tilde{x}_1, \tilde{y}_1), \ldots, (\tilde{x}_m, \tilde{y}_m)\}$.

Theorem 2 (Generalization Bounds of Similarity as Features) *Suppose that (x, y) and the elements of $\mathcal{D}$ are drawn i.i.d. from a distribution on $\Omega \times \{\pm 1\}$. Let F_I be the set of real valued functions such that $\left\{f : f(\tilde{s}) = w^T \tilde{s} \mid w^T w \leq \beta^2\right\}$ and let ψ be positive definite with $\psi(a, b) \leq \kappa^2$ for all $a, b \in \Omega$. Then with probability at least $1 - \delta$ with respect to $\widetilde{\mathcal{D}}$, every function in F_I satisfies*

$$P\left(yf(s) \leq 0\right) \leq \hat{R}_{\widetilde{\mathcal{D}}}(f, L_t) + 4\beta\kappa^2\sqrt{\frac{m}{n}} + \sqrt{\frac{\ln(2/\delta)}{2n}}.$$

The proof is in the appendix.

Theorem 2 differs from Theorem 1 in the term $\sqrt{m/n}$, which means that if the m prototypes used as features grow no faster than $o(n)$, then with high probability as $n \to \infty$, the misclassification rate is tightly bounded by the empirical risk on the remaining training set $\hat{R}_{\tilde{\mathcal{D}}}(f, L_t)$. Note that Theorem 2 is unable to claim anything about the generalization when $m = n$ and the entire training set is chosen as prototypes. For a further discussion, see the Appendix.

Balcan et al. [37] analyze similarities as features and show that if a similarity is good in the sense that the expected intraclass similarity is sufficiently large compared to the expected interclass similarity, then given n training samples, there will exist a linear separator on the similarities as features that has a specifiable maximum error at a margin that depends on n [37, Theorem 3, 4]. Wang et al. [38] show that under slightly less restrictive assumptions on the similarity function there exists with high probability a convex combination of simple classifiers on the similarities as features which has a maximum specifiable error. We note that similarities as features may not capture discriminative information if there is a large intraclass variance compared to the interclass variance, even if the classes are well-separated. A simple example is if the two classes are generated by Gaussian distributions with highly-ellipsoidal covariances, and the similarity function is taken to be a negative linear function of the distance.

4 Similarity-based Weighted Nearest-Neighbors

In this section, we consider design goals and propose solutions for weighted nearest-neighbors for similarity-based classification. Nearest-neighbor learning is the algorithmic parallel of the *exemplar* model of human learning [39]. Weighted nearest-neighbor algorithms are task-flexible because the weights on the neighbors can be used as probabilities as long as they are non-negative and sum to one. For classification, such weights can be summed for each class to form posteriors, which is helpful for use with asymmetric misclassification costs and when the similarity-based classifier is a component of a larger decision-making system. As a lazy learning method, weighted nearest-neighbor classifiers do not require training before the arrival of test samples. This can be advantageous to certain applications where the amount of training data is huge, or there are a large number of classes, or the training data is constantly evolving.

4.1 Design Goals for Similarity-based Weighted k-NN

In this section, we use x_i to denote the ith nearest neighbor from the training set $\mathcal{D}$ of a test sample as defined by the similarity function ψ for $i = 1, \ldots, k$, and y_i to denote the label of x_i. Also, we redefine S as the $k \times k$ matrix of the similarities between the k-nearest neighbors and s the $k \times 1$ vector of the similarities between the test sample x and its k-nearest neighbors. For each test sample, weighted k-NN assigns weights w_i to the ith nearest neighbor. Weighted k-NN classifies the test sample x as the class $\hat{y}$ that is assigned the most weight,

$$\hat{y} = \arg\max_{g \in \mathcal{G}} \sum_{i=1}^{k} w_i I_{\{y_i = g\}}. \tag{8}$$

It is common to additionally require that the weights be nonnegative and normalized such that the weights form a posterior distribution over the set of classes $\mathcal{G}$. Then the estimated probability for class g is $\sum_{i=1}^{k} w_i I_{\{y_i=g\}}$, which can be used with asymmetric misclassification costs or class priors.

An intuitive and standard approach to weighting nearest neighbors is to give larger weight to neighbors that are more similar to the test sample. Formally, we state:

Design Goal 1 (Affinity): w_i should be an increasing function of $\psi(x, x_i)$.

In addition, we propose a second design goal. In practice, some samples in the training set are often very similar, for example, a random sampling of emails by one person may include many emails from the same thread that contain repeated text due to replies and forwarding. Such similar training samples provide highly-correlated information to the classifier. In fact, many of the nearest neighbors may provide very similar information which can bias the classifier. To address this problem, one can choose weights to downweight highly similar samples and ensure a diverse set of the neighbors has a voice in the classification decision. We formalize this goal as:

Design Goal 2 (Diversity): w_i and w_j should be decreasing functions of $\psi(x_i, x_j)$.

Next we propose two approaches to weighting neighbors for similarity-based classification that aim to satisfy these goals.

4.2 Kernel Ridge Interpolation (KRI) Weights

First, we show that weights formed by solving kernel regularized linear interpolation satisfy the design goals. Gupta et al. [40] proposed weights for k-NN in Euclidean space that satisfy a linear interpolation with maximum entropy (LIME) objective such that the weight vector $w \in \mathbb{R}^k$ solves the convex optimization problem:

$$\begin{aligned} \underset{w}{\text{minimize}} \quad & \left\| \sum_{i=1}^{k} w_i x_i - x \right\|_2^2 - \lambda H(w) \\ \text{subject to} \quad & \sum_{i=1}^{k} w_i = 1,\ w_i \geq 0,\ i = 1, \ldots, k, \end{aligned} \tag{9}$$

where $\lambda > 0$ is a regularization parameter and $H(w) = -\sum_{i=1}^{k} w_i \log w_i$ is the entropy of the weights. The first term of the objective in (9), linear interpolation, favors weights that approximate the test point by a convex combination of the training samples while the second term, entropy maximization, favors more uniform weights.

We simplify (9) to a quadratic programming (QP) problem by replacing the negative entropy regularization with a ridge regularizer $w^T w$, and we rewrite (9) in matrix form:

$$\begin{aligned} \underset{w}{\text{minimize}} \quad & \frac{1}{2} w^T X^T X w - x^T X w + \frac{\lambda}{2} w^T w \\ \text{subject to} \quad & w \succeq 0, \quad \mathbf{1}^T w = 1, \end{aligned} \tag{10}$$

where $X = \begin{bmatrix} x_1 & x_2 & \cdots & x_k \end{bmatrix}$ and $\succeq$ denotes componentwise inequality. Note that (10) is completely specified in terms of the inner products of the feature vectors: $\langle x_i, x_j \rangle$ and $\langle x, x_i \rangle$, and thus we refer to the solution to (10) as *kernel ridge interpolation* (KRI) weights. Generalizing from inner products to similarities, we form the KRI similarity-based weights:

$$\begin{aligned} \underset{w}{\text{minimize}} \quad & \frac{1}{2} w^T S w - s^T w + \frac{\lambda}{2} w^T w \\ \text{subject to} \quad & w \succeq 0, \quad \mathbf{1}^T w = 1. \end{aligned} \tag{11}$$

There are three terms in the objective function (11). Acting alone, the linear term $-s^T w$ would give all the weight to the 1-nearest neighbor. This is prevented by the ridge regularization term $\frac{1}{2}\lambda w^T w$, which regularizes the variance of w due to the norm-constraint on w and therefore pushes the weights towards the uniform weights. These two terms work together to give more weight to the training samples that are more similar to the test sample, and thus help the resulting weights satisfy the first design goal of rewarding neighbors with high affinity to the test sample. The quadratic term of (11) can be expanded as follows,

$$\frac{1}{2} w^T S w = \frac{1}{2} \sum_{i,j} \psi(x_i, x_j) w_i w_j.$$

From the above expansion, one sees that holding all else constant, an increase in $\psi(x_i, x_j)$ or $\psi(x_j, x_i)$ will cause a decrease in both w_i and w_j. Thus the quadratic term tends to down-weight the neighbors that are similar to each other and acts to achieve the second design goal of spreading the weight among a diverse set of neighbors.

Experimentally, we found little statistically significant different between using negative entropy or the ridge regularization for the KRI weights. Analytically, entropy regularization leads to an exponential form for the weights that can be used to prove consistency [41]. Computationally, the ridge regularizer is more practical because it results in a QP with box constraints and an equality constraint if the S matrix is PSD or approximated by a PSD matrix, and can thus be solved by the fast SMO algorithm [16], which is commonly employed by SVM solvers.

4.2.1 Kernel Ridge Regression Weights

A closed-form solution to (11) is possible if one relaxes the problem by removing the constraints $w_i \in [0,1]$ and $\sum_i w_i = 1$ that ensure the weight solution forms a probability mass function. Then for PSD S, the objective $\frac{1}{2}w^T S w - s^T w + \frac{1}{2}\lambda w^T w$ is solved by

$$w = (S + \lambda I)^{-1} s. \tag{12}$$

The k-NN decision rule (8) using these weights is equivalent to classifying by maximizing the discriminant of a local kernel ridge regression. For each class $g \in \mathcal{G}$, local kernel ridge regression (without intercept) solves

$$\underset{\beta_g}{\text{minimize}} \quad \sum_{i=1}^{k} \left(I_{\{y_i = g\}} - \langle \beta_g, \phi(x_i) \rangle \right)^2 + \lambda \langle \beta_g, \beta_g \rangle, \tag{13}$$

where ϕ denotes the mapping from the sample space Ω to a Hilbert space with inner product $\langle \phi(x_i), \phi(x_j) \rangle = \psi(x_i, x_j)$. Each solution to (13) yields the discriminant $f_g(x) = \langle \beta_g, \phi(x) \rangle = \nu_g^T (S + \lambda I)^{-1} s$ for class g [42], where $\nu_g = \begin{bmatrix} I_{\{y_1 = g\}} & \dots & I_{\{y_k = g\}} \end{bmatrix}^T$. Maximizing $f_g(x)$ over $g \in \mathcal{G}$ produces the same estimated class label as (8) using the weights given in (12), thus we refer to these weights as kernel ridge regression (KRR) weights.

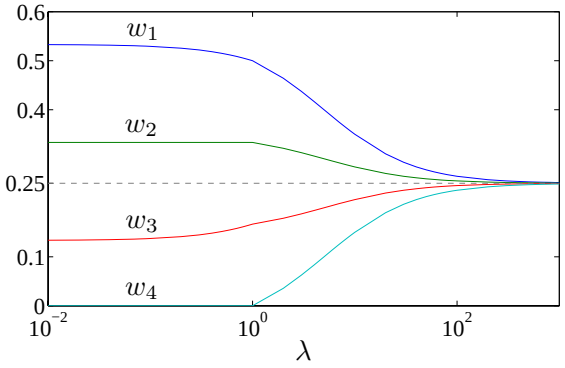
For a non-PSD S, it is possible that $S + \lambda I$ is singular. In the experiments, we compare handling non-PSD S by flip, clip, shift, or taking the pseudo-inverse (pinv) $(S + \lambda I)^{\dagger}$.

4.2.2 Illustrative Examples

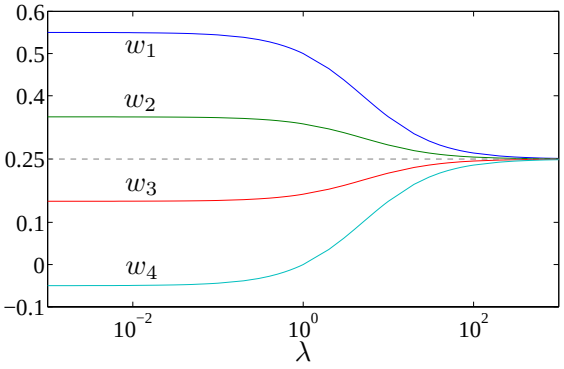
We illustrate the KRI and KRR[3] with three toy examples shown on this page and the next. For each example, there are $k = 4$ nearest-neighbors, and the KRI and KRR weights are shown for a range of regularization parameter λ.

In Example 1, affinity is illustrated. One sees from s that the four distinct training samples are not equally similar to the test sample, and from S that the training samples have zero similarity to each other. Both KRI and KRR give more weight to training samples that are more similar to the test sample, illustrating that the weighting methods achieve the design goal of affinity.

Example 1: $s^T = \begin{bmatrix} 4 & 3 & 2 & 1 \end{bmatrix}$, $S = \begin{bmatrix} 5 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 \\ 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 5 \end{bmatrix}$



KRI weights



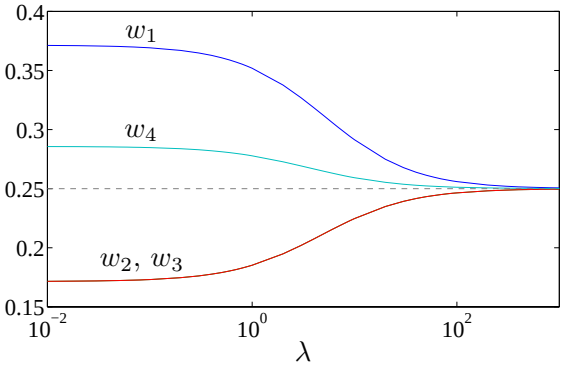
KRR weights

In Example 2, diversity is illustrated. The four training samples all have similarity 3 to the test sample, but x_2 and x_3 are very similar to each other, and are thus weighted down as prescribed the by the design goal of diversity. Because of the symmetry of the similarities, the weights for x_2 and x_3 are exactly the same for both KRI and KRR.

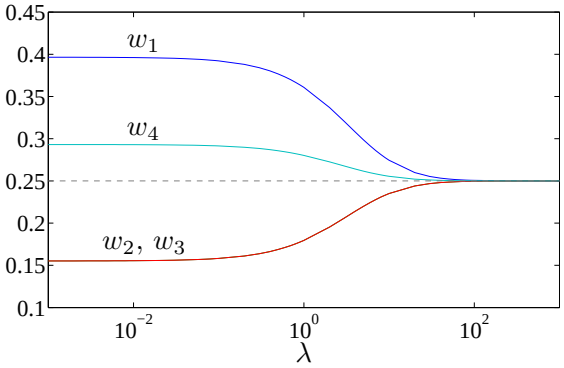
In Example 3, the interaction between the two design goals is illustrated. The S matrix is the same as in Example 2, but here s is no longer uniform. In fact, although x_1 is less similar to the test sample than x_3, x_1 receives more

[3]For the purpose of comparison, we show the normalized KRR weights $\tilde{w}$ where $\tilde{w} = \left(I - \frac{1}{k}\mathbf{1}\mathbf{1}^T\right) w + \frac{1}{k}\mathbf{1}$. This does not affect the result of classification since each weight is shifted by the same constant.

Example 2: $s^T = \begin{bmatrix} 3 & 3 & 3 & 3 \end{bmatrix}$, $S = \begin{bmatrix} 5 & 1 & 1 & 1 \\ 1 & 5 & 4 & 2 \\ 1 & 4 & 5 & 2 \\ 1 & 2 & 2 & 5 \end{bmatrix}$



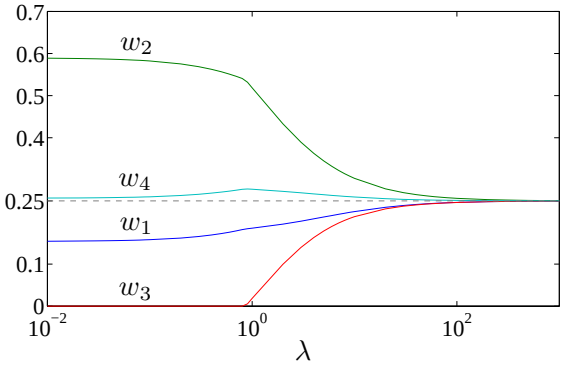
KRI weights



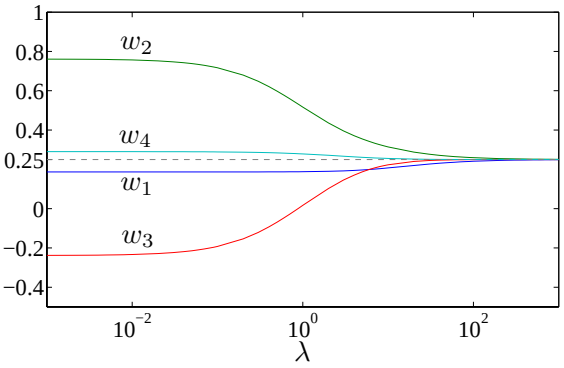
KRR weights

weight because it is less similar to other training samples. The affinity goal allots the largest weight to the most similar neighbor x_2, but because x_2 and x_3 are highly similar, the diversity goal forces them to share weight, resulting in x_3 receiving little weight. One observes from these examples that the KRR weights tend to be smoother than the KRI weights, because the KRI weights have the additional constraint that the weights must be within $[0, 1]$, while the KRR weights are unconstrained.

Example 3: $s^T = \begin{bmatrix} 2 & 4 & 3 & 3 \end{bmatrix}$, $S = \begin{bmatrix} 5 & 1 & 1 & 1 \\ 1 & 5 & 4 & 2 \\ 1 & 4 & 5 & 2 \\ 1 & 2 & 2 & 5 \end{bmatrix}$



KRI weights



KRR weights

5 Generative Classifiers

A generative framework for similarity-based classification termed *similarity discriminant analysis* (SDA) has been proposed that models the class-conditional distributions of similarity statistics [43]. Generative classifiers provide probabilistic outputs that can be easily fused with other probabilistic information or used to minimize expected misclassification costs. First we review the basic SDA model, then consider a local approach [44] and a mixture model approach designed to reduce bias.

Let X denote a random test sample and x denote a realization of X. Assume that the relevant information about X's class label is captured by a finite set $\mathcal{T}(X)$ of M descriptive statistics, where the mth descriptive statistic is

denoted $\mathcal{T}_m(X)$. For example, given a centroid μ_g to represent the gth class for $g \in \mathcal{G}$, a useful set of descriptive statistics might be the similarities to class centroids:

$$\mathcal{T}(x) = \{\psi(x, \mu_1), \psi(x, \mu_2), \ldots, \psi(x, \mu_G)\}, \tag{14}$$

where $\mu_g \in \Omega$ denotes the gth class centroid. Although there are many possible definitions of a centroid given similarities, in the SDA work and in this paper a class centroid is defined to be that training sample that has maximum sum-similarity to the other training samples of its class. One alternate set of descriptive statistics is to use the similarity to each of the n training samples as descriptive statistics such that $\mathcal{T}(x_i) = s_i$. The centroid-based descriptive statistics given by (14) were shown to be overall more effective than other considered descriptive statistics on simulations and a small set of real-data experiments [45].

The minimum expected misclassification cost classification rule for SDA is: classify x as the class

$$\hat{y} = \arg\min_{f \in \mathcal{G}} \sum_{g=1}^{G} C(f, g) P(\mathcal{T}(x)|Y = g) P(Y = g), \tag{15}$$

where $C(f, g)$ is the cost of classifying a sample as class f if the truth is class g.

To estimate the class-conditional distributions $\{P(\mathcal{T}(x)|Y = g)\}$ the SDA model estimates the expected value of the mth descriptive statistic $\mathcal{T}_m(X)$ with respect to the class conditional distribution $P(\mathcal{T}(x)|Y = g)$ to be the average of the training sample data for each class g:

$$E_{P(\mathcal{T}(x)|g)}[\mathcal{T}_m(X)] = \frac{1}{|\mathcal{X}_g|} \sum_{z \in \mathcal{X}_g} \mathcal{T}_m(z), \tag{16}$$

where $\mathcal{X}_g$ is the set of training samples from class g. Given the $M \times G$ constraints specified by (16), the SDA model estimates each class-conditional distribution as the solution to (16) with maximum entropy, which is the exponential [46]:

$$\hat{P}(\mathcal{T}(x)|g) = \prod_{m=1}^{M} \gamma_{gm} e^{\lambda_{gm} \mathcal{T}_m(x)}. \tag{17}$$

Substituting the maximum entropy solution (17) into (15) yields the SDA classification rule: classify x as the class

$$\hat{y} = \arg\min_{f \in \mathcal{G}} \sum_{g=1}^{G} C(f, g) P(Y = g) \prod_{m=1}^{M} \gamma_{gm} e^{\lambda_{gm} \mathcal{T}_m(x)} . \tag{18}$$

Each pair of parameters $\{\lambda_{gm}, \gamma_{gm}\}$ can be separately calculated from the constraints given in (16) by one-dimensional optimization (e.g. Nelder-Mead) and normalization.

5.1 Reducing SDA Model Bias

The SDA model may not be flexible enough to capture a given decision boundary. To address this model bias issue, one could apply SDA locally to a neighborhood of k nearest neighbors from each class for each test point, or learn a mixture SDA model.

In this paper we experimentally compare to *local SDA* [44] with local centroid-similarity descriptive statistics given by (14), in which SDA is applied to the nearest k neighbors of a test point, where the parameter k is trained by cross-validation. If any class in the neighborhood has fewer than three samples, there are not enough data samples to fit distributions of similarities, so every λ is assumed to be zero, and the local SDA model is reduced to a simple local centroid classifier. Given a discrete set of possible similarities, local SDA has been shown to be a consistent classifier in the sense that its error rate asymptotically converges to the Bayes error rate under the usual asymptotic assumptions that the number of training samples $n \to \infty$, the neighborhood size $k \to \infty$, but that the neighborhood size grows relatively slowly such that $k/n \to 0$ [44].

Cazzanti explored mixture SDA models analogous to Gaussian mixture models (GMM). He developed a hybrid k-medoids/EM approach to fitting SDA mixture models [45], which has the following decision rule:

$$\hat{y} = \arg\min_{f \in \mathcal{G}} \sum_{g=1}^{G} C(f, g) \left(\prod_{h=1}^{G} \sum_{l=1}^{c_h} w_{ghl} \gamma_{ghl} e^{\lambda_{ghl} \psi(x, \mu_{hl})} \right) P(Y = g), \tag{19}$$

where $\sum_{l=1}^{c_h} w_{hl} = 1$, and $w_{hl} > 0$. The number of components c_h was determined by cross-validation. The component weights $\{w_{ghl}\}$ and the component SDA parameters $\{\lambda_{ghl}\}$ and $\{\gamma_{ghl}\}$ were estimated by an expectation-maximization (EM) algorithm [45], analogous to EM-fitting of a GMM, except that the centroids are calculated only once (rather than iteratively) at the beginning using a k-medoids algorithm [47]. Simulations and a small set of experiments using the similarities between x and c_h prototypes for class h as descriptive statistics ($\mathcal{T} = \{\psi(x, \mu_{h1}), \psi(x, \mu_{h2}), \ldots, \psi(x, \mu_{hc_h})\}$) showed that this mixture SDA performed similarly to local SDA, but that the model training for mixture SDA was much more computationally intensive [45].

6 Experiments

We compare eight similarity-based classification approaches: a linear and a Gaussian RBF SVM using similarities as features, the P-SVM [32], a local SVM using the given similarities as a kernel (SVM-KNN) [28], a global SVM using the given similarities as a kernel, local SDA, k-NN, and the three weighted k-NN methods discussed in Section 4: the proposed KRR and KRI weights, and affinity weights as a control, defined by $w_i = a\psi(x, x_i)$, $i = 1, \ldots, k$, where a is a normalization constant. Results are shown in Tables 3 and 4.

For algorithms that require a PSD S, we make S PSD by a shift, clip, or flip of the eigenvalues, as discussed in Sections 2.1.4, 2.1.2, and 2.1.3, and pinv for KRR weights. The results in Table 3 are for clip; the experimental differences between shift, clip and flip, and pinv are discussed in Section 6.4 and shown in Table 4.

6.1 Data Sets

We tested the proposed classifiers on eight real data sets representing a diverse set of similarities ranging from the human judgement of audio signals to sequence alignment of proteins. These data sets along with the randomized partitions will be made available at `http://idl.ee.washington.edu/SimilarityLearning/`.

The *Amazon-47* data set, created for this paper, consists of 204 books written by 47 authors. Each book listed on amazon.com links to the top four books that customers bought after viewing it, along with the percentage of customers who did so. We take the similarity of book A to book B to be the percentage of customers who bought B after viewing A, and the classification problem is to determine the book's author.

The *Aural Sonar* data set is from a recent paper which investigated people's ability to distinguish different types of sonar signals by ear [48]. The signals were returns from a broadband active sonar system, with 50 target-of-interest signals and 50 clutter signals. Every pair of signals was assigned a similarity score from 1 to 5 by two randomly chosen human subjects unaware of the true labels, and these scores were added to produce a 100×100 similarity matrix with integer values from 2 to 10.

The *Caltech-101* data set [49] is an object recognition benchmark data set consisting of 8677 images from 101 object categories. Similarities between images were computed using the pyramid match kernel [5] on SIFT features [50]. Here, the similarity is positive definite.

The *Face Rec* data set consists of 945 sample faces of 139 people from the NIST Face Recognition Grand Challenge data set.[4] There are 139 classes, one for each person. Similarities for pairs of the original three-dimensional face data were computed as the cosine similarity between integral invariant signatures based on surface curves of the face [51]. The original paper demonstrated comparable results to the state-of-the-art using these similarities with a 1-NN classifier.

The *Mirex07* data set was obtained from the human-rated, fine-scale audio similarity data used in the MIREX 2007 Audio Music Similarity and Retrieval[5] task. Mirex07 consists of 3090 samples, divided roughly evenly amongst 10 classes that correspond to different music genres. Humans judged how similar two songs are on a 0-10 scale with 0.1 increments. Each song pair was evaluated by three people, and the three similarity values were averaged. Self-similarity was assumed to be 10, the maximum similarity. The classification task is to correctly label each song with its genre.

The *Patrol* data set was collected by Driskell and McDonald [52]. Members of seven patrol units were asked to name five members of their unit; in some cases the respondents inaccurately named people who were not in their unit, including people who did not belong to any unit. Of the original 385 respondents and named people, only the ones that were named at least once were kept, reducing the data set to 241 samples. The similarity between any two people

[4] See `http://face.nist.gov/frgc/`.

[5] See `http://www.music--ir.org/mirex/2007/index.php/Audio_Music_Similarity_and_Retrieval/`.

Table 2: Cross-validation Parameter Choices

All local methods	k:	1, 2, 3, ..., 16, 32, 64, 128
KRR	λ:	$10^{-3}, 10^{-2}, \ldots, 10$
KRI	λ:	$10^{-6}, 10^{-5}, \ldots, 10,$ and 10^{6}
PSVM	ϵ:	$10^{-4}, 10^{-3}, \ldots, 10$
PSVM	C:	$10^{0}, 10^{1}, \ldots, 10^{4}$
SVM-KNN	C:	$10^{-3}, 10^{-2}, \ldots, 10^{5}$
SVM (linear, shift, clip, flip)	C:	$10^{-3}, 10^{-2}, \ldots, 10^{5}$
SVM (RBF)	C:	$10^{-3}, \ldots, 10$
SVM (RBF)	γ:	$10^{-5}, 10^{-4}, \ldots, 10$

a and b is $(N(a,b)+N(b,a))/2$, where $N(a,b)$ is the number of times person a names person b. Thus, this similarity ϕ has a range $\{0, 0.5, 1\}$. The classification problem is to estimate to which of the seven patrol units a person belongs, or to correctly place them in an eighth class that corresponds to "not in any of the units."

The *Protein* data set has sequence-alignment similarities for 213 proteins from 4 classes,[6] where class one through four contains 72, 72, 39, and 30 samples, respectively [53]. As further discussed in the results, we define an additional similarity termed ***RBF-sim*** for the Protein data set: $\psi_{\mathrm{RBF}}(x_i, x_j) = e^{-\|s(x_i)-s(x_j)\|_2}$, where $s(x)$ is the 213×1 vector of similarities with lth component $\psi(x, x_l)$.

The *Voting* data set comes from the UCI Repository [54]. It is a two-class classification problem with 435 samples, where each sample is a categorical feature vector with 16 components and three possibilities for each component. We compute the value difference metric [55] from the categorical data, which is a dissimilarity that uses the training class labels to weight different components differently so as to achieve maximum probability of class separation.

Shown in Figure 1 are the similarity matrices of all the data sets. The rows and columns are ordered by class label; on many of the data sets, particularly those with a fewer number of classes, a block-diagonal structure is visible along the class boundaries, indicated by tick marks. Note that a purely block-diagonal similarity matrix would indicate a particularly easy classification problem, as objects have nonzero similarity only to objects of the same class.

6.2 Other Experimental Details

For each data set, we randomly selected 20% of the data for testing and used the remaining 80% for training. We chose the classifier parameters such as C for the SVM, λ for kernel ridge regression and interpolation weights, and k for local classifiers by 10-fold cross-validation on the training set, and then used them to classify the held out test data. This process was repeated for 20 random partitions of test and training data and the statistical significance of the classification error was computed by a one-sided Wilcoxon signed-rank test. Multi-class implementations of the SVM classifiers used $\binom{n}{2}$ pairwise classifiers.

Nearest neighbors for local methods were determined using symmetrized similarities $\psi(x, x_i) + \psi(x_i, x)$ (only Amazon and Patrol are natively asymmetric). Cross-validation choices are listed in Table 2. These choices were based on recommendations and usage in previous literature, and on preliminary experiments we conducted with a larger range of cross-validation parameters on the Voting and Protein data sets.

6.3 Results

The average error and standard deviation of error across the 20 randomized training/test partitions are shown in Table 3. The bold results in each column indicate the classifier with lowest average error; also bolded are any classifiers that were not statistically significantly worse than the classifier with lowest average error.

[6] The original data set has 226 samples with 9 classes. As is standard practice with this data set, we removed those classes which contain less than 7 samples.

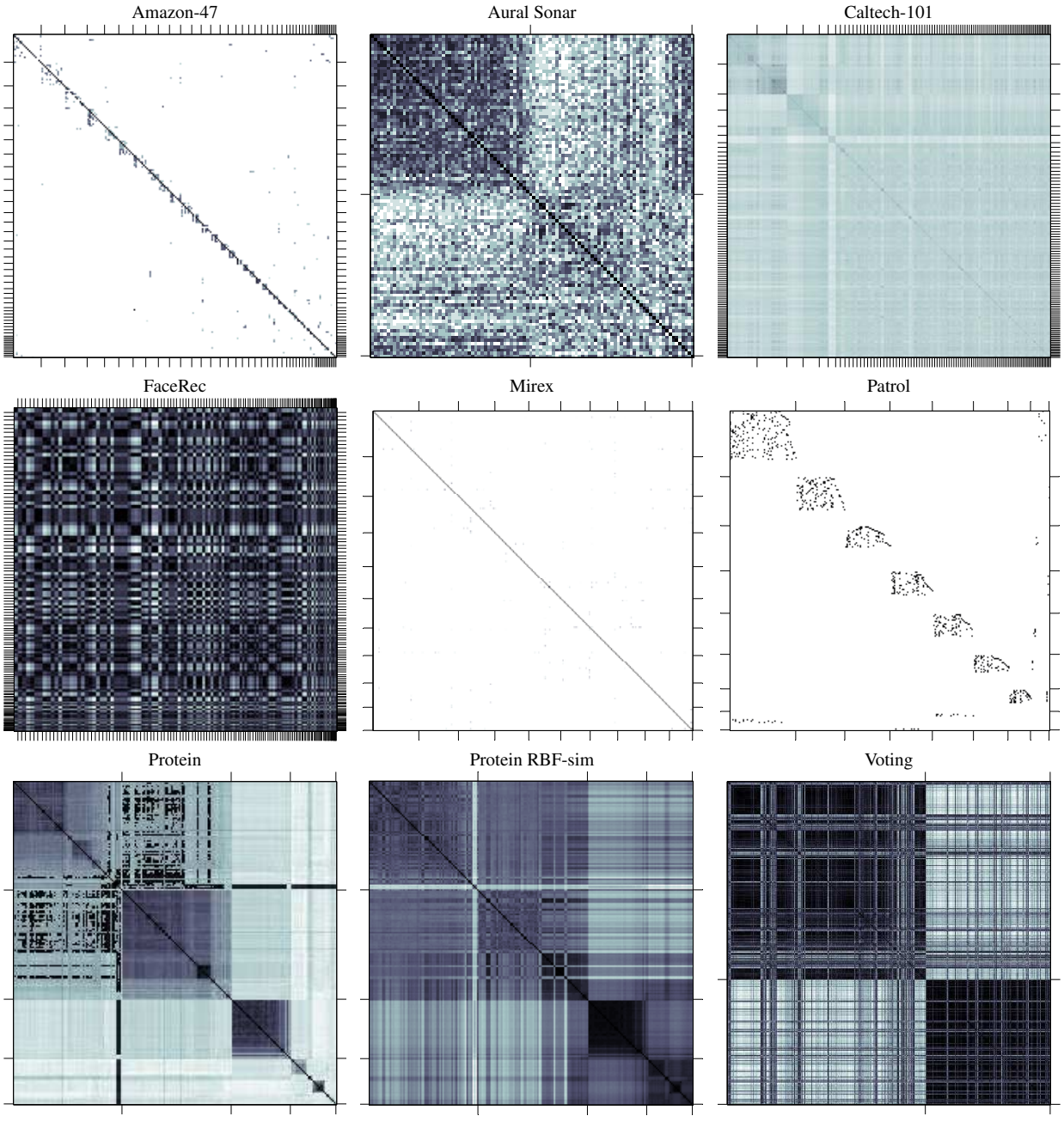


Figure 1: Similarity matrices of each data set with class divisions indicated by tick marks; black corresponds to maximum similarity and white to zero.

The similarity matrices in Figure 1 show that Aural Sonar and Voting exhibit fairly nice block-diagonal structures, indicating that these are somewhat easy classification problems. This is reflected in the relatively low errors across the board and few statistically significant differences in performance. More interesting results can be observed on the more difficult classification problems posed by the other data sets.

The Amazon-47 data set is very sparse with at most four non-zero similarities per row. With such sparse data, one might expect a 1-NN classifier to perform well; indeed for the uniform k-NN classifier, the cross-validation chose $k = 1$ on all 20 of the randomized partitions. For all of the local classifiers, the k chosen by cross-validation on this data set was never larger than $k = 3$, and out of the 20 randomized partitions, $k = 1$ was chosen the majority of the time for all the local classifiers. In contrast, the global classifiers, SVM-global and SVM-linear, perform poorly on this sparse data set. The Patrol data set is the next sparsest data set, and the results show a similar pattern. However, the Mirex data set is also relatively sparse, and yet the global classifiers do well, in particular the SVM with similarities as features. The Amazon-47 and Patrol data sets do differ from the Mirex data set in that the self-similarities for are not all maximal. Whether (and how) this difference causes or correlates the relative differences in performance is an open question.

The Protein data set exhibits large differences in performance, with the statistically significantly best performances achieved by the three SVMs that use similarity as features. The reason that similarities on features performs so well while others do so poorly can be seen in the Protein similarity matrix in Figure 1. The first and second classes (roughly the first and second thirds of the matrix) exhibit a strong inter-class similarity, and rows belonging to the same class exhibit very similar patterns, thus treating the rows of the similarity matrix as a feature space provides good discrimination of classes. To investigate this effect, we transformed the entire data set using a radial basis function (RBF) to create a similarity based on the Euclidean distance between rows of the original similarity matrix, yielding the 213×213 matrix Protein RBF-sim. One sees from Table 3 that this transformation increases the performance of classifiers across the board, indicating that this is indeed a better measure of similarity for this data set. Furthermore, after this transformation we see a complete turnaround in performance: for Protein RBF-sim the SVM-linear classifier is the worst performer, and the basic k-NN does better than the best classifier given the original Protein similarities.

The Caltech-101 data set is the largest data set, and with 8677 samples in 101 classes, an analysis of the structure of this similarity matrix is difficult. Here we see the most dramatic enhancement in performance (25% lower error) by using the KRR and KRI weights rather than k-NN or the affinity k-NN, suggesting that there are highly correlated samples that bias the classification. In contrast, for the Amazon-47, Aural Sonar, Face Rec, Mirex, and Patrol data sets there is only a small win by using the KRI or KRR weights, or a statistically insignificant small decline in performance (we hypothesize this occurs because of overfitting due to the additional parameter λ). On Protein the KRR error is 1/3 the error of the other weighted methods; this is a consequence of using the pinv rather than another type of spectrum modification, as can be seen from Table 4. The other significant difference between the weighting methods is a roughly 10% improvement in average error on Voting by using the KRR or KRI weights. In conclusion, the use of diverse weights may not matter on some data sets, but can be very helpful on certain data sets.

SVM-KNN was proposed by Zhang et al. [28] in part as a way to reduce the computations required to train a global SVM, and their results showed that it performed similarly to a global SVM. That is somewhat true here, but some differences emerge. For the Amazon-47 and Patrol data sets the local methods all do well including SVM-KNN, but the global methods do poorly, including the global SVM. On the other hand, the global SVM is statistically significantly better than SVM-KNN on Caltech-101, even though the best performance on that data set is achieved by a local method (KRR). From this sampling of data sets, we conclude that applying the SVM locally or globally can in fact make a difference, but whether it is a positive or negative difference depends on the application.

P-SVM performs similarly to the other two SVMs that use similarities as features, but is statistically significantly better than the other two SVMs using similarities as features on Amazon, Caltech-101, Protein RBF, and Voting; and statistically significantly worse on FaceRec and Mirex. The results do not show statistically significant differences between using the linear or RBF version of the SVM with similarities as features. Lastly, there are only two large differences between the SVM with similarity as kernel or the threesome of SVMs with similarities as features. For Caltech-101 the SVM with similarity as kernel is statistically significantly better than all three SVMs using similarities as features, and on Protein the situation is reversed, but on the modified Protein RBF data set the SVM with similarity as kernel is not statistically different from the PSVM.

Table 3: % Test misclassified averaged over 20 randomized training/test partitions.

	Amazon-47	Aural Sonar	Caltech-101
k-NN	16.95 (4.85)	17.00 (7.65)	41.55 (0.95)
affinity k-NN	**15.00** (4.77)	**15.00** (6.12)	39.20 (0.86)
KRI k-NN (clip)	17.68 (4.75)	**14.00** (6.82)	30.13 (0.42)
KRR k-NN (pinv)	**16.10** (4.90)	15.25 (6.22)	**29.90** (0.44)
Local SDA	16.83 (5.11)	17.75 (7.66)	41.99 (0.52)
SVM-KNN (clip)	17.56 (4.60)	**13.75** (7.40)	36.82 (0.60)
SVM-similarities as kernel (clip)	81.34 (4.77)	**13.00** (5.34)	33.49 (0.78)
SVM-similarities as features (linear)	76.10 (6.92)	**14.25** (6.94)	38.18 (0.78)
SVM-similarities as features (RBF)	75.98 (7.33)	**14.25** (7.46)	38.16 (0.75)
P-SVM	70.12 (8.82)	**14.25** (5.97)	34.23 (0.95)
	FaceRec	Mirex	Patrol
k-NN	4.23 (1.43)	61.21 (1.97)	**11.88** (4.42)
affinity k-NN	4.23 (1.48)	61.15 (1.90)	**11.67** (4.08)
KRI k-NN (clip)	4.15 (1.32)	61.20 (2.03)	**11.56** (4.54)
KRR k-NN (pinv)	4.31 (1.86)	61.18 (1.96)	12.81 (4.62)
Local SDA	4.55 (1.67)	60.94 (1.94)	**11.77** (4.62)
SVM-KNN (clip)	4.23 (1.25)	61.25 (1.95)	**11.98** (4.36)
SVM-similarities as kernel (clip)	4.18 (1.25)	57.83 (2.05)	38.75 (4.81)
SVM-similarities as features (linear)	4.29 (1.36)	**55.54** (2.52)	42.19 (5.85)
SVM-similarities as features (RBF)	**3.92** (1.29)	**55.72** (2.06)	40.73 (5.95)
P-SVM	4.05 (1.44)	63.81 (2.70)	40.42 (5.94)
	Protein	Protein RBF	Voting
k-NN	29.88 (9.96)	**0.93** (1.71)	5.80 (1.83)
affinity k-NN	30.81 (6.61)	**0.93** (1.71)	5.86 (1.78)
KRI k-NN (clip)	30.35 (9.71)	**1.05** (1.72)	**5.29** (1.80)
KRR k-NN (pinv)	9.53 (5.04)	**1.05** (1.72)	**5.52** (1.69)
Local SDA	17.44 (6.52)	**0.93** (1.71)	6.38 (2.07)
SVM-KNN (clip)	11.86 (5.50)	**1.16** (1.72)	**5.23** (2.25)
SVM-similarities as kernel (clip)	5.35 (4.60)	**1.16** (1.72)	**4.89** (2.05)
SVM-similarities as features (linear)	**3.02** (2.76)	2.67 (2.12)	5.40 (2.03)
SVM-similarities as features (RBF)	**2.67** (2.97)	2.44 (2.60)	5.52 (1.77)
P-SVM	**1.86** (1.89)	**1.05** (1.56)	**5.34** (1.72)

6.4 Flip, Clip, or Shift?

Different approaches to modify similarities to form a kernel were discussed in Sec 2.1. We experimentally compared flip, clip, and shift for the KRI weights, SVM-KNN, and SVM and flip, clip, shift and pinv for the KRR weights on the nine data sets. Table 4 shows the five data sets for which at least one method showed significantly different results depending on the choice of spectrum modification.

For KRR weights, one sees that the pinv solution is never statistically significantly worse than flip, clip, or shift, which are worse than pinv at least once. For KRI weights, the significant differences are negligible, but based on average error we recommend one use clip.

Flip takes the absolute value of the eigenvalues, which is similar to the effect of using SS^T (as discussed in Section 2.1.5), which for an SVM is equivalent to using the SVM on similarities-as-features. Thus it is not surprising that for the Protein data set, which we have seen in Table 3 works best with similarities as features, flip makes a large positive difference for SVM-KNN and SVM. One sees different effects of the spectrum modification on the local methods vs the global SVM because the modification is only done locally for the local methods.

Table 4: Flip, clip, shift, and pinv comparison. Table shows % test misclassified averaged over 20 randomized training/test partitions for the five data sets that exhibit statistically significant differences between these spectrum modifications. If there are statistically significant differences for a given algorithm and a given data set, then the worst score, and scores not statistically better, are shown in italics.

	KRI					KRR				
	Amazon	Mirex	Patrol	Protein	Voting	Amazon	Mirex	Patrol	Protein	Voting
flip	17.56	61.17	11.67	31.28	5.34	16.22	61.12	*12.08*	*30.47*	5.29
clip	17.68	61.20	11.56	30.35	5.34	16.22	*61.22*	11.67	*30.35*	5.34
shift	17.68	61.25	*13.23*	30.35	5.29	16.34	*61.25*	11.88	*30.35*	5.52
pinv	-	-	-	-	-	16.10	61.18	*12.81*	9.53	5.52

	SVM-KNN					SVM				
	Amazon	Mirex	Patrol	Protein	Voting	Amazon	Mirex	Patrol	Protein	Voting
flip	17.56	61.25	11.88	1.74	5.23	*84.27*	56.34	*47.29*	1.51	*4.94*
clip	17.56	61.25	11.98	*11.86*	5.23	81.34	*57.83*	38.75	*5.35*	4.89
shift	17.56	61.25	11.88	*30.23*	5.34	77.68	*85.29*	40.83	23.49	*5.17*

7 Conclusions and Some Open Questions

Similarity-based learning is a practical learning framework for many problems in bioinformatics, computer vision, and problems regarding human behavior. Kernel methods can be applied in this framework, but similarity-based learning creates a richer set of challenges because the data may not be natively PSD. In this paper we explored four different approximations of similarities: flipping, clipping, and shifting eigenvalues, and in some cases a pseudoinverse solution. Experimental results show small but sometimes statistically significant differences. Based on the theoretical justification and results, we suggest practitioners clip. Flipping eigenvalues does create significantly better performance for the original Protein problem because, as we noted earlier, flipping eigenvalues has a similar effect to using the similarities-as-features, which is discriminating for these classes. However, it should be easy to recognize when flip will be advantageous, modify the similarity as we did for the Protein RBF problem, and possibly achieve even better results. Concerning approximating similarities, we addressed the issue of consistent treatment of training and test samples when approximating the similarity to be PSD. Although our linear solution is consistent, we do not argue it is optimal, and consider this issue still open.

A fundamental question is whether it is more advisable to approximate the similarities as kernels or to use the similarities as features. We showed that the difference for SVMs is a regularization difference, and that for approximating similarities as kernels generalization bounds can be proven using standard learning theory machinery. However, it is not straightforward to apply standard learning theory machinery to similarities-as-features because the normal Rademacher complexity bounds do not hold with the resulting adaptive non-independent functions. To address this, we considered splitting the training set into prototype-similarities-as-features and a separate set to evaluate the empirical risk. Even then, we were only able to show generalization bounds if the number of prototypes grows slowly. Complementary results have been shown for similarities-as-features [37, 38], but further analysis would be valuable.

We proposed design goals of affinity and diversity for weighting nearest-neighbors, and suggested two related methods for constructing weights that satisfy these design goals. Experimental results on eight diverse data sets demonstrate that the proposed weighted k-NN methods can significantly reduce error compared to standard k-NN. In particular, on the largest data Caltech-101, the proposed KRI and KRR weights provide a roughly 25% improvement over k-NN and affinity-weighted k-NN. The Caltech-101 similarities are PSD, and it may be that the KRI and KRR methods are particularly sensitive to approximations of the matrix S. Preliminary experiments using the un-modified local S and solving KRI or KRR weight objective functions using a global optimizer show an increase in performance may be possible at the price of greatly increased computational time. Compared to the local SVM (SVM-KNN), the proposed KRR weights were statistically significantly worse on the two-class data sets Aural Sonar and Patrol, but statistically significantly better for both of the highly multi-class data sets, Amazon-47 and Caltech-101.

Overall, the results show that local methods are effective for similarity-based learning. It is tempting from the obvious discrepancy between the performance of local and global methods on Amazon and Patrol to argue that local methods will work best for sparse similarities. However, Mirex is also relatively sparse, and the best methods are global. The Amazon and Patrol data sets are differentiated from the other data sets in that they are the only two data sets with non-maximal self-similarities, and this issue may be the root of the discrepancy. For local methods an open question not addressed here is how to efficiently find nearest-neighbors given only similarities. Some research has been done in this area, for example, methods have been developed for fast search with logarithmic query times that depend on how "disordered" the similarity is [56], wherein a disorder constant D of a set of samples is the D that ensures that if x_i is the kth most similar sample to x, and x_j is the qth most similar sample to x, then x is among the $D(q+k)$ most similar samples to x_j.

Lastly, we note that similarity-based learning can be considered a special case of graph-based learning (see, for example, [57]), where the graph is fully-connected. However, we have seen no cross-referential literature between these two fields, and it remains an open question which techniques developed for the one problem will be useful for the other other problem.

Appendix

Proof of Proposition 1: Recall the eigenvalue decomposition $\tilde{S}_{\mathrm{c}} = U^T \tilde{\Lambda}_{\mathrm{c}} U$. After removing the zero eigenvalues in $\tilde{\Lambda}_{\mathrm{c}}$ and their corresponding eigenvectors in U and U^T, one can express $\tilde{S}_{\mathrm{c}} = \check{U}^T \check{\Lambda}_{\mathrm{c}} \check{U}$, where $\check{\Lambda}_{\mathrm{c}}$ is an $m \times m$ diagonal matrix with m the number of nonzero eigenvalues and $\check{U}$ an $m \times n$ matrix satisfying $\check{U}\check{U}^T = I$. The vector representation of the training samples implicitly used via $\tilde{S}_{\mathrm{c}}$ is $\tilde{X}_{\mathrm{c}} = \check{\Lambda}_{\mathrm{c}}^{1/2} \check{U}$. Given test similarity vector s, the least-squares solution to the equation $\tilde{X}_{\mathrm{c}}^T x = s$ is $\tilde{x}_{\mathrm{c}} = \left(\tilde{X}_{\mathrm{c}} \tilde{X}_{\mathrm{c}}^T\right)^{-1} \tilde{X}_{\mathrm{c}} s$. Let $\tilde{s}_{\mathrm{c}}$ be the vector of inner products between $\tilde{x}_{\mathrm{c}}$ and the training samples $\tilde{X}_{\mathrm{c}}$, then

$$\tilde{s}_{\mathrm{c}} = \tilde{X}_{\mathrm{c}}^T \tilde{x}_{\mathrm{c}} = \tilde{X}_{\mathrm{c}}^T \left(\tilde{X}_{\mathrm{c}} \tilde{X}_{\mathrm{c}}^T\right)^{-1} \tilde{X}_{\mathrm{c}} s = \check{U}^T \check{U} s = U^T M_{\mathrm{c}} U s = P_{\mathrm{c}} s. \qquad \blacksquare$$

Proof of Observation 1: The SVM learns a discriminant function $f(x) = \langle \beta^*, \phi(x) \rangle + b^*$ by solving:

$$(\beta^*, b^*) = \underset{\beta \in \mathcal{H}, b \in \mathbb{R}}{\arg\min} \quad \frac{1}{n} \sum_{i=1}^{n} L\big(y_i(\langle \beta, \phi(x_i) \rangle + b)\big) + \lambda_n \langle \beta, \beta \rangle, \tag{20}$$

where $\lambda_n = \frac{1}{2nC}$, $L(x) = \max(1 - x, 0)$ is the hinge loss and ϕ is a mapping from Ω to a Hilbert space $\mathcal{H}$ with inner product $\langle \cdot, \cdot \rangle$.

Note that by the representer theorem, β^* necessarily lies in the span of $\{\phi(x_1), \ldots, \phi(x_n)\}$. To see this, consider that any $\beta \in \mathcal{H}$ can be written as $\beta = \beta_\phi + \beta_{\phi^\perp}$ where $\beta_\phi = \sum_{i=1}^n \alpha_i \phi(x_i)$ and $\langle \beta_{\phi^\perp}, \phi(x_i) \rangle = 0$ for $i \in \{1, \ldots, n\}$. Since $\langle \beta, \phi(x_i) \rangle = \langle \beta_\phi, \phi(x_i) \rangle$ and $\langle \beta, \beta \rangle \geq \langle \beta_\phi, \beta_\phi \rangle$, it must be that ${\beta_{\phi^\perp}}^* = 0$ for any β^* that minimizes (20).

Given the form of (20), the distinction between using similarity as a kernel and using similarity as features manifests itself in $\mathcal{H}$. Specifically, when similarity is used as features, $\mathcal{H}$ is the n-dimensional Euclidean space where $\beta \in \mathbb{R}^n$ and $\phi(x_i) = s_i \in \mathbb{R}^n$. In this case, (20) can be expressed as

$$\underset{\beta \in \mathbb{R}^n, b \in \mathbb{R}}{\arg\min} \quad \frac{1}{n} \sum_{i=1}^n L\big(y_i(\beta^T s_i + b)\big) + \lambda_n \beta^T \beta.$$

Alternatively, when a positive definite similarity is used as a kernel, $\mathcal{H}$ is the reproducing kernel Hilbert space (RKHS) with reproducing kernel ψ. The fact that this space is an RKHS is not central to the proof; we require only the following result [58, Theorem 6.1]. For any positive definite similarity $\psi : \Omega \times \Omega \to \mathbb{R}$, there exists a mapping $\phi : \Omega \to \mathcal{H}$ such that $\langle \phi(a), \phi(b) \rangle = \psi(a, b)$ for all $a, b \in \Omega$. With the corresponding choice of ϕ and because $\beta = \sum_{j=1}^n \alpha_j \phi(x_j)$, the minimization (20) can be expressed as

$$\underset{\alpha \in \mathbb{R}^n, b \in \mathbb{R}}{\arg\min} \quad \frac{1}{n} \sum_{i=1}^n L\big(y_i(\alpha^T s_i + b)\big) + \lambda_n \alpha^T S \alpha,$$

which concludes the proof of Observation 1. ■

The proofs of the generalization bounds of Theorem 1 and Theorem 2 rely on bounding the Rademacher complexity of the function classes F_S and F_I, respectively. We provide the definition of Rademacher complexity here for convenience.

Definition 1 (Rademacher Complexity) *Suppose that $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$ are samples drawn independently from a distribution on Ω and let $F : \Omega \to \mathbb{R}$ be a class of functions. Then, the Rademacher complexity of F is*

$$\mathcal{R}_\mathcal{X}(F) = \mathbf{E}_{\sigma, \mathcal{X}} \left[\sup_{f \in F} \left| \frac{2}{n} \sum_{i=1}^n \sigma_i f(X_i) \right| \right],$$

where $\sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_n\}$ is a set of independent random variables drawn according to $P\{\sigma_i = +1\} = P\{\sigma_i = -1\} = 1/2$ for all i.

The following lemma establishes that for a class of bounded functions F, the generalization error for any $f \in F$ is bounded above by a function of $\hat{R}_\mathcal{D}(f, L)$ and $\mathcal{R}_\mathcal{D}(F)$.

Lemma 1 [36, Theorem 7] *Suppose that (X, Y) and the elements of $\mathcal{D}$ are drawn i.i.d. from a distribution on $\Omega \times \{\pm 1\}$. Let $F \subseteq \{f : \Omega \to [-\gamma, \gamma]\}$ be a class of bounded functions and suppose $L : \mathbb{R} \to [0, 1]$ is Lipschitz with constant C and satisfies $L(a) \geq I_{\{a \leq 0\}}$. Then with probability at least $1 - \delta$ with respect to $\mathcal{D}$, every function in F satisfies*

$$P\big(Yf(X) \leq 0\big) \leq \hat{R}_\mathcal{D}(f, L) + 2C\mathcal{R}_\mathcal{D}(F) + \sqrt{\frac{\ln(2/\delta)}{2n}}.$$

For the proof of Theorem 1, we also require the following bound on the Rademacher complexity of kernel methods.

Lemma 2 [36, Lemma 22] *Suppose that the elements of $\mathcal{D}$ are drawn i.i.d. from a distribution on $\Omega \times \{\pm 1\}$. Let F_K be the set of functions $\{f : f(x) = \sum_i \alpha_i K(X_i, x)\}$ such that $\sum_{i,j} \alpha_i \alpha_j K(X_i, X_j) \leq \beta^2$. Then by Jensen's inequality,*

$$\mathcal{R}_\mathcal{D}(F_K) \leq 2\beta \sqrt{\frac{\mathbf{E}K(X, X)}{n}}.$$

Proof of Theorem 1: Theorem 1 is an application of Lemmas 1 and 2 for the function class $F_S \subseteq \{f \mid f(s) = w^T s, w^T S w \leq \beta^2\}$. In Lemma 2, the kernel function K is directly replaced by ψ and $\mathbf{E}K(X, X) \leq \kappa^2$. It can be verified that the function class is bounded as $|f(s)| \leq \beta\kappa$ for all $f \in F_S$, and thus we may apply Lemma 1. Noting that L_t is Lipschitz with $C = 1$ completes the proof. ■

Proof of Theorem 2: Recall that $\tilde{s} = \big(\psi(x,\tilde{x}_1),\ldots,\psi(x,\tilde{x}_m)\big)^T$ where $\{(\tilde{x}_1,\tilde{y}_1),\ldots,(\tilde{x}_m,\tilde{y}_m)\} \subseteq \mathcal{D}$ is a subset of the training data and $\widetilde{\mathcal{D}} = \mathcal{D}\backslash\{(\tilde{x}_1,\tilde{y}_1),\ldots,(\tilde{x}_m,\tilde{y}_m)\}$ is the remaining training set. It is tempting to apply Lemma 2 with the linear kernel, but in this case, this does not satisfy the definition of the function class $F_I \subseteq \{f \mid f(\tilde{s}) = w^T\tilde{s}, w^Tw \leq \beta^2\}$ defined on these prototypes. The following bound on the Rademacher complexity mirrors that of [36, Lemma 22], but requires an important modification noted below:

$$
\begin{aligned}
\mathcal{R}_{\widetilde{\mathcal{D}}}(F_I) &= \mathbf{E}_{\widetilde{\mathcal{D}},\sigma}\left[\sup_{f\in F_I}\left|\frac{2}{n}\sum_{i=1}^{n-m}\sigma_i f(\tilde{s}_i)\right|\right] \\
&\leq \frac{2}{n}\mathbf{E}_{\widetilde{\mathcal{D}},\sigma}\left[\sup_{\|w\|\leq\beta}\left|w^T\left(\sum_{i=1}^{n-m}\sigma_i\tilde{s}_i\right)\right|\right] \\
&\overset{(a)}{\leq} \frac{2}{n}\mathbf{E}_{\widetilde{\mathcal{D}},\sigma}\left[\beta\left\|\sum_{i=1}^{n-m}\sigma_i\tilde{s}_i\right\|\right] \\
&= \frac{2\beta}{n}\mathbf{E}_{\widetilde{\mathcal{D}},\sigma}\left[\left(\sum_{i,j}\sigma_i\sigma_j\tilde{s}_i^T\tilde{s}_j\right)^{1/2}\right] \\
&\overset{(b)}{\leq} \frac{2\beta}{n}\left(\sum_{i,j}\mathbf{E}_{\widetilde{\mathcal{D}},\sigma}\,\sigma_i\sigma_j\tilde{s}_i^T\tilde{s}_j\right)^{1/2} \\
&= \frac{2\beta}{n}\left(\sum_{i=1}^{n-m}\mathbf{E}_{\widetilde{\mathcal{D}}}\,\tilde{s}_i^T\tilde{s}_i\right)^{1/2} \\
&= \frac{2\beta}{n}\left(\sum_{i=1}^{n-m}\sum_{j=1}^{m}\mathbf{E}_{\widetilde{\mathcal{D}}}\,\psi^2(x_i,\tilde{x}_j)\right)^{1/2} \\
&\leq 2\beta\kappa^2\sqrt{\frac{m}{n}-\frac{m^2}{n^2}} \\
&\leq 2\beta\kappa^2\sqrt{\frac{m}{n}}
\end{aligned}
$$

where (a) follows from Cauchy-Schwartz and (b) follows from Jensen's inequality. Note that in the proof of Theorem 1 and in [36, Lemma 22] the Cauchy-Schwartz inequality is applied in the RKHS space whereas here it is applied in $\mathbb{R}^m$. Because of this, the upper bound involves $\sum_{j=1}^{m}\psi^2(x_i,\tilde{x}_j)$ instead of only $\psi(x_i,x_i)$.

It can be verified that the function class is bounded as $|f(\tilde{s})| \leq \beta\kappa^2\sqrt{m}$ for all $f \in F_I$, and thus we may apply Lemma 1. As before, noting that L_t is Lipschitz with $C = 1$ completes the proof. ■

The proof of Theorem 2 illustrates why using similarities as features has a poorer guarantee on the generalization than using similarities as a kernel. Specifically, the function class corresponding to regularization on w^Tw is too large. Of course, this flexibility can be mitigated by using only a set of m prototypes whose size grows as $o(n)$, which can be seen as an additional form of capacity control.

References

[1] S. Santini and R. Jain, "Similarity measures," *IEEE Trans. Pattern Anal. and Machine Intel.*, vol. 21, no. 9, pp. 871–883, Sept. 1999.

[2] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. New York: John Wiley & Sons, 2001.

[3] Y. Rubner, C. Tomasi, and L. J. Guibas, "The earth mover's distance as a metric for image retrieval," *Intl. J. Computer Vision*, vol. 40, no. 2, pp. 99–121, Nov. 2000.

[4] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Trans. Pattern Anal. and Machine Intel.*, vol. 24, no. 4, pp. 509–522, April 2002.

[5] K. Grauman and T. Darrell, "The pyramid match kernel: Efficient learning with sets of features," *J. Machine Learning Research*, vol. 8, pp. 725–760, April 2007.

[6] T. F. Smith and M. S. Waterman, "Identification of common molecular subsequences," *J. Molecular Biology*, vol. 147, no. 1, pp. 195–197, March 1981.

[7] D. J. Lipman and W. R. Pearson, "Rapid and sensitive protein similarity searches," *Science*, vol. 227, no. 4693, pp. 1435–1441, March 1985.

[8] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic local alignment search tool," *J. Molecular Biology*, vol. 215, no. 3, pp. 403–410, Oct. 1990.

[9] A. Tversky, "Features of similarity," *Psychological Review*, vol. 84, no. 2, pp. 327–352, July 1977.

[10] A. Tversky and I. Gati, "Similarity, separability, and the triangle inequality," *Psychological Review*, vol. 89, no. 2, pp. 123–154, March 1982.

[11] I. Gati and A. Tversky, "Weighting common and distinctive features in perceptual and conceptual judgments," *Cognitive Psychology*, vol. 16, no. 3, pp. 341–370, July 1984.

[12] I. Borg and P. J. F. Groenen, *Modern Multidimensional Scaling: Theory and Applications*, 2nd ed. New York: Springer, 2005.

[13] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA: MIT Press, 2002.

[14] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multiclass support vector machines," *IEEE Trans. Neural Networks*, vol. 13, no. 2, pp. 415–425, March 2002.

[15] G. Wu, E. Y. Chang, and Z. Zhang, "An analysis of transformation on non-positive semidefinite similarity matrix for kernel machines," University of California, Santa Barbara, Tech. Rep., March 2005.

[16] J. C. Platt, "Using analytic QP and sparseness to speed training of support vector machines," in *Advances in Neural Information Processing Systems*, vol. 11, 1998.

[17] H.-T. Lin and C.-J. Lin, "A study on sigmoid kernels for SVM and the training of non-PSD kernels by SMO-type methods," National Taiwan University, Tech. Rep., March 2003.

[18] C. S. Ong, X. Mary, S. Canu, and A. J. Smola, "Learning with non-positive kernels," in *Proc. Intl. Conf. Machine Learning*, 2004.

[19] B. Haasdonk, "Feature space interpretation of SVMs with indefinite kernels," *IEEE Trans. Pattern Anal. and Machine Intel.*, vol. 27, no. 4, pp. 482–492, April 2005.

[20] N. J. Higham, "Computing a nearest symmetric positive semidefinite matrix," *Linear Algebra and its Applications*, vol. 103, pp. 103–118, May 1988.

[21] R. Luss and A. d'Aspremont, "Support vector machine classification with indefinite kernels," in *Advances in Neural Information Processing Systems*, vol. 20, 2007.

[22] J. Laub and K.-R. Müller, "Feature discovery in non-metric pairwise data," *J. Machine Learning Research*, vol. 5, pp. 801–808, July 2004.

[23] J. Laub, V. Roth, J. M. Buhmann, and K.-R. Müller, "On the information and representation of non-Euclidean pairwise data," *Pattern Recognition*, vol. 39, no. 10, pp. 1815–1826, Oct. 2006.

[24] I. Gati and A. Tversky, "Representations of qualitative and quantitative dimensions," *J. Experimental Psychology: Human Perception & Performance*, vol. 8, no. 2, pp. 325–340, April 1982.

[25] T. Graepel, R. Herbrich, P. Bollmann-Sdorra, and K. Obermayer, "Classification on pairwise proximity data," in *Advances in Neural Information Processing Systems*, vol. 11, 1998, pp. 438–444.

[26] E. Pekalska, P. Paclík, and R. P. W. Duin, "A generalized kernel approach to dissimilarity-based classification," *J. Machine Learning Research*, vol. 2, pp. 175–211, Dec. 2001.

[27] V. Roth, J. Laub, M. Kawanabe, and J. M. Buhmann, "Optimal cluster preserving embedding of nonmetric proximity data," *IEEE Trans. Pattern Anal. and Machine Intel.*, vol. 25, no. 12, pp. 1540–1551, Dec. 2003.

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

[29] T. Graepel, R. Herbrich, B. Schölkopf, A. Smola, P. Bartlett, K.-R. Müller, K. Obermayer, and R. Williamson, "Classification on proximity data with LP–machines," in *Proc. Intl. Conf. Artificial Neural Networks*, vol. 1, 1999, pp. 304–309.

[30] E. Pekalska and R. P. W. Duin, "Dissimilarity representations allow for building good classifiers," *Pattern Recognition Letters*, vol. 23, no. 8, pp. 943–956, June 2002.

[31] L. Liao and W. S. Noble, "Combining pairwise sequence similarity and support vector machines for detecting remote protein evolutionary and structural relationships," *J. Computational Biology*, vol. 10, no. 6, pp. 857–868, 2003.

[32] S. Hochreiter and K. Obermayer, "Support vector machines for dyadic data," *Neural Computation*, vol. 18, no. 6, pp. 1472–1510, June 2006.

[33] T. Knebel, S. Hochreiter, and K. Obermayer, "An SMO algorithm for the potential support vectormachine," *Neural Computation*, vol. 20, no. 1, pp. 271–287, Jan. 2008.

[34] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Royal Statistical Society, Series B (Statistical Methodology)*, vol. 58, no. 1, pp. 267–288, 1996.

[35] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *J. Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, no. 2, pp. 301–320, April 2005.

[36] P. L. Bartlett and S. Mendelson, "Rademacher and Gaussian complexities: Risk bounds and structural results," in *Conf. on Computational Learning Theory, (COLT) 2001*, vol. 2111. Springer, Berlin, 2001, pp. 224–240.

[37] M.-F. Balcan, A. Blum, and N. Srebo, "A theory of learning with similarity functions," *Machine Learning*, vol. 72, no. 1, pp. 89–112, 2008.

[38] L. Wang, C. Yang, and J. Feng, "On learning with dissimilarity functions," in *Proc. Intl. Conf. Machine Learning*, 2007.

[39] R. L. Goldstone and A. Kersten, *Comprehensive Handbook of Psychology*. New Jersey: Wiley, 2003, vol. 4, ch. 22: Concepts and Categorization, pp. 599–621.

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

[41] M. P. Friedlander and M. R. Gupta, "On minimizing distortion and relative entropy," *IEEE Trans. on Information Theory*, vol. 52, no. 1, pp. 238–245, 2006.

[42] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines*. Cambridge, UK: Cambridge University Press, 2000.

[43] L. Cazzanti, M. R. Gupta, and A. J. Koppal, "Generative models for similarity-based classification," *Pattern Recognition*, vol. 41, no. 7, pp. 2289–2297, July 2008.

[44] L. Cazzanti and M. R. Gupta, "Local similarity discriminant analysis," in *Proc. Intl. Conf. Machine Learning*, 2007.

[45] L. Cazzanti, "Generative models for similarity-based classification," Ph.D. dissertation, Department of Electrical Engineerng, University of Washington, 2007.

[46] E. T. Jaynes, "On the rationale for maximum entropy methods," *Proc. IEEE*, vol. 70, no. 9, pp. 939–952, September 1982.

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

[48] S. Philips, J. Pitton, and L. Atlas, "Perceptual feature identification for active sonar echoes," in *Proc. of the 2006 IEEE OCEANS Conf.*, 2006.

[49] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories," in *Proc. of the 2004 IEEE Computer Soc. Conf. on Computer Vision and Pattern Recognition*, 2004.

[50] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Intl. J. of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[51] S. Feng, H. Krim, and I. A. Kogan, "3D face recognition using Euclidean integral invariants signature," in *Proc. IEEE Workshop Statistical Signal Processing*, 2007, pp. 156–160.

[52] J. E. Driskell and T. McDonald, "Identification of incomplete networks," *Florida Maxima Corporation Technical Report*, no. 08–01, 2008.

[53] T. Hofmann and J. M. Buhmann, "Pairwise data clustering by deterministic annealing," *IEEE Trans. on Pattern Anal. and Machine Intel.*, vol. 19, no. 1, pp. 1–14, January 1997.

[54] A. Asuncion and D. J. Newman, "UCI machine learning repository," 2007. [Online]. Available: http://www.ics.uci.edu/∼mlearn/MLRepository.html

[55] C. Stanfill and D. Waltz, "Toward memory-based reasoning," *Communications of the ACM*, vol. 29, no. 12, pp. 1213–1228, December 1986.

[56] N. Goyal, Y. Lifshits, and H. Schütze, "Disorder inequality: A combinatorial approach to nearest neighbor search," in *Proc. ACM Symposium Web Search and Data Mining*, 2008.

[57] X. Zhu, "Semi-supervised learning with graphs," Ph.D. dissertation, Carnegie Mellon University, 2005, cMU-LTI-05-192.

[58] B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds., *Advances in kernel methods: support vector learning*. Cambridge, MA, USA: MIT Press, 1999.