*LETTER* **Communicated by Mauro Forti**

# Theory and Algorithm for Learning with Dissimilarity Functions

**Liwei Wang**
*wanglw@cis.pku.edu.cn*
*Key Laboratory of Machine Perception, MOE School of Electronics Engineering and Computer Science, Peking University, Beijing, 100871, P.R.C.*

**Masashi Sugiyama**
*sugi@cs.titech.ac.jp*
*Department of Computer Science, Tokyo Institute of Technology, Meguro-ku, Tokyo, 152-8552, Japan*

**Cheng Yang**
*yangch@cis.pku.edu.cn*
*Key Laboratory of Machine Perception, MOE School of Electronics Engineering and Computer Science, Peking University, Beijing, 100871, P.R.C.*

**Kohei Hatano**
*hatano@i.kyushu-u.ac.jp*
*Department of Informatics, Kyushu University, Nishi-ku, Fukuoka-City, 819-0395, Japan*

**Jufu Feng**
*fjf@cis.pku.edu.cn*
*Key Laboratory of Machine Perception, MOE School of Electronics Engineering and Computer Science, Peking University, Beijing, 100871, P.R.C.*

**We study the problem of classification when only a dissimilarity function between objects is accessible. That is, data samples are represented not by feature vectors but in terms of their pairwise dissimilarities. We establish sufficient conditions for dissimilarity functions to allow building accurate classifiers. The theory immediately suggests a learning paradigm: construct an ensemble of simple classifiers, each depending on a pair of examples; then find a convex combination of them to achieve a large margin. We next develop a practical algorithm referred to as dissimilarity-based boosting (DBoost) for learning with dissimilarity functions under theoretical guidance. Experiments on a variety of databases demonstrate that the DBoost algorithm is promising for several dissimilarity measures widely used in practice.**

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

 © 2008 Massachusetts Institute of Technology

## 1 Introduction

In classification problems, objects are often represented by feature vectors in a Euclidean space. The Euclidean feature space provides many more analytical tools for classification than other representations. However, such a representation requires the selection of features, which is usually difficult and domain dependent. For example, in the area of fingerprint analysis, it took scientists more than 100 years to discover useful features for fingerprint recognition (Maltoni, Maio, Jain, & Prabhakar, 2003). It is not clear even today what kinds of features have good discrimination ability for human face recognition, and existing feature extraction algorithms are not reliable or accurate (Zhao, Chellappa, Phillips, & Rosenfeld, 2003).

An alternative way is to describe the patterns using dissimilarity functions. Dissimilarity is a function that reflects the "distance" between two objects but with no restrictions on its mathematical properties. For some applications, such as image retrieval, dissimilarity representation has the advantage that it is more convenient to define such a measure than a set of meaningful features (Jacobs, Weinshall, & Gdalyahu, 2000; Jain & Zongker, 1997). A number of image dissimilarities have been proposed (Simard, LeCun, & Denker, 1993; Huttenlocher, Klanderman, & Rucklidge, 1993; Li, Chen, & Chi, 2002; Wang, Zhang, & Feng, 2005) and successfully used in real-world applications. Dissimilarity functions can also be defined on strings, graphs, and other structured objects (Gärtner, 2003; Saigo, Vert, Ueda, & Akutsu, 2004). This procedure thus provides a bridge between classical and structural approaches to pattern classification (Graepel, Herbrich, Bollmann-Sdorra, & Obermayer, 1999; Goldfarb, 1985).

The simplest method to classify objects in dissimilarity representations is the nearest neighbor (NN) rule. NN has an appealing asymptotic property that its error rate converges to the Bayes optimal risk (Hart & Cover, 1967). However, the rate of convergence of NN could be very slow (Fukunaga, 1990), and it is observed in practice that NN is sensitive to the choice of dissimilarity measures and noise (Breiman, Friedman, Olshen, & Stone, 1984).

In contrast to NN, several algorithms take into account global information as well. One type of method first embeds the data into a (possibly pseudo) Euclidean space, and then applies traditional Euclidean classification algorithms, with modifications adapted to the pseudo-Euclidean if necessary (Graepel et al., 1999). Another type of method explicitly constructs feature representations of the objects via their (dis)similarities to a set of prototypes and then runs standard linear separator algorithms like support vector machines (Vapnik, 1998) in the new space (Balcan, Blum, & Vempala, 2004, 2006; Pekalska & Duin, 2002). All of these algorithms demonstrate superior performance to NN on a number of data sets.

In accordance with the progress made in algorithm development, a theoretical foundation of learning with dissimilarities is needed. In some cases, a theory of learning can be given in the form of sufficient conditions for

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

efficient learning. The kernel theory is such an example. The theory states that a large margin is a sufficient condition for good kernels. If the data are well separated with a large margin in an implicit high-dimensional space induced by the kernel, then the kernel is good. That is, there exists a learning algorithm that can generate a classifier having a low generalization error with a small number of training examples. Also important in practice is that the large margin condition implies learning algorithms that are computationally efficient. The algorithms that involve only the inner product in the original input space can be kernelized by replacing the inner product by a positive semidefinite kernel.

Recently, Balcan and Blum (2006) developed a theory of learning with similarities in this way. They defined a notion of what it means for a pairwise function to be a good similarity function for a learning problem. They showed that their definition is sufficient to allow one to learn well and captures the standard notion of a good kernel with some degradation on learning parameters (see also Srebro, 2007; Balcan, Blum, & Srebro, 2008b, for details). This theory immediately suggests algorithms that use feature representation based on prototypes as described earlier and therefore provides a theoretical explanation of their good empirical performances.

In this letter, we develop a theory for learning with dissimilarity functions, in parallel to Balcan and Blum's results. We propose new sufficient conditions for a dissimilarity function that yields good learning guarantees. These sufficient conditions also suggest a computationally efficient algorithm, which is a boosting-type algorithm that combines an ensemble of simple classifiers of special forms. We then make the algorithm more suitable for practical use. An advantage of our theory and algorithm is that they are applicable to unbounded dissimilarity functions, while previous results deal with normalized similarity measures.

The letter is organized as follows. We describe our theory in section 2. In section 3, a practical algorithm, DBoost, is proposed for learning with dissimilarity functions as a consequence of the theory. We provide experimental evidence of the benefits of our algorithm in sections 4 and 5 and conclude in section 6.

## 2 Theory

In this section we describe our theory of learning with dissimilarity functions. We propose sufficient conditions that have good learning guarantees and imply computationally efficient algorithms. We begin with a simple yet intuitively reasonable sufficient condition and then generalize it to incorporate more sophisticated cases.

**2.1 Notations.** By *dissimilarity*, we mean any nonnegative bivariate function $d(x, x')$, where $x, x' \in X$, and $X$ is an instance space. The axioms of

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

a metric—reflectivity, symmetry, and triangle inequality—are not necessary for a dissimilarity function.

Labeled examples are represented by $z, z', z'', \ldots,$ where $z = (x, y)$, $x \in X$, and $y \in \{-1, 1\}$. The examples are drawn randomly and either independently or conditionally independently from the underlying distribution $P$ of the problem over $X \times \{-1, 1\}$. These are always clear from the context. $I$ denotes the indicator function, and $\mathrm{sgn}(x) = 1$ if $x > 0$ and $-1$ otherwise.

**2.2 Sufficient Conditions for Learning with Dissimilarity Functions.** We propose in this section sufficient conditions for a dissimilarity function that are useful for learning.

*2.2.1 Strong $(\epsilon, \gamma)$-goodness.* We first give a notion of good dissimilarity functions, which is quite intuitive. This definition expresses that if most examples are more likely to be close to random examples $z'$ of the same class than to $z''$ of the opposite class, the dissimilarity function is good. More precisely, we use an accuracy parameter $\epsilon$ and a margin parameter $\gamma$ to characterize the goodness of a dissimilarity function.

**Definition 1.** *A dissimilarity function $d(x, x')$ is said to be strongly $(\epsilon, \gamma)$-good for the learning problem, if at least $1 - \epsilon$ probability mass of examples $z$ satisfies*[1]

$$P(d(x, x') < d(x, x'') \mid y' = y,\ y'' = -y) \geq 1/2 + \gamma/2, \tag{2.1}$$

*where the probability is over random examples $z' = (x', y')$ and $z'' = (x'', y'')$.*

The notion of strongly $(\epsilon, \gamma)$-good dissimilarity functions suggests a simple learning algorithm: draw pairs of examples of different labels, and vote according to which class the test example is more likely to be close to. This is summarized in the following theorem:

**Theorem 1:** *If $d$ is a strongly $(\epsilon, \gamma)$-good dissimilarity function, then with probability at least $1 - \delta$ over the choice of $n = (4/\gamma^2) ln(1/\delta)$ pairs of examples $(z_i', z_i'')$ with labels $y_i' = 1$, $y_i'' = -1$, $i = 1, 2, \ldots, n$, the following classifier*

$$H(x) = sgn\,[f(x)]\,,$$

*where*

$$f(x) = \frac{1}{n} \sum_{i=1}^{n} sgn[d(x, x_i'') - d(x, x_i')],$$

[1] In equation 2.1 using $\gamma/2$ is to let $\gamma$ in $[0, 1]$.

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

*has an error rate of no more than $\epsilon + \delta$, that is,*

$$P\,(yf(x) \le 0) \le \epsilon + \delta.$$

**Proof.** The proof uses a technique in Balcan and Blum (2006). Let $M$ be the set of examples satisfying equation 2.1. For any fixed $z = (x, y) \in M$,

$$\begin{aligned} &P(d(x, x') < d(x, x'') \mid y' = y, y'' = -y) \\ &\quad = E(I(d(x, x') < d(x, x'')) \mid y' = y, y'' = -y) \\ &\quad = \frac{1}{2} E(\mathrm{sgn}[d(x, x'') - d(x, x')] \mid y' = y,\ y'' = -y) + \frac{1}{2}, \end{aligned}$$

where the probability is over random examples $z' = (x', y')$ and $z'' = (x'', y'')$. Thus, inequality 2.1 is equivalent to

$$E(\mathrm{sgn}[d(x, x'') - d(x, x')] \mid y' = y, y'' = -y) \ge \gamma.$$

The Chernoff bound then implies that

$$P_{S_n}(y \cdot f(x) \le 0) = P_{S_n}\left(y \cdot \frac{1}{n} \sum_{i=1}^{n} \mathrm{sgn}[d(x, x_i'') - d(x, x_i')] \le 0\right) \le e^{-n\gamma^2/2},$$

where $P_{S_n}$ denotes the probability over the choice of $n$ pairs of training examples. Since the above inequality holds for every $z \in M$, we can take the expectation over all $z \in M$, which results in that the expected error is at most $e^{-n\gamma^2/2}$, that is,

$$E_{z \in M}[P_{S_n}(yf(x) \le 0)] \le e^{-n\gamma^2/2}.$$

Note that

$$E_{z \in M}[P_{S_n}(yf(x) \le 0)] = P_{z \in M,\ S_n}(yf(x) \le 0) = E_{S_n}[P_{z \in M}(yf(x) \le 0)].$$

Thus, we have

$$E_{S_n}[P_{z \in M}(yf(x) \le 0)] \le e^{-n\gamma^2/2}.$$

Next, using the Markov inequality, we obtain that the probability that the error rate over the set $M$ is larger than $\theta$ is at most $e^{-n\gamma^2/2}/\theta$ for arbitrary $\theta > 0$:

$$P_{S_n}[P_{z \in M}(yf(x) \le 0) \ge \theta] \le e^{-n\gamma^2/2}/\theta.$$

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

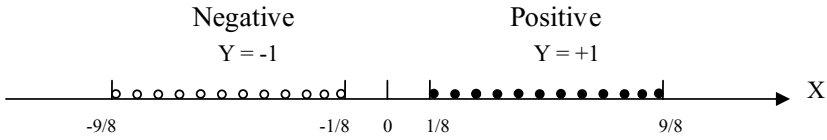


Figure 1: Positive examples are uniformly distributed in $[1/8, 9/8]$. Negative examples are uniformly distributed in $[-9/8, -1/8]$. $d(x, x') = \mid x - x' \mid$.

Finally, setting $\delta = e^{-n\gamma^2/2}/\theta$ and adding the $\epsilon$ probability of examples $z$ not in $M$ completes the proof.

*2.2.2* $(\epsilon, \gamma, B)$*-goodness.* Although definition 1 and its suggested algorithm is natural, the notion of strong $(\epsilon, \gamma)$-goodness may be too restrictive. Many good dissimilarity functions that can be used in a more intellectual way to learn well do not satisfy definition 1. To illustrate this, we show in Figure 1 a simple one-dimensional example. In this learning problem, positive and negative examples are uniformly distributed in $[1/8, 9/8]$ and $[-9/8, -1/8]$, respectively. The dissimilarity function of two points $x$ and $x'$ used here is $d(x, x') = \mid x - x' \mid$. This problem should be perfectly learned with this dissimilarity function. However, for positive example $x \approx 1/8$ or negative example $x \approx -1/8$, the probability $P\left(d(x, x') < d(x, x'') \mid y' = y,\ y'' = -y\right)$ is not one. In fact, it is not difficult to show that for any positive example $x \in [1/8, 1/4]$ or negative example $x \in [-1/4, -1/8]$, we have

$$P(d(x, x') < d(x, x'') \mid y' = y,\ y'' = -y) \leq \frac{1}{2} + \frac{3/4}{2}.$$

That is, $1/8$ probability mass of examples does not have a margin larger than $3/4$. Thus, the dissimilarity function is not strongly $(1/8, \gamma)$-good for any $\gamma > 3/4$.

Notice however, for any example $(x, y)$ in the problem, when randomly choosing $x'$ (or $x''$) of the same (or opposite) class of $x$, if we use only the examples near the boundary, then we would have that $d(x, x') < d(x, x'')$ $(y' = y,\ y'' = -y)$ holds for all examples $(x, y)$. To be concrete, if we draw $x'$ according to a new distribution $\tilde{p}(x \mid y = 1)$, which is the uniform distribution on $[1/8, 3/8]$, and draw $x''$ according to $\tilde{p}(x \mid y = -1)$, which is the uniform distribution on $[-3/8, -1/8]$, we can learn a zero-error classifier. Therefore, with respect to the new distributions $\tilde{p}(x \mid y)$, the dissimilarity function is perfect: it is strongly $(0, 1)$-good for the problem.

Generally if we know that the dissimilarity function is strongly good with respect to distributions $\tilde{p}(x \mid y = 1)$ and $\tilde{p}(x \mid y = -1)$, we could reweight the data as if they were generated from $\tilde{p}$ and learn the classifier in the same way as described in theorem 1.

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

In the following definition, a further step is made. We assume the existence of the new distributions, which are not necessarily known a priori. This definition therefore captures a broad class of dissimilarity functions. Later it will become clear that this assumption alone is sufficient to learn an accurate classifier.

**Definition 2.** *Denote by $p(x \mid y = 1)$ and $p(x \mid y = -1)$ the conditional pdfs of the learning problem. A dissimilarity function $d$ is said to be $(\epsilon, \gamma, B)$-good for the learning problem if:*

1. *There exist two conditional pdfs $\tilde{p}(x \mid y = 1)$ and $\tilde{p}(x \mid y = -1)$ such that for all $x \in X$,*

$$\frac{\tilde{p}(x \mid y = 1)}{p(x \mid y = 1)} \leq \sqrt{B}, \qquad \frac{\tilde{p}(x \mid y = -1)}{p(x \mid y = -1)} \leq \sqrt{B}.$$

2. *At least $1 - \epsilon$ probability mass of examples $z$ satisfies*

$$\tilde{P}(d(x, x') < d(x, x'') \mid y' = y, y'' = -y) \geq 1/2 + \gamma/2, \tag{2.2}$$

*where $\tilde{P}$ is the probability with respect to $\tilde{p}(x' \mid y')$ and $\tilde{p}(x'' \mid y'')$, that is,*

$$\tilde{P}(d(x, x') < d(x, x'') \mid y' = y, y'' = -y) = \int\int I[d(x, x') < d(x, x'')]\tilde{p}(x' \mid y' = y)\, \tilde{p}(x'' \mid y'' = -y)\, dx' dx''.$$

The next theorem says that $(\epsilon, \gamma, B)$-goodness guarantees the existence of a low-error, large-margin classifier, which is a convex combination of the base classifiers:

**Theorem 2.** *If $d$ is a $(\epsilon, \gamma, B)$-good dissimilarity function, then with probability at least $1 - \delta$ over the choice of $n = 16B^2/\gamma^2\ ln(1/\delta)$ pairs of examples $((x_i', 1), (x_i'', -1))$, $i = 1, 2, \ldots, n$, there exists a convex combination classifier $f(x)$ of $n$ base classifiers $h_i(x)$:*

$$f(x) = \sum_{i=1}^{n} \alpha_i h_i(x), \quad \sum \alpha_i = 1, \quad \alpha_i \geq 0,$$

*where*

$$h_i(x) = sgn[d(x, x_i'') - d(x, x_i')],$$

*such that the error rate of the combined classifier at margin $\gamma/2B$ is at most $\epsilon + \delta$, that is,*

$$P\,(y \cdot f(x) \leq \gamma/2B) \leq \epsilon + \delta.$$

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

**Proof.** Denote[2]

$$w_1(x) = \frac{\tilde{p}(x \mid y=1)}{p(x \mid y=1)}, \qquad w_{-1}(x) = \frac{\tilde{p}(x \mid y=-1)}{p(x \mid y=-1)}.$$

Let $M$ be the set of examples satisfying equation 2.2. For a fixed $z = (x, y) \in M$,

$$\begin{aligned}
&\tilde{P}(d(x,x') < d(x,x'') \mid y'=y,\ y''=-y) \\
&= \int\int \tilde{p}(x' \mid y'=y)\ \tilde{p}(x'' \mid y''=-y)\ I[d(x,x') < d(x,x'')]\, dx'dx'' \\
&= \int\int w_y(x')\ w_{-y}(x'')\ p(x' \mid y'=y)\ p(x'' \mid y''=-y) \\
&\quad \times \left\{ \frac{\operatorname{sgn}[d(x,x'') - d(x,x')] + 1}{2} \right\} dx dx' \\
&= \frac{1}{2}\ E\{w_y(x')w_{-y}(x'')\ \operatorname{sgn}[d(x,x'') - d(x,x')] \mid y'=y,\ y''=-y\} + \frac{1}{2}.
\end{aligned}$$

Hence,

$$\tilde{P}(d(x,x') < d(x,x'') \mid y'=y,\ y''=-y) \geq 1/2 + \gamma/2$$

is equivalent to

$$E\{w_y(x')w_{-y}(x'')\ \operatorname{sgn}[d(x,x'') - d(x,x')] \mid y'=y,\ y''=-y\} \geq \gamma.$$

Note that $0 \leq w_1(x')w_{-1}(x'') \leq B$, the above inequality, together with Hoeffding's inequality, implies that

$$P\left\{ y\frac{1}{n}\sum_{i=1}^{n} w_1(x_i')w_{-1}(x_i'')\ \operatorname{sgn}[d(x,x_i'') - d(x,x_i')] \leq \gamma/2 \right\} \leq e^{-n\gamma^2/8B^2}.$$

Let $\alpha_i = \dfrac{w_1(x_i')\ w_{-1}(x_i'')}{\sum_j w_1(x_j')\ w_{-1}(x_j'')}$. We have:

$$P\left\{ y\sum_{i=1}^{n} \alpha_i\ \operatorname{sgn}[d(x,x_i'') - d(x,x_i')] \leq \frac{n\gamma/2}{\sum_{j=1}^{n} w_1(x_j')w_{-1}(x_j'')} \right\} \leq e^{-n\gamma^2/8B^2}.$$

[2] $w_1(x)$ and $w_{-1}(x)$ are called the *importance* in the context of covariate shift adaptation (Shimodaira, 2000; Sugiyama, Krauledat, & Müller, 2007).

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

Since $w_1(x_i')w_{-1}(x_i'') \le B$, we obtain

$$P\left\{y \cdot \sum_{i=1}^{n} \alpha_i \operatorname{sgn}[d(x, x_i'') - d(x, x_i')] \le \gamma/2B\right\} \le e^{-n\gamma^2/8B^2}.$$

According to the previous inequality, taking expectation over all $z \in M$ and then using the Markov inequality as in the proof of theorem 1, we complete the proof.

This theorem suggests a simple algorithm for learning with a $(\epsilon, \gamma, B)$-good dissimilarity function $d$. First, draw a set $S_1$ that contains $n = O(\frac{B^2}{\gamma^2} \ln 1/\delta)$ pairs of examples $((x_i', 1), (x_i'', -1))$, $i = 1, 2, \ldots, n$, and then construct $n$ *base* classifiers

$$h_i(x) = \operatorname{sgn}[d(x, x_i'') - d(x, x_i')].$$

It is guaranteed that with probability $1 - \delta$, there exists a low-error and large-margin classifier, which is a convex combination of these $h_i(x)$. Boosting would be natural for learning this large-margin voting classifier. Thus, one draws an additional set of examples $S_2$, uses boosting to learn the combination coefficients $\alpha_i$, and obtains the final classifier,

$$H(x) = \operatorname{sgn}\left[\sum_{i=1}^{n} \alpha_i h_i(x)\right].$$

In order that the final classifier $H(x)$ has an error rate at most $\epsilon + \epsilon_1$ with probability at least $1 - 2\delta$, the size of the second training set $S_2$ can be set as $\tilde{O}(\frac{B^2}{\epsilon_1^2\gamma^2})$ according to the margin bound for convex combination classifiers (Schapire, Freund, Bartlett, & Lee, 1998; Wang, Sugiyama, Yang, Zhou, & Feng, 2008).[3] The total number of examples needed to achieve such a learning guarantee is $\tilde{O}(\frac{B^2}{\gamma^2\epsilon_1^2})$ if we set $\delta = \epsilon_1$.

*2.2.3 Generalized* $(\epsilon, \gamma, B)$*-Goodness.* In this section we present a generalization of the $(\epsilon, \gamma, B)$-goodness. Recall in definition 2 that for a $(\epsilon, \gamma, B)$-good dissimilarity function $d$, most examples $z = (x, y)$ satisfy

$$\tilde{P}(d(x, x') - d(x, x'') < 0 \mid z,\ y' = y,\ y'' = -y) \ge 1/2 + \gamma/2,$$

where $\tilde{P}$ is the probability with respect to two (unknown) pdfs $\tilde{p}(x' \mid y')$ and $\tilde{p}(x'' \mid y'')$. A broader class of dissimilarity functions would be that

[3] In $\tilde{O}$, we hide the negligible logarithm terms of the learning parameters.

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

most examples $z = (x, y)$ satisfy

$$\tilde{P}(d(x, x') - d(x, x'') < v \mid z,\ y' = y,\ y'' = -y) \geq 1/2 + \gamma/2$$

for some threshold $v$, which may depend on the example pair $(x', x'')$.

**Definition 3.** *Denote by $p(x \mid y = 1)$ and $p(x \mid y = -1)$ the conditional pdfs of the learning problem. A dissimilarity function $d$ is said to be generalized $(\epsilon, \gamma, B)$-good for the learning problem if:*

1. *There exist two conditional pdfs $\tilde{p}(x \mid y = 1)$ and $\tilde{p}(x \mid y = -1)$ such that for all $x \in X$,*

$$\frac{\tilde{p}(x \mid y = 1)}{p(x \mid y = 1)} \leq \sqrt{B}, \qquad \frac{\tilde{p}(x \mid y = -1)}{p(x \mid y = -1)} \leq \sqrt{B}.$$

2. *There exists a threshold $v(x', x'')$ such that at least $1 - \epsilon$ probability mass of examples $z$ satisfies*

$$\tilde{P}(d(x, x') - d(x, x'') < v(x', x'') \mid y' = y, y'' = -y) \geq 1/2 + \gamma/2, \tag{2.3}$$

*where $\tilde{P}$ is the probability with respect to $\tilde{p}(x' \mid y')$ and $\tilde{p}(x'' \mid y'')$.*

The learning guarantee of the generalized $(\epsilon, \gamma, B)$-good dissimilarity functions is the same as that of the $(\epsilon, \gamma, B)$-good dissimilarities if the threshold is known.

**Theorem 3.** *Let $d$ be a generalized $(\epsilon, \gamma, B)$-good dissimilarity function. Assume that the threshold $v(x', x'')$ is known. Then with probability at least $1 - \delta$ over the choice of $n = 16B^2/\gamma^2\ ln(1/\delta)$ pairs of examples $(z_i', z_i'')$ with labels $y_i' = 1,\ y_i'' = -1,\ i = 1, 2, \ldots, n$, there exists a convex combination classifier $f(x)$ of $n$ base classifiers $h_i(x)$:*

$$f(x) = \sum_{i=1}^{n} \alpha_i h_i(x), \qquad \sum \alpha_i = 1, \qquad \alpha_i \geq 0,$$

*where*

$$h_i(x) = sgn[d(x, x_i'') - d(x, x_i') + v(x', x'')],$$

*such that the error rate of the combined classifier at margin $\gamma/2B$ is at most $\epsilon + \delta$.*

**Proof.** The proof follows the proof of theorem 2 by replacing $\text{sgn}[d(x, x'') - d(x, x')]$ with $\text{sgn}[d(x, x'') - d(x, x') + v(x', x'')]$.

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

A generalized $(\epsilon, \gamma, B)$-good dissimilarity function guarantees efficient learning if the threshold is given. In practice, it is difficult to know the threshold a priori. However, one can learn the thresholds and the linear coefficients simultaneously in the boosting framework. We draw a set $S_1$ of $n$ pairs of examples $((x', 1), (x'', -1))$ $i = 1, 2, \ldots, n$, and a training set $S_2$. Then we use boosting to learn the thresholds $v(x', x'')$ and the coefficients $\alpha_i$ so that the combined classifier,

$$f(x) = \sum_{i=1}^{n} \alpha_i h_i(x), \quad \sum \alpha_i = 1, \quad \alpha_i \geq 0,$$
$$h_i(x) = \text{sgn}[d(x, x_i'') - d(x, x_i') + v(x', x'')]$$

has low error and large margin on the training set $S_2$. We describe this algorithm in detail and make it more practical in section 3.

*2.2.4 $(\epsilon, \gamma, B, \eta)$-Goodness.* We propose a further generalization of the previous goodness definitions by weakening the first condition in definition 3. Here we do not require uniform upper bounds of $\frac{\tilde{p}(x|y)}{p(x|y)}$. We only need that $\frac{\tilde{p}(x|y)}{p(x|y)} \leq B$ for most examples. This new definition of goodness can be applied to a much broader class of dissimilarity functions and contains all previous goodness notions as special cases.

**Definition 4.** *Denote by $p(x \mid y = 1)$ and $p(x \mid y = -1)$ the conditional pdfs of the learning problem. A dissimilarity function $d$ is said to be $(\epsilon, \gamma, B, \eta)$-good for the learning problem if:*

1. *There exist two conditional pdfs $\tilde{p}(x \mid y = 1)$ and $\tilde{p}(x \mid y = -1)$ such that*
$$\frac{\tilde{p}(x \mid y = 1)}{p(x \mid y = 1)} \leq \sqrt{B}, \qquad \frac{\tilde{p}(x \mid y = -1)}{p(x \mid y = -1)} \leq \sqrt{B}$$
*hold for at least $1 - \eta$ probability mass of examples $(x, y)$.*
2. *There exists a threshold $v(x', x'')$ such that at least $1 - \epsilon$ probability mass of examples $z$ satisfies*
$$\tilde{P}(d(x, x') - d(x, x'') < v(x', x'') \mid y' = y, y'' = -y) \geq 1/2 + \gamma/2, \tag{2.4}$$
*where $\tilde{P}$ is the probability with respect to $\tilde{p}(x' \mid y')$ and $\tilde{p}(x'' \mid y'')$.*

The learning guarantee of a $(\epsilon, \gamma, B, \eta)$-good dissimilarity function is the same as the $(\epsilon, \gamma, B)$-good dissimilarity only up to a constant factor.

**Theorem 4.** *If $d$ is a $(\epsilon, \gamma, B, \eta)$-good dissimilarity function, then with probability at least $1 - \delta$ over the choice of $n = 16B^2/\gamma^2 \, ln(2/\delta)$ pairs of examples $(z_i', z_i'')$ with labels $y_i' = 1$, $y_i'' = -1$, $i = 1, 2, \ldots, n$, there exists a convex combination*

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

*classifier $f(x)$ of $n$ base classifiers $h_i(x)$:*

$$f(x) = \sum_{i=1}^{n} \alpha_i h_i(x), \quad \sum \alpha_i = 1, \quad \alpha_i \geq 0,$$

*where*

$$h_i(x) = sgn[d(x, x_i'') - d(x, x_i') + v(x_i', x_i'')],$$

*such that the error rate of the combined classifier at margin $\gamma/2B$ is at most $\epsilon + \delta$, provided $\eta \leq \frac{\gamma^2 \delta}{64B^2 \cdot \ln(2/\delta)}$ and the threshold is known.*

**Proof.** The proof is almost the same as theorem 3. The only difference is that by our assumption, when choosing $n = 16B^2/\gamma^2 \ln(2/\delta)$ pairs of examples, the probability that there contains a "bad" example is at most $\delta/2$, where a bad example $z' = (x', y')$ or $z'' = (x'', y'')$ means that $\frac{\tilde{p}(x'|y'=1)}{p(x'|y'=1)} > \sqrt{B}$ or $\frac{\tilde{p}(x''|y''=-1)}{p(x''|y''=-1)} > \sqrt{B}$, respectively.

**2.3 Discussions of the Sufficient Conditions.** In this section we compare our results with existing theory on learning with (dis)similarity functions and study possible extension of the proposed sufficient conditions.

*2.3.1 Comparison to Previous Theory on Learning with Similarity Functions.* We first point out that our results of dissimilarity functions can be easily extended to similarity functions. Let $s(x, x')$ denote a similarity function. Replacing $d(x, x') < d(x, x'')$ by $s(x, x') > s(x, x'')$ in all definitions and theorems gives the theory for similarity functions. Therefore, the theory is a unified framework for learning with similarity and dissimilarity functions.

In Balcan and Blum's (2006) theory, a similarity function $s$ is said to be $(\epsilon, \gamma)$-good for a (deterministic label) learning problem if there exists a weighting function $w(x) \in [0, 1]$, such that at least $1 - \epsilon$ probability mass of examples $x$ satisfies

$$E[w(x')s(x, x') \mid y' = y] \geq E[w(x')s(x, x') \mid y' = -y] + \gamma.$$

For a $(\epsilon, \gamma)$-good similarity function, there is a simple learning approach. First, draw a set of examples $\{x_1, x_2, \ldots, x_n\}$. Then construct an $n$-dimensional feature vector of each object $x$ as

$$(s(x, x_1), s(x, x_2), \ldots, s(x, x_n)).$$

It can be shown with high probability that there is a large-margin, low-error linear separator in the $n$-dimensional feature space, so linear SVM can be used to learn the classifier in the feature space with a new set of examples.

On the practical side, Balcan and Blum's theory implies an SVM-type algorithm, while ours suggests a boosting-type algorithm. On the theoretical

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

side, their notion of good similarity functions and ours are different sufficient conditions for learning with (dis)similarity functions. That is, neither is a subset of the other, as described in the following proposition:

**Proposition 1.**

1. *For every $\gamma$ and $B$, there is a similarity function $s(\cdot,\cdot)$ and a learning problem $P$, such that $s(\cdot,\cdot)$ is $(0,\gamma,B)$-good for $P$ in our sense, but not $(0,\gamma/B)$-good in Balcan and Blum's sense.*
2. *For every $\gamma$, there is a similarity function $s(\cdot,\cdot)$ and a learning problem $P$, such that $s(\cdot,\cdot)$ is $(0,\gamma)$-good for $P$ in Balcan and Blum's sense, but not $(0,B\gamma,B)$-good in our sense for any $B$.*

Recently, Balcan, Blum, and Srebro (2008a) proposed an improved sufficient condition for learning with similarity functions: $s$ is said to be a $(\epsilon,\gamma,\tau)$-good similarity function for a learning problem $P$ if there is a random indicator variable $R$ (depending on $x'$) such that $E[yy's(x,x') \mid R] \geq \gamma$ and $\Pr(R) \geq \tau$. Here $R$ can be understood as a (stochastic) membership function on $x'$, indicating whether it is "important."

This new notion of good similarity function and our definition of goodness are still different sufficient conditions, as shown in the following proposition:

**Proposition 2.**

1. *For every $\gamma$ and $B$, there is a similarity function $s(\cdot,\cdot)$ and a learning problem $P$, such that $s(\cdot,\cdot)$ is $(0,\gamma,B)$-good for $P$ in our sense but not $(0,\gamma',\tau')$-good in the Balcan et al. improved sense for any $\gamma'$ and $\tau'$ such that $\gamma'\tau' \geq \gamma/B$.*
2. *For every $\gamma$, there is a similarity function $s(\cdot,\cdot)$ and a learning problem $P$, such that $s(\cdot,\cdot)$ is $(0,\gamma,\tau)$-good for $P$ in the Balcan et al. improved sense, but not $(0,B\gamma',B)$-good in our sense for any $B$ and any $\gamma' \geq max(\gamma,\tau)$.*

**Proof.** This is immediate from the previous proposition and the following relation between improved $(\epsilon,\gamma,\tau)$-goodness and the $(\epsilon,\gamma)$-goodness described in Balcan et al. (2008a). A $(\epsilon,\gamma,\tau)$-good similarity function is also a $(\epsilon,\gamma\tau)$-good similarity function, and a $(\epsilon,\gamma')$-good similarity function is also a $(\epsilon,\gamma,\tau)$ similarity function, where $\gamma' \geq \max(\gamma,\tau)$.

To conclude the comparison, our $(\epsilon,\gamma,B)$-goodness implies that the "order" of the (dis)similarity is important. That is, data from the same class are closer than those from different classes. But how much closer is not crucial. On the other hand, the $(\epsilon,\gamma)$-goodness and its improvement of Balcan et al. imply that the average value of the similarity is important. From a practical viewpoint, if the user has some confidence that the data from the same class are more likely to be closer to each other, our result and the suggested

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

boosting-type algorithm apply. This is especially suitable for the applications in which an appropriate scaling of the (dis)similarities is difficult or expensive. In case the (dis)similarities are well scaled so that there is a significant difference of the within-class and between-class (average) similarity, the SVM-type algorithm suggested by the $(\epsilon, \gamma)$-goodness applies.

*2.3.2 Pseudo-Good Dissimilarity Functions.* We require in our definitions of good dissimilarity functions that most of the examples (at least $1 - \epsilon$ probability mass) are more likely to be close to a random example of the same class than to an example of the opposite class. Another natural but weaker notion would be the following:

**Definition 5.** *A dissimilarity function $d(x, x')$ is said to be pseudo-$\gamma$-good for the learning problem if*

$$P\left(d(x, x') < d(x, x'') \mid y' = y, y'' = -y\right) \geqslant 1/2 + \gamma/2, \tag{2.5}$$

*where the probability is taken over the random examples $z = (x, y)$, $z' = (x', y')$, and $z'' = (x'', y'')$.*

Note that the pseudo-$\gamma$-goodness is a weaker notion than strong $(\epsilon, \gamma)$-goodness, as described in the following proposition:

**Proposition 3.** *If a dissimilarity function is strongly $(\epsilon, \gamma)$-good for a learning problem, then it is also pseudo-$\gamma'$-good, where $\gamma' = (1 - \epsilon)\gamma - \epsilon$ (if $\gamma' \geq 0$).*

**Proof.** The proposition is immediate by noting that the left-hand side of equation 2.5 is the expectation of the left-hand side of equation 2.1 over $z = (x, y)$.

One might expect that the pseudo-goodness would also imply learnability, possibly in a weak sense. However, the next proposition shows that the majority voting scheme given in previous theorems in general does not guarantee any learnability, even in the weakest sense. This result means that our sufficient conditions for efficient learning with dissimilarity functions may not be weakened too much.

**Proposition 4.** *For any pseudo-$\gamma$-good ($0 < \gamma < 1/2$) dissimilarity function $d$, there exists a learning problem such that even if we have infinitely many pairs of training examples $(z_i', z_i'')$ with labels $y_i' = 1$, $y_i'' = -1$, $i = 1, 2, \ldots, n$, the error rate of the voting classifier is higher than $1/2$, that is,*

$$P\left(y \cdot f(x) < 0\right) > 1/2,$$

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

*where*

$$f(x) = \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} sgn\left[d(x, x_i'') - d(x, x_i')\right].$$

**Proof.** For a fixed example $z = (x, y)$, when $n \to \infty$, the law of large numbers gives that $f(x)$ converges in probability to

$$E\left\{sgn[d(x, x'') - d(x, x')] \mid z\,, y' = 1, y'' = -1\right\}.$$

Note further that

$$\begin{aligned} &E\{sgn[d(x, x'') - d(x, x')] \mid z\,, y' = 1, y'' = -1\} \\ &\quad = 2P\{d(x, x') < d(x, x'') \mid z\,, y' = 1, y'' = -1\} - 1. \end{aligned}$$

An error occurs, $yf(x) < 0$, if

$$P\{d(x, x') < d(x, x'') \mid z, y' = y, y'' = -y\} < 1/2.$$

Denote

$$g(z) = P(d(x, x') < d(x, x'') \mid z, y' = y, y'' = -y).$$

For $0 < \gamma < 1/2$, it is easy to construct a distribution such that the following two inequalities hold simultaneously:[4]

$$\begin{aligned} &E[g(z)] \geq 1/2 + \gamma/2, \\ &P(g(z) < 1/2) > 1/2. \end{aligned}$$

The first inequality is equivalent to equation 2.3, meaning that the dissimilarity function is pseudo-$\gamma$-good for the problem. The second inequality implies that the error rate of the voting classifier is larger than $1/2$.

## 3 The DBoost Algorithm

In this section, we slightly modify the algorithm suggested by our theory to make it more suitable for practical use. The proposed algorithm is essentially a dissimilarity-based boosting and will be referred to as DBoost.

Recall that under the theoretical guidance, to learn with a dissimilarity function $d$, one needs to draw two sets of examples. One set contains $n$ pairs

[4] For example: $P\,(g(z) = 1/2 - \delta) = 1/2 + \delta$, $P\,(g(z) = 1) = 1/2 - \delta$, for some small $\delta$.

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

of examples $((x_i', 1), (x_i'', -1))$, with which we construct the base classifiers

$$h_i(x) = \mathrm{sgn}\left[d(x, x_i'') - d(x, x_i') + v_i\right], \qquad i = 1, 2, \ldots, n.$$

The other set of examples is used as training data for boosting to learn the thresholds $v_i$ and the combination coefficients $\alpha_i$ so that the voting classifier

$$f(x) = \sum_{i=1}^{n} \alpha_i h_i(x), \qquad \sum \alpha_i = 1, \qquad \alpha_i \geq 0$$

has a low error and large margin on the training set.

In practice however, the users often have only one fixed set of examples. So in the DBoost algorithm, we try to make use of the data efficiently. Denote by $S$ the set of data the user has. The DBoost algorithm first constructs the pairs $(x_i', 1), (x_i'', -1)$ by considering all possible pairs of examples with different labels in $S$. Then $S$ is also served as the training set for boosting to learn the final large-margin convex-combination classifier.

In the DBoost algorithm, we choose AdaBoost as the booster due to its good ability to generate large-margin classifiers (Freund & Schapire, 1996). The thresholds $v_i$ and the coefficients $\alpha_i$ are learned by AdaBoost in a series of rounds. At the $i$th round, we have a distribution $D_i$ over the training set $S$. The algorithm then searches for the example pair $(x_i', x_i'')$ and the threshold $v_i$ so that the base classifier,

$$h_i(x) = \mathrm{sgn}[d(x, x_i'') - d(x, x_i') + v_i],$$

has the minimum training error on $S$ with respect to the distribution $D_i$. The (unnormalized) coefficient $\alpha_i$ is determined by this training error, and the distribution $D_i$ is also updated accordingly (see Figure 3 for details). After $T$ rounds, the DBoost outputs the final classifier,

$$H(x) = \mathrm{sgn}[f(x)],$$

where

$$f(x) = \sum_{i=1}^{T} \alpha_i h_i(x).$$

One difficulty in the above procedure is that at each round, searching for the best example pair $(x_i', x_i'')$ over all possible pairs is computationally expensive. We solve this problem in DBoost by searching for the best pair not over the whole set of possible pairs, but over only a small number $M$ of example pairs randomly selected at each round. The selection of these example pairs is according to the current distribution $D_i$. It is well known

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

**Input**: Dissimilarity function $d(\cdot, \cdot)$

Training set $S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_l, y_l)\}$

Distribution $D$ on the training examples

**Initialization:** $err^* = \infty$

**for** $m = 1$ **to** $M$ **do**

1. Randomly select two examples $x_+^m, x_-^m$ according to $D$, and construct the pair.

2. Find $v^m$ so that the classifier

$$h^{(m)}(x) = \text{sgn}\left[d(x, x_+^m) - d(x, x_-^m) + v^m\right]$$

has the minimum error rate $err^m$, where $err^m = \sum_{i=1}^{l} D(i)\text{I}\left[y_i \neq h^{(m)}(x_i)\right]$, and I is the indicator function.

3. **if** $err^m < err^*$,

$$err^* = err^m, x_+^* = x_+^m, x_-^* = x_-^m, v^* = v^m$$

**endif**

**end**

**Output**: The base classifier

$$h(x) = \text{sgn}\left[d(x, x_+^*) - d(x, x_-^*) + v^*\right].$$

Figure 2: The subroutine: BaseLearn.

that as the round $i$ increases, boosting makes the distribution $D_i$ put larger weights on the examples that are harder to classify. Therefore, the example pair would be two data near the classification boundary since they are the most difficult examples.

The DBoost algorithm is described in detail in Figures 2 and 3. Figure 2 shows the subroutine of searching for the pair $(x', x'')$ and the threshold $v$ to construct the base classifier at a certain round of boosting. Figure 3

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

**Input**: Dissimilarity function $d(\cdot, \cdot)$

$S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_l, y_l)\}$

where $x_i \in X$, $y_i \in \{-1, 1\}$.

**Initialization:** $D_1(i) = 1/l$.

**for** $t = 1$ **to** $T$ **do**

1. Call BaseLearn$(d, S, D_t)$; The returned classifier is denoted as $h_t$.

2. Let

$$\alpha_t = \frac{1}{2} \ln \frac{1 + \gamma_t}{1 - \gamma_t},$$

where $\gamma_t = \sum_{i=1}^{l} D_t(i) y_i h_t(x_i)$.

3. Update:

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t},$$

where $Z_t$ is a normalization factor chosen so that $D_{t+1}$ will be a distribution.

**end**

**Output**: The final Classifier

$$H(x) = \operatorname{sgn}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right).$$

Figure 3: The main DBoost algorithm.

describes the boosting framework in DBoost, which is essentially the AdaBoost algorithm. Here we show the original version of the AdaBoost given by Freund and Schapire (1996). An improved version, RealBoost (Schapire & Singer, 1999), can often achieve better performance. RealBoost folds the coefficient $\alpha_t$ into the base classifier $h_t$, and hence $h_t$ outputs a real number. (For details, refer to Schapire & Singer, 1999.)

## 4 Experiments on Learning with Dissimilarity Functions

In this section, we perform experiments on algorithms that learn with dissimilarity functions. We compare DBoost to the nearest-neighbor rule and

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

the algorithm based on Balcan and Blum's theory mentioned in section 1. The algorithms will be denoted by NN and LSVM for short, respectively. The details of the implementation are described below:

1. **DBoost:** For DBoost, there are two parameters to be specified. One is $M$ in Figure 2, that is, the number of example pairs forming the search space. As discussed earlier, $M$ is introduced to reduce the computational cost. We found that the value of $M$ has no significant effect on the performance unless $M$ is too small. So in all the experiments, we simply set $M = 100$. The other parameter is $T$ in Figure 3: the number of base classifiers generated in boosting. We set $T = 1000$ in all the experiments.
2. **NN:** We use the one-nearest neighbor (1NN) classifier.
3. **LSVM:** This approach is based on Balcan and Blum's theory of learning with similarity functions. Given a similarity function $s$, we choose a set of prototypes $\{p_1, p_2, \ldots, p_r\}$. For each training data $x$, calculate the similarity of $x$ and the prototypes, and then construct the vector $(s(x, p_1), s(x, p_2), \ldots, s(x, p_r))$. This vector is treated as the feature representation of $x$. The final step is to run linear SVM on this $r$-dimensional space to obtain the classifier. As mentioned earlier, to have a learning guarantee for this algorithm, the similarity should be a normalized function. That is, $|s| \leq 1$. So we need to transform the dissimilarity $d$ to a normalized $s$. We consider

   $$s = e^{-d^2/\sigma}.$$

   The value of the parameter $\sigma$ is tuned by cross-validation on the training set. In our implementation, we randomly select 20% of the training examples as prototypes, and we run libsvm (Chang & Lin, 2001) to obtain the linear SVM classifier.

In the first set of experiments, we study image classification. For some cases, it is more convenient to directly define dissimilarities between images than to construct meaningful features. Many dissimilarity measures of images have been proposed in the literature. We adopt in this experiment three measures: the tangent distance (Simard et al., 1993), the fuzzy image metric (Li et al., 2002), and the Euclidean distance. We perform the experiments on the USPS database, which consists of images of handwritten digits. The data set has been partitioned into a fixed training set and a test set, consisting of 7291 and 2007 examples, respectively.

Figure 4 shows the results of the three algorithms with the three dissimilarity measures, respectively. By using the tangent distance, NN achieves the best performance. In fact, the tangent distance is developed specifically for the handwritten-digit classification problem. It incorporates strong domain knowledge and is invariant to local within-class transformations (Simard et al., 1993). Therefore the tangent distance determines a good local topology for handwritten digit images. If two images have a very small tangent distance, they are highly likely to be in the same class.

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

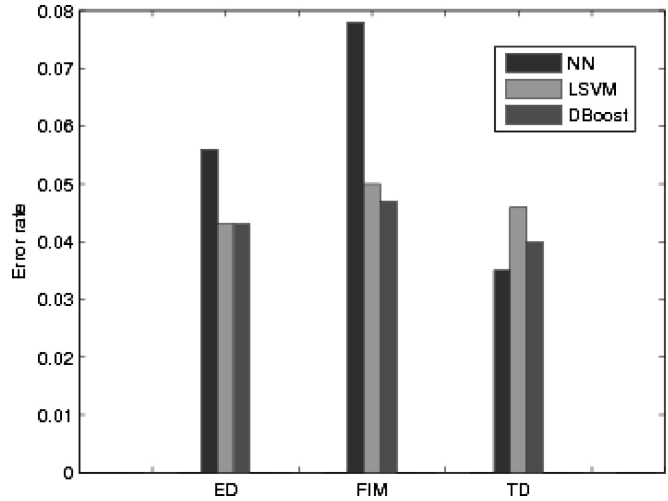


Figure 4: Comparison of the algorithms on the USPS data set using three dissimilarity functions: Euclidean distance (ED), fuzzy image metric (FIM), and tangent distance (TD).

The other two dissimilarities—the fuzzy image metric and the Euclidean distance—are more general-purpose measures. The information of the local distance alone is not enough for an accurate classification in this handwritten digit problem. So the performance of NN with these two dissimilarities is not excellent. On the other hand, DBoost works well in these cases. This result implies that the fuzzy image metric and the Euclidean distance are still good dissimilarity functions. In other words, they contain enough information to build accurate classifiers.

We then consider the dissimilarity with discrete (qualitative) values. For instance, when people make a subjective evaluation of the similarity between images, only qualitative (discrete) values can be given. For example, *similar*, *average*, and *dissimilar* are possible values of a three-level qualitative dissimilarity. We evaluate the algorithms on such qualitative measures. To conduct the experiments, Euclidean distances are quantized to 3 to 15, levels respectively. The results on the USPS data sets are shown in Figure 5: even with the three-level measures, for which the information of local topology is mostly lost, DBoost still has a low error rate.

We also evaluate the performance of the algorithms against noisy data. We add to the USPS images gaussian white noise with different variances and feed the Euclidean distance to the algorithm as input. The results are depicted in Figure 6, showing that DBoost is the most robust to noise.

In the second set of experiments, we evaluate the algorithms on a variety of domains. We adopt 22 benchmark data sets from the UCI repository

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

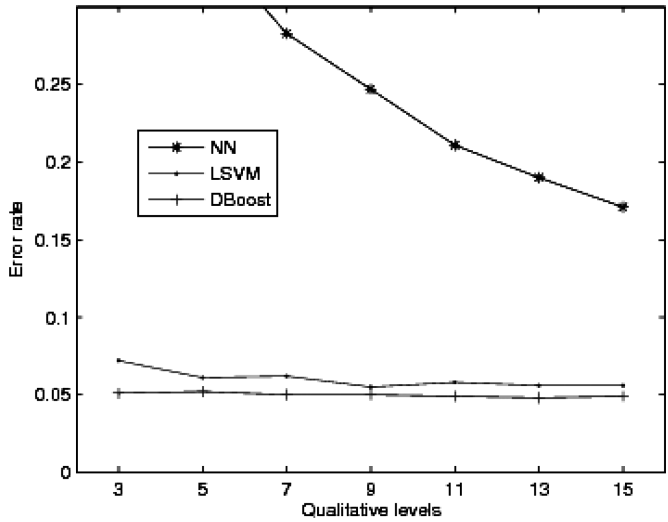


Figure 5: Comparison of the algorithms using quantized dissimilarity functions on the USPS database.

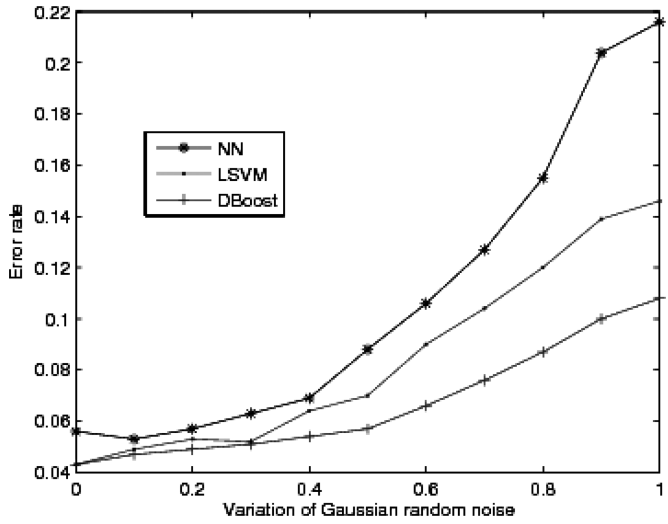


Figure 6: Comparison of the algorithms on the USPS database with noise.

(Asuncion & Newman, 2007). The aim is to see whether the DBoost algorithm works well for dissimilarity functions that are widely used in practice. Here we consider two dissimilarity measures: $l_1$ and $l_\infty$. In the experiments,

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

Table 1: Description of the 22 Data Sets from UCI Repository.

| Data Set | Number of Classes | Number of Examples | Data Set | Number of Classes | Number of Examples |
|---|---|---|---|---|---|
| Balance | 3 | 625 | Letter | 26 | 20,000 |
| Breast | 2 | 699 | Liver | 2 | 345 |
| Cleveland | 2 | 297 | Monk1 | 2 | 556 |
| Diabetes | 2 | 768 | Monk2 | 2 | 601 |
| Echo | 2 | 106 | Monk3 | 2 | 554 |
| German | 2 | 1000 | Satimage | 6 | 6435 |
| Hays | 3 | 160 | Vehicle | 4 | 846 |
| Hepatitis | 2 | 155 | Vowel | 11 | 990 |
| Image | 7 | 2310 | Wdbc | 2 | 569 |
| Ionosphere | 2 | 351 | Wine | 3 | 178 |
| Iris | 3 | 150 | Wpbc | 2 | 194 |

each data set is used in a fivefold cross-validation fashion. The data sets are described in Table 1.

The results are listed in Table 2. For each data set, the algorithm that has the best performance and those that are comparable to the best according to the t-test at the significant level 0.01 are marked in boldface. From these results, one can see that on the whole, DBoost has the best performance with these two dissimilarities, even though they are not good measures for the NN classifier.

## 5 Experiments on Learning with Similarities

In this section, we provide experimental results of algorithms on learning with a similarity function. As pointed out in section 2.3, our theory and algorithm can be applied to similarities with only minor modifications. To be specific, in order to use the DBoost algorithm for a similarity function $s(x', x'')$, we need to change only the base classifier,

$$h(x) = \operatorname{sgn}\big[d(x, x_+^*) - d(x, x_-^*) + v^*\big],$$

in Figure 2 to

$$h(x) = \operatorname{sgn}\big[s(x, x_-^*) - s(x, x_+^*) - v^*\big].$$

We run experiments in the similarity setting with a music classification task. The goal is to evaluate the algorithms with a practical similarity measure. The data consist of 50 Japanese pop songs and 50 Japanese traditional songs (used in a fivefold cross-validation fashion). Each song is represented in the MIDI format, which contains a set of sequences of musical notes, cues, tones, volumes, and so on. We extract the main melody from each song and

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

Table 2: Comparison of the Performances of the Six Algorithms on UCI Data Sets.

| | $l_1$ | | | $l_\infty$ | | |
|---|---|---|---|---|---|---|
| Data Set | NN | LSVM | DBoost | NN | LSVM | DBoost |
| Balance | $20.2 \pm 1.4$ | $7.8 \pm 2.9$ | $\mathbf{5.6 \pm 2.3}$ | $24.0 \pm 3.9$ | $\mathbf{6.1 \pm 1.7}$ | $\mathbf{5.3 \pm 2.6}$ |
| Breast | $\mathbf{3.2 \pm 1.7}$ | $\mathbf{2.5 \pm 1.7}$ | $\mathbf{2.9 \pm 1.8}$ | $5.0 \pm 1.9$ | $\mathbf{3.1 \pm 2.1}$ | $\mathbf{3.4 \pm 2.6}$ |
| Cleveland | $22.6 \pm 5.4$ | $\mathbf{17.8 \pm 6.0}$ | $20.5 \pm 5.5$ | $\mathbf{26.2 \pm 6.0}$ | $\mathbf{27.0 \pm 4.3}$ | $\mathbf{24.9 \pm 8.5}$ |
| Diabetes | $29.3 \pm 1.7$ | $\mathbf{23.8 \pm 3.2}$ | $26.4 \pm 2.8$ | $29.0 \pm 1.5$ | $\mathbf{22.9 \pm 2.8}$ | $27.2 \pm 1.3$ |
| Echo | $21.7 \pm 7.3$ | $16.9 \pm 3.8$ | $\mathbf{12.3 \pm 2.7}$ | $\mathbf{22.6 \pm 7.8}$ | $\mathbf{16.9 \pm 3.8}$ | $\mathbf{17.9 \pm 3.9}$ |
| German | $30.4 \pm 2.9$ | $\mathbf{26.7 \pm 2.2}$ | $\mathbf{27.3 \pm 2.2}$ | $33.1 \pm 2.9$ | $\mathbf{30.1 \pm 1.8}$ | $\mathbf{29.5 \pm 3.4}$ |
| Hayes | $27.5 \pm 7.4$ | $\mathbf{16.2 \pm 3.4}$ | $18.7 \pm 3.2$ | $32.5 \pm 6.8$ | $\mathbf{26.2 \pm 4.7}$ | $\mathbf{21.3 \pm 4.6}$ |
| Hepatitis | $\mathbf{21.3 \pm 9.3}$ | $\mathbf{18.4 \pm 3.3}$ | $\mathbf{17.8 \pm 9.6}$ | $\mathbf{19.1 \pm 8.8}$ | $\mathbf{19.1 \pm 6.6}$ | $\mathbf{19.1 \pm 8.8}$ |
| Image | $2.1 \pm 0.6$ | $3.4 \pm 1.0$ | $\mathbf{1.6 \pm 0.4}$ | $\mathbf{3.3 \pm 1.0}$ | $3.9 \pm 1.0$ | $\mathbf{2.7 \pm 0.6}$ |
| Ionosphere | $10.8 \pm 2.8$ | $\mathbf{6.8 \pm 2.7}$ | $\mathbf{5.4 \pm 2.9}$ | $13.1 \pm 1.6$ | $\mathbf{7.7 \pm 2.9}$ | $\mathbf{5.7 \pm 3.9}$ |
| Iris | $\mathbf{7.3 \pm 4.3}$ | $\mathbf{4.0 \pm 3.7}$ | $\mathbf{6.7 \pm 3.4}$ | $\mathbf{4.7 \pm 3.8}$ | $\mathbf{2.7 \pm 2.8}$ | $\mathbf{4.0 \pm 3.7}$ |
| Letter | $4.7 \pm 0.3$ | $\mathbf{4.2 \pm 0.2}$ | $5.0 \pm 0.1$ | $8.8 \pm 0.5$ | $\mathbf{4.1 \pm 0.4}$ | $6.3 \pm 0.3$ |
| Liver | $40.3 \pm 6.7$ | $\mathbf{31.6 \pm 4.7}$ | $\mathbf{30.4 \pm 6.4}$ | $41.4 \pm 3.8$ | $37.4 \pm 4.0$ | $\mathbf{31.2 \pm 5.7}$ |
| Monk1 | $20.3 \pm 4.6$ | $8.6 \pm 3.7$ | $\mathbf{0.0 \pm 0.0}$ | $20.1 \pm 4.9$ | $19.0 \pm 2.8$ | $\mathbf{1.8 \pm 1.4}$ |
| Monk2 | $14.0 \pm 3.4$ | $24.1 \pm 2.5$ | $\mathbf{1.7 \pm 1.8}$ | $14.0 \pm 3.3$ | $27.8 \pm 3.9$ | $\mathbf{4.5 \pm 2.8}$ |
| Monk3 | $19.1 \pm 9.1$ | $3.6 \pm 0.9$ | $\mathbf{2.3 \pm 0.8}$ | $24.5 \pm 1.4$ | $6.5 \pm 2.0$ | $\mathbf{4.1 \pm 1.0}$ |
| Satimage | $9.4 \pm 1.0$ | $8.8 \pm 0.9$ | $\mathbf{8.0 \pm 0.9}$ | $12.1 \pm 1.6$ | $\mathbf{9.6 \pm 1.0}$ | $\mathbf{9.8 \pm 0.8}$ |
| Vehicle | $30.9 \pm 2.4$ | $\mathbf{28.3 \pm 0.9}$ | $\mathbf{27.0 \pm 2.4}$ | $32.6 \pm 4.9$ | $28.8 \pm 1.6$ | $\mathbf{22.6 \pm 2.1}$ |
| Vowel | $\mathbf{2.0 \pm 1.2}$ | $8.5 \pm 2.0$ | $\mathbf{2.7 \pm 1.5}$ | $\mathbf{2.6 \pm 1.0}$ | $7.7 \pm 2.4$ | $\mathbf{3.6 \pm 1.4}$ |
| Wdbc | $4.2 \pm 1.4$ | $3.5 \pm 0.6$ | $\mathbf{2.3 \pm 0.8}$ | $\mathbf{6.1 \pm 1.4}$ | $\mathbf{5.4 \pm 1.4}$ | $\mathbf{5.8 \pm 2.5}$ |
| Wine | $\mathbf{4.5 \pm 1.5}$ | $\mathbf{2.2 \pm 2.3}$ | $\mathbf{3.9 \pm 3.2}$ | $6.2 \pm 3.7$ | $\mathbf{1.7 \pm 1.5}$ | $\mathbf{2.8 \pm 2.8}$ |
| Wpbc | $31.4 \pm 3.8$ | $\mathbf{23.7 \pm 5.7}$ | $\mathbf{22.7 \pm 4.1}$ | $37.1 \pm 5.1$ | $\mathbf{23.7 \pm 5.7}$ | $\mathbf{26.3 \pm 5.0}$ |

convert it to a string, where each character corresponds to a sixteenth note or a sixteenth rest.

The similarity measure we use for a pair of songs is the length of the longest common subsequence (LCS). The LCS of a string $s$ and a string $t$ is the longest (possibly nonconsecutive) sequence of characters, which appear in both $s$ and $t$. For example, given two strings $s = aabbcc$ and $t = abca$, the LCS of $s$ and $t$ is $abc$. The length of LCS is a natural measure to capture the similarity between two melodies since similar songs would share longer subsequences.

However, the similarity matrix based on the length of LCS is, in general, not positive semidefinite (Hagio, 2006). For example, consider the set of strings $\{a, b, ab, ba\}$. It is easy to check that the LCS-based similarity matrix is

$$\begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix},$$

which has a negative eigenvalue.

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

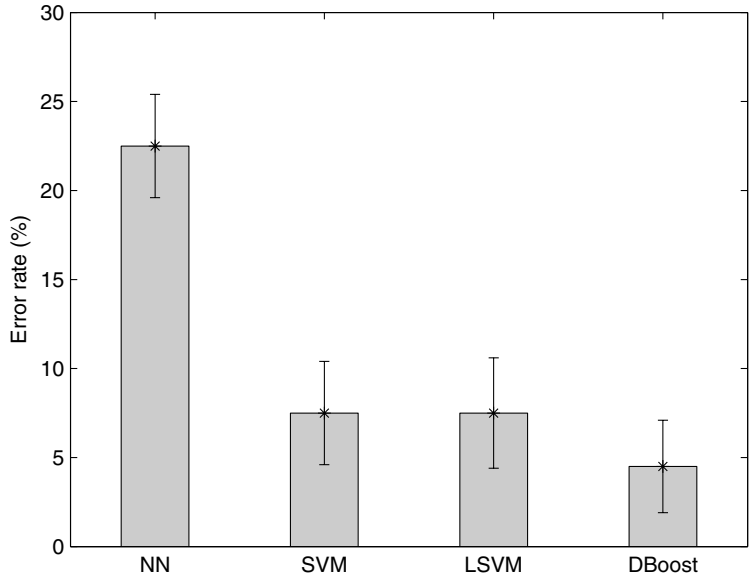


Figure 7: Results on music similarity data.

We evaluate four algorithms in this experiment. Besides NN, LSVM, and DBoost, we also consider ordinary SVM for this nonpositive similarity kernel. We adopt a common approach, which adds to the similarity (kernel) matrix a positive diagonal to enforce it to be positive definite. Figure 7 shows the performances of four algorithms. These results demonstrate that DBoost is promising for this simple and intuitive similarity measure.

## 6 Conclusion

In this work, we gave sufficient conditions for dissimilarity functions to allow one to learn well. We showed that if most examples are more likely to be close to a randomly selected example of the same class than to a random example of the other class, there is a simple algorithm that can learn well with the dissimilarity measure. The random selection of the examples could be according to arbitrary probability distributions satisfying a mild condition. Therefore, the sufficient condition captures a large class of dissimilarity functions. We also developed a more practical algorithm, named DBoost, under the theoretical guidance. DBoost learns a large-margin convex-combination classifier of a set of base classifiers, each of which depends on only the dissimilarities. Experimental results demonstrate that DBoost has good performance with several dissimilarity measures that are widely used in practice.

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

**Acknowledgments**

We thank Masayuki Takeda for kindly providing us Japanese song data, and we also thank Kazuhito Hagio for preprocessing them. This work was supported by NSFC(60775005, 60635030) and Global COE Program of the Tokyo Institute of Technology.

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021

**References**

Asuncion, A., & Newman, D. J. (2007). UCI machine learning repository. 2007. Available online at http://archive.ics.uci.edu/ml.

Balcan, M.-F., & Blum, A. (2006). On a theory of learning with similarity functions. In W. W. Cohen & A. Moore (Eds.), *Proceedings of the International Conference on Machine Learning*. New York: ACM Press.

Balcan, M.-F., Blum, A., & Srebro, N. (2008a). Improved guarantees for learning via similarity functions. In *Proceedings of the 21st Annual Conference on Learning Theory*. Madison, WI: Omnipress.

Balcan, M.-F., Blum, A., & Srebro, N. (2008b). A theory of learning with similarity functions. *Machine Learning*, *72*, 89–112.

Balcan, M.-F., Blum, A., & Vempala, S. (2004). On kernels, margins, and low-dimensional mappings. In *Proceedings of the International Workshop on Algorithmic Learning Theory*. New York: Springer.

Balcan, M.-F., Blum, A., & Vempala, S. (2006). Kernels as features: On kernels, margins, and low-dimensional mappings. *Machine Learning*, *65*, 79–94.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees*. Belmont, CA: Wadsworth.

Chang, C. C., & Lin, C. J. (2001). *A library for support vector machines*. Available online at http://www.csie.ntu.edu.tw/∼cjlin/libsvm, 2001.

Freund, Y., & Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *Proceedings of the International Conference on Machine Learning*. San Francisco: Morgan Kaufmann.

Fukunaga, K. (1990). *Introduction to statistical pattern recognition* (2nd ed.). San Diego, CA: Academic Press.

Gärtner, T. (2003). A survey of kernels for structured data. *SIGKDD Explorations*, *5*(1), S268–S275.

Goldfarb, L. (1985). A new approach to pattern recognition. In L. N. Kannal & A. Rosenfeld (eds.), *Progress in pattern recognition* (vol. 2, pp. 241–402). Dordrecht: Elsevier.

Graepel, T., Herbrich, R., Bollmann-Sdorra, P., & Obermayer, K. (1999). Classification on pairwise proximity data. In S. A. Solla, T. K. Leen, & K.-R. Muller (Eds.), *Advances in neural information processing systems*, *12*. Cambridge, MA: MIT Press.

Hagio, K. (2006). *Design and evaluation of string similarity measure based kernels* Unpublished bachelor's thesis, Kyushu University (in Japanese).

Hart, P., & Cover, T. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*, 21–27.

Huttenlocher, D. P., Klanderman, G. A., & Rucklidge, W. J. (1993). Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *15*(9), 850–863.

Jacobs, D. W., Weinshall, D., & Gdalyahu, Y. (2000). Classification with nonmetric distances: Image retrieval and class representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(6), 583–600.

Jain, A. K., & Zongker, D. E. (1997). Representation and recognition of handwritten digits using deformable templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*(12), 1386–1391.

Li, J., Chen, G., & Chi, Z. (2002). A fuzzy image metric with application to fractal coding. *IEEE Transactions on Image Processing*, *11*(6), 636–643.

Maltoni, D., Maio, D., Jain, A. K., & Prabhakar, S. (2003). *Handbook of fingerprint recognition*. New York: Springer.

Pekalska, E., & Duin, R. P. W. (2002). Dissimilarity representations allow for building good classifiers. *Pattern Recognition Letters*, *23*(8), 943–956.

Saigo, H., Vert, J.-P., Ueda, N., & Akutsu, T. (2004). Protein homology detection using string alignment kernels. *Bioinformatics*, *11*, 1682–1689.

Schapire, R., Freund, Y., Bartlett, P., & Lee, W. (1998). Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals of Statistics*, *26*, 1651–1686.

Schapire, R. E., & Singer, Y. (1999). Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, *37*, 297–336.

Shimodaira, H. (2000). Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, *90*(2), 227–244.

Simard, P., LeCun, Y., & Denker, J. (1993). Efficient pattern recognition using a new transformation distance. In S. J. Hanson, J. D. Cowan, & C. L. Giles (Eds.), *Advances in neural information processing systems*, *5*. San Francisco: Morgan Kaufmann.

Srebro, N. (2007). How good is a kernel when used as a similarity measure? In *Proceedings of the 20th Annual Conference on Learning Theory*. New York: Springer.

Sugiyama, M., Krauledat, M., & Müller, K.-R. (2007). Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research*, *8*, 985–1005.

Vapnik, V. (1998). *Statistical learning theory*. New York: Wiley.

Wang, L., Sugiyama, M., Yang, C., Zhou, Z., & Feng, J. (2008). On the margin explanation of boosting algorithms. In *Proceedings of the 21st Annual Conference on Learning Theory*. Madison, UI: Omnipress.

Wang, L., Zhang, Y., & Feng, J. (2005). On the Euclidean distance of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(8), 1334–1339.

Zhao, W., Chellappa, R., Phillips, P. J., & Rosenfeld, A. (2003). Face recognition: A literature survey. *ACM Computing Surveys*, *35*, 399–458.

Received June 17, 2008; accepted September 24, 2008.

Downloaded from http://direct.mit.edu/neco/article-pdf/21/5/1459/822513/neco.2008.08-06-805.pdf by guest on 16 September 2021