Learning Local Metrics and Influential Regions for Classification

Mingzhi Dong, Yujiang Wang, Xiaochen Yang, Jing-Hao Xue

Abstract—The performance of distance-based classifiers heavily depends on the underlying distance metric, so it is valuable to learn a suitable metric from the data. To address the problem of multimodality, it is desirable to learn local metrics. In this short paper, we define a new intuitive distance with local metrics and influential regions, and subsequently propose a novel local metric learning algorithm called LMLIR for distance-based classification. Our key intuition is to partition the metric space into influential regions and a background region, and then regulate the effectiveness of each local metric to be within the related influential regions. We learn multiple local metrics and influential regions to reduce the empirical hinge loss, and regularize the parameters on the basis of a resultant learning bound. Encouraging experimental results are obtained from various public and popular data sets.

Index Terms—Distance-based classification, distance metric, metric learning, local metric.

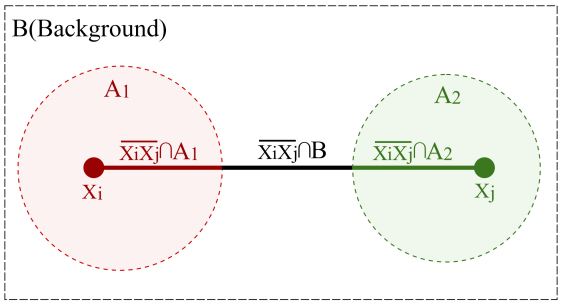


Fig. 1. An example of calculating the distance between two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. A_1 and A_2 are different influential regions with metrics $M(A_1)$ and $M(A_2)$, and B is the background region with metric $M(B)$. The distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ equals to the sum of three line segments' local distances, i.e. $l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j} \cap A_1; M(A_1))$, $l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j} \cap A_2; M(A_2))$ and $l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j} \cap B; M(B))$.

1 Introduction

Classification is a fundamental task in the field of machine learning. While deep learning classifiers have obtained superior performance on numerous applications, they generally require a large amount of labeled data. For small data sets, traditional classification algorithms remain valuable.

The nearest neighbor (NN) classifier is one of the oldest established methods for classification, which compares the distances between a new instance and the training instances. However, with different metrics, the performance of NN would be quite different. Hence it is very beneficial if we can find a well-suited and adaptive distance metric for specific applications. To this end, metric learning is an appealing technique. It enables the algorithms to automatically learn a metric from the available data. Metric learning with a convex objective function was first proposed in the seminal work of Xing et al. [1]. After that, many other metric learning methods have been developed and widely adopted, such as the large margin nearest neighbor (LMNN) [2] and the information theoretic metric learning [3]. Some theoretical work has also been proposed for metric learning, especially on deriving different generalization bounds [4]–[7] and deep networks have been used to represent nonlinear metrics [8], [9]. In addition, metric learning methods have been developed for specific purposes, including multi-output tasks [10], multi-view learning [11], medical image retrieval [12], kinship verification tasks [13], face recognition tasks [14], tracking problems [15] and so on.

Most aforementioned methods use a single metric for the whole metric space and thus may not be well-suited for data sets with multimodality. To solve this problem, local metric learning algorithms have been proposed [2], [16]–[23].

Most of these localized algorithms can be categorized into two groups: 1) Each data point or cluster of data points has a local metric $M(\boldsymbol{x}_i)$. This, however, results in an asymmetric distance as illustrated in [17], i.e. $M(\boldsymbol{x}_i) \neq M(\boldsymbol{x}_j)$ would cause $D(\boldsymbol{x}_i, \boldsymbol{x}_j; M(\boldsymbol{x}_i)) \neq D(\boldsymbol{x}_j, \boldsymbol{x}_i; M(\boldsymbol{x}_j))$. 2) Each line segment or cluster of line segments has a local metric, i.e. $M(\boldsymbol{x}_i, \boldsymbol{x}_j)$. In [19], $M(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_k w_k(\boldsymbol{x}_i, \boldsymbol{x}_j) M_k$, where w_k is defined as $P(k|\boldsymbol{x}_i) + P(k|\boldsymbol{x}_j)$ so as to guarantee the symmetry and $P(k|\boldsymbol{x}_i)$ or $P(k|\boldsymbol{x}_j)$ is based on the posterior probability that the point $\boldsymbol{x}$ belongs to the kth Gaussian cluster in a Gaussian mixture (GMM). However, most of the line segment approaches are based on certain heuristic design. Geometric properties of line segments, which are very intuitive and interpretable, have scarcely been considered.

In this short paper, we define a geometrically interpretable, symmetric distance, and propose a novel local metric learning algorithm that learns local metrics and locations of the local metrics simultaneously; the proposed method is termed as LMLIR. By splitting the metric space into influential regions and a background region, we define the distance between any two points as the sum of lengths of line segments in each region, as illustrated in Fig. 1. Building multiple influential regions solves the multimodality issues; and learning a suitable local metric in

M. Dong, X. Yang and J.-H. Xue are with the Department of Statistical Science, University College London, London WC1E 6BT, UK (e-mail: mingzhi.dong.13@ucl.ac.uk; xiaochen.yang.16@ucl.ac.uk; jinghao.xue@ucl.ac.uk).
Y. Wang is with the Department of Computing, Imperial College London, London SW7 2AZ, UK (e-mail: yujiang.wang14@imperial.ac.uk).

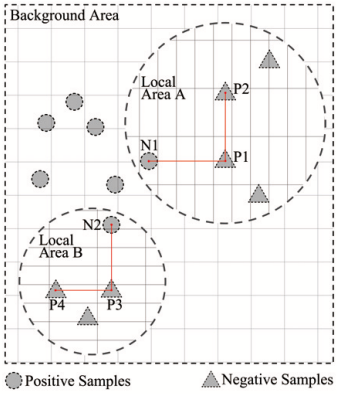


Fig. 2. An illustration of learning local influential regions. The distance between the adjacent vertical/horizontal grids is one unit. The location and radius of a local area could be learned and a suitable local metric could help to enhance the separability of the data, such as increasing $l(\overline{N_1P_1})$ and $l(\overline{N_2P_3})$ while decreasing $l(\overline{P_1P_2})$ and $l(\overline{P_3P_4})$.

each influential region improves class separability, as shown in Fig. 2.

To establish our new distance and local metric learning method, we first define some key concepts, namely influential regions, local metrics and line segments, which lead to the definition of the new distance. Then we calculate the distance by discussing the geometric relationship between line segment and influential regions. After that, we use the proposed local metric to build a novel classifier and study its learnability. The penalty terms from the derived learning bound, together with the empirical hinge loss, form an optimization problem, which is solved via gradient descent due to the non-convexity. Finally we experiment the proposed local metric learning algorithm on 20 publicly available data sets. On ten of these data sets, the proposed algorithm achieves the best performance, much better than the state-of-the-art metric learning competitors.

2 Definitions of Influential Regions, Local Metrics and Distance

In this section, we will first define influential regions $A_s, s = 1, \ldots, S$, and the background region B. With a local metric for each region $M(A_s)$ and $M(B)$, the distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ will be defined as the sum of lengths of line segments in each influential region and the background region, as illustrated in Fig. 1. Since the metric is defined with respect to line segments, the distance is symmetric, i.e. $D(\boldsymbol{x}_i, \boldsymbol{x}_j) = D_{M(\overline{\boldsymbol{x}_i\boldsymbol{x}_j})}(\boldsymbol{x}_i, \boldsymbol{x}_j) = D_{M(\overline{\boldsymbol{x}_j\boldsymbol{x}_i})}(\boldsymbol{x}_j, \boldsymbol{x}_i) = D(\boldsymbol{x}_j, \boldsymbol{x}_i)$.

To simplify the calculation required later, we restrict the shape of each influential region to be a ball.

Definition 1. *Influential regions* are defined to be any set of balls or hyperspheres inside the metric space:

$$A = \{A_s, s = 1, \ldots, S\},$$

where S denotes the number of influential regions; $A_s = \text{Ball}(\boldsymbol{o}_s, r_s)$, a ball with the center $\boldsymbol{o}_s$ and radius r_s. Points $\boldsymbol{x} \in A_s$ construct a set with the following form:

$$\{\boldsymbol{x} | (\boldsymbol{o}_s - \boldsymbol{x})^T (\boldsymbol{o}_s - \boldsymbol{x}) \leq r_s^2\}. \tag{1}$$

The location of each influential region is determined by using the Euclidean distance.

Definition 2. *Background region* is defined to be the region excluding influential regions:

$$B = U - \bigcup_{s=1,\ldots,S} A_s,$$

where U denotes the universe set.

Throughout this paper, the distance between two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is equivalent to the length of line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$, i.e. $D(\boldsymbol{x}_i, \boldsymbol{x}_j) = l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j})$. Length $l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j})$ in influential regions and the background region will be defined separately with respective metrics.

Definition 3. Each influential region A_s has its own *local metric* $M(A_s)$. The length of a line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ inside an influential region A_s is defined as[1]

$$\begin{aligned} l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j}; M(A_s)) =& D_{M(A_s)}(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ =& \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T M(A_s)(\boldsymbol{x}_i - \boldsymbol{x}_j)}. \end{aligned} \tag{2}$$

Here, we adopt the Mahalanobis distance, rather than the widely used squared Mahalanobis distance, since it simplifies the later optimization problem.

Definition 4. The background region B has a *background metric* $M(B)$. For any two points $\boldsymbol{x}_i, \boldsymbol{x}_j \in B$ and $\overline{\boldsymbol{x}_i\boldsymbol{x}_j} \subseteq B$, the length of a line segment is defined as

$$\begin{aligned} l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j}; M(B)) &= D_{M(B)}(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ &= \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T M(B)(\boldsymbol{x}_i - \boldsymbol{x}_j)}. \end{aligned}$$

Note that for $\boldsymbol{x}_i, \boldsymbol{x}_j \in B$ and $\overline{\boldsymbol{x}_i\boldsymbol{x}_j} \nsubseteq B$, the distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is usually different from $D_{M(B)}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. This is because some parts of $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ may lie in influential regions so their lengths should be calculated via the related local metrics.

For any $\boldsymbol{x}_i \in U$ and $\boldsymbol{x}_j \in U$, its line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ may intersect with multiple influential regions and the background region. Therefore, we calculate the distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ as the sum of lengths of line segments in each region. More precisely, as defined below, the distance is the sum of lengths of intersection of $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ and influential regions, plus the length of intersection of $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ and the background region.

Definition 5. The *length of intersection* of a line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ and an influential region A_s is defined as $l(A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}; M(A_s))$, where $\cap$ denotes the intersection operator. The *length of the intersection* of a line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ and the background region B is defined as

$$\begin{aligned} & l(B \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}; M(B)) \\ =& l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j} - \bigcup_{s=1\ldots S} (A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}); M(B)) \\ =& l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j}; M(B)) - l(\bigcup_{s=1\ldots S} (A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}); M(B)), \end{aligned} \tag{3}$$

where $\bigcup_{s=1\ldots S}(A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j})$ denotes the union of intersections between the line segment and all influential regions.

Definition 6. The *length of line segment* $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ is defined as

$$\begin{aligned} & l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j}; M(\overline{\boldsymbol{x}_i\boldsymbol{x}_j})) \\ =& \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T M(\overline{\boldsymbol{x}_i\boldsymbol{x}_j})(\boldsymbol{x}_i - \boldsymbol{x}_j)} \\ =& l(B \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}; M(B)) + \sum_s l(A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}; M(A_s)), \end{aligned} \tag{4}$$

1. Since influential regions are restricted to be ball-shaped and a ball is a convex set, $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ would lie in the ball for any $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ inside the ball.

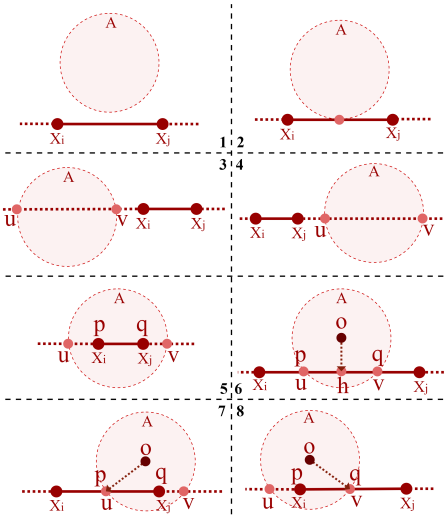


Fig. 3. The positions of $\boldsymbol{u}, \boldsymbol{v}$ (intersection points between line $\boldsymbol{x}_i\boldsymbol{x}_j$ and the influential region A) and $\boldsymbol{p}, \boldsymbol{q}$ (intersection points between line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ and A) under different situations. $\boldsymbol{h}$ is the middle point of line segment $\overline{\boldsymbol{p}\boldsymbol{q}}$.

where $M(\overline{\boldsymbol{x}_i\boldsymbol{x}_j})$ is the metric of the line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$. $M(\overline{\boldsymbol{x}_i\boldsymbol{x}_j})$ will be simplified as M afterwards.

3 Calculation of Distances

3.1 Calculation of the Length of Intersection with Influential Regions

We start by providing an intuitive explanation of calculating the length of intersection with influential regions, as illustrated in Fig. 3. If the line $\boldsymbol{x}_i\boldsymbol{x}_j$ does not intersect with (Fig. 3.1) or is the tangent to the influential ball (Fig. 3.2), the length is zero. If the line intersects with the ball (Fig. 3.3-3.8), we will calculate the length by considering the relationship between the intersection of the line $\boldsymbol{x}_i\boldsymbol{x}_j$ and the influential ball, i.e. $\boldsymbol{u}\boldsymbol{v}$, and the intersection of the line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ and the influential ball, i.e. $\boldsymbol{p}\boldsymbol{q}$.

First, we show that the length of intersection can be calculated given the local metric $M(A_s)$ and the intersection ratio γ defined below.

Definition 7. The *intersection points* of the line $\boldsymbol{x}_i\boldsymbol{x}_j$ and the influential region A_s are represented as $\boldsymbol{u} = \boldsymbol{x}_i + \lambda_u(\boldsymbol{x}_j - \boldsymbol{x}_i)$ and $\boldsymbol{v} = \boldsymbol{x}_i + \lambda_v(\boldsymbol{x}_j - \boldsymbol{x}_i)$, where $\lambda_u, \lambda_v \in \mathbb{R}$, $\lambda_u \leq \lambda_v$, and λ_u, λ_v are called the *intersection coefficients* between the line $\boldsymbol{x}_i\boldsymbol{x}_j$ and A_s. The *intersection points* of the line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ and the influential region are represented as $\boldsymbol{p} = \boldsymbol{x}_i + \lambda_p(\boldsymbol{x}_j - \boldsymbol{x}_i)$ and $\boldsymbol{q} = \boldsymbol{x}_i + \lambda_q(\boldsymbol{x}_j - \boldsymbol{x}_i)$, where $0 \leq \lambda_p \leq \lambda_q \leq 1$ and λ_p, λ_q are called the *intersection coefficients* between the line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ and A_s. $\gamma = \lambda_q - \lambda_p$ is called the *intersection ratio*.

Proposition 1. The length of intersection between line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ and the influential region A_s, with the intersection points $\boldsymbol{p}, \boldsymbol{q}$ and intersection coefficients λ_p, λ_q, is

$$\begin{aligned} l(A \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}; M(A_s)) &= \sqrt{(\boldsymbol{q}-\boldsymbol{p})^T M(A_s)(\boldsymbol{q}-\boldsymbol{p})} \\ &= \gamma\sqrt{(\boldsymbol{x}_i-\boldsymbol{x}_j)^T M(A_s)(\boldsymbol{x}_i-\boldsymbol{x}_j)}. \end{aligned} \tag{5}$$

Next, we figure out the relationship between the line $\boldsymbol{x}_i\boldsymbol{x}_j$ and the influential ball via the one-variable quadratic equation and calculate λ_u, λ_v when they exist. $\boldsymbol{x}_i\boldsymbol{x}_j$ intersects with A_s if we can find $\boldsymbol{u}, \boldsymbol{v}$ that lie on the surface of the ball, which is equivalent to solving the following quadratic equation in one variable λ:

$$\|\boldsymbol{x}_i + \lambda(\boldsymbol{x}_j - \boldsymbol{x}_i) - \boldsymbol{o}_s\|_2^2 = r_s^2. \tag{6}$$

If the discriminant of (6) is positive, then $\boldsymbol{u}, \boldsymbol{v}$ exist and the solutions $\lambda_{u,ij}^s \leq \lambda_{v,ij}^s$ are given by the quadratic equation

$$\begin{aligned} \lambda_{u,ij}^s &= \frac{-2(\boldsymbol{x}_j-\boldsymbol{x}_i)^T(\boldsymbol{x}_i-\boldsymbol{o}_s) - \sqrt{\Delta}}{2(\boldsymbol{x}_j-\boldsymbol{x}_i)^T(\boldsymbol{x}_j-\boldsymbol{x}_i)}, \\ \lambda_{v,ij}^s &= \frac{-2(\boldsymbol{x}_j-\boldsymbol{x}_i)^T(\boldsymbol{x}_i-\boldsymbol{o}_s) + \sqrt{\Delta}}{2(\boldsymbol{x}_j-\boldsymbol{x}_i)^T(\boldsymbol{x}_j-\boldsymbol{x}_i)}, \\ \Delta &= [2(\boldsymbol{x}_j-\boldsymbol{x}_i)^T(\boldsymbol{x}_i-\boldsymbol{o}_s)]^2 \\ &\quad - 4[(\boldsymbol{x}_j-\boldsymbol{x}_i)^T(\boldsymbol{x}_j-\boldsymbol{x}_i)][(\boldsymbol{x}_i-\boldsymbol{o}_s)^T(\boldsymbol{x}_i-\boldsymbol{o}_s) - r_s^2]. \end{aligned}$$

For simplicity, we will drop the superscript s and subscript ij.

Last, we calculate λ_q, λ_p based on λ_u, λ_v. Since $0 \leq \lambda_p \leq \lambda_q \leq 1$, we set $\lambda_p = \lambda_u$ if and only if $\lambda_u \in [0, 1]$ and similarly for λ_q. In other words, we set λ_u, λ_v as follows: $\lambda_p = \min(\max(\lambda_u, 0), 1), \lambda_q = \min(\max(\lambda_v, 0), 1)$. Details are given in Table 1.

TABLE 1
Relationship between λ_u, λ_v and λ_p, λ_q for different positions of $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$

Illustration	λ_u, λ_p	λ_v, λ_q
Fig. 3.3	$\lambda_u < 0 \Rightarrow \lambda_p = 0$	$\lambda_v < 0 \Rightarrow \lambda_q = 0$
Fig. 3.4	$\lambda_u > 1 \Rightarrow \lambda_p = 1$	$\lambda_v > 1 \Rightarrow \lambda_q = 1$
Fig. 3.5	$\lambda_u < 0 \Rightarrow \lambda_p = 0$	$\lambda_v > 1 \Rightarrow \lambda_q = 1$
Fig. 3.6	$0 \leq \lambda_u \leq 1 \Rightarrow \lambda_p = \lambda_u$	$0 \leq \lambda_v \leq 1 \Rightarrow \lambda_q = \lambda_v$
Fig. 3.7	$0 \leq \lambda_u \leq 1 \Rightarrow \lambda_p = \lambda_u$	$\lambda_v > 1 \Rightarrow \lambda_q = 1$
Fig. 3.8	$\lambda_u < 0 \Rightarrow \lambda_p = 0$	$0 \leq \lambda_v \leq 1 \Rightarrow \lambda_q = \lambda_v$

3.2 Calculation of the Length of Intersection Using Local Metrics

Proposition 2. In the case of non-overlapping influential regions, i.e. $A_i \cap A_j = \emptyset, \forall i \neq j$,

$$\begin{aligned} D_M(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \gamma_b\sqrt{(\boldsymbol{x}_i-\boldsymbol{x}_j)^T M(B)(\boldsymbol{x}_i-\boldsymbol{x}_j)} \\ &\quad + \sum_s \gamma_s\sqrt{(\boldsymbol{x}_i-\boldsymbol{x}_j)^T M(A_s)(\boldsymbol{x}_i-\boldsymbol{x}_j)}, \end{aligned} \tag{7}$$

where γ_b is defined as the intersection ratio of the background region and $\gamma_b = 1 - \sum_s \gamma_s$.

Proof:

$$\begin{aligned} D_M(\boldsymbol{x}_i, \boldsymbol{x}_j) &= l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j}; M(B)) - l(\bigcup_{s=1\ldots S}(A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}); M(B)) \\ &\quad + \sum_s l(A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}; M(A_s)) \\ &= (1 - \sum_s \gamma_s)\sqrt{(\boldsymbol{x}_i-\boldsymbol{x}_j)^T M(B)(\boldsymbol{x}_i-\boldsymbol{x}_j)} \\ &\quad + \sum_s \gamma_s\sqrt{(\boldsymbol{x}_i-\boldsymbol{x}_j)^T M(A_s)(\boldsymbol{x}_i-\boldsymbol{x}_j)}. \end{aligned}$$

□

Proposition 2 suggests that the distance can be obtained given metrics $(M(A_s), M(B))$ and the intersection ratio γ_s. All calculations are in closed form and hence the computation is efficient.

To avoid creating overlapping influential regions, we will conduct overlap detection during parameter updates. If the update

of location parameters $(\boldsymbol{o}, r)$ leads to overlap, then we will skip this update and continue on learning other parameters.

4 Classifier and Learnability

In this paper, we select Lipschitz continuous functions as our classifiers since they are a family of smooth functions which are learnable [24]. Based on the resultant learning bounds, we obtain the regularization terms in order to improve the classifier's generalization ability.

4.1 Classifier

To start with, we can see that the following classifier gives the same classification result as 1-NN:

$$f(\boldsymbol{x}) = \min D_{set}(\boldsymbol{x}, \boldsymbol{X}^-) - \min D_{set}(\boldsymbol{x}, \boldsymbol{X}^+),$$

where $D_{set}(\boldsymbol{x}, \boldsymbol{X}^{-/+}) = \{D(\boldsymbol{x}, \boldsymbol{x}_t) | \forall \boldsymbol{x}_t \in \text{negative class / positive class}\}$ and $D(\boldsymbol{x}_i, \boldsymbol{x}_j)$ denotes the distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ defined by any metric. $f(\boldsymbol{x}) < 0$ indicates that $\boldsymbol{x}$ belongs to negative class and $f(\boldsymbol{x}) > 0$ indicates that $\boldsymbol{x}$ belongs to positive class.

In this paper, in order to achieve robustness to noisy instances and incorporate more flexible distance metrics, we extend the above equation by considering more nearby instances as follows:

$$f(\boldsymbol{x}) = \frac{1}{K}\sum_{k=1}^{K} D_{[k]}(x, \boldsymbol{X}^-) - \frac{1}{K}\sum_{k=1}^{K} D_{[k]}(x, \boldsymbol{X}^+), \quad (8)$$

where $D_{[k]}(\boldsymbol{x}, \boldsymbol{X}) = \{D(\boldsymbol{x}, \boldsymbol{x}_t) | \forall \boldsymbol{x}_t \in \boldsymbol{X}\}_{[k]}$ denotes the kth smallest element of the distance set $\{D(\boldsymbol{x}, \boldsymbol{x}_t) | \forall \boldsymbol{x}_t \in \boldsymbol{X}\}$. This function will be used as the classifier in our algorithm.

For multiclass classification, the result will be given by

$$y = \text{argmin}_c \sum_{k=1}^{K} D_{[k]}(\boldsymbol{x}, \boldsymbol{X}^c),$$

where $\boldsymbol{X}^c$ denotes the training instances of class c. It gives the same classification result as (8) in the binary case.

4.2 Learnability of the Classifier with Local Metrics

We will discuss learnability of functions based on the Lipschitz constant, which characterizes the smoothness of a function. The smaller the Lipschitz constant is, the smoother the function is.

Definition 8. [25] Let $(\mathcal{X}, \rho_{\mathcal{X}})$, $(\mathcal{Y}, \rho_{\mathcal{Y}})$ be two metric spaces. The *Lipschitz constant* of a function f is

$$\begin{aligned}\text{Lip}(f) =& \min\{C \in \mathbb{R} | \forall \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X}, \boldsymbol{x}_i \neq \boldsymbol{x}_j, \\ & \rho_{\mathcal{Y}}(f(\boldsymbol{x}_i), f(\boldsymbol{x}_j)) \leq C\rho_{\mathcal{X}}(\boldsymbol{x}_i, \boldsymbol{x}_j)\} \\ =& \max_{\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X}: \boldsymbol{x}_i \neq \boldsymbol{x}_j} \frac{\rho_{\mathcal{Y}}(f(\boldsymbol{x}_i), f(\boldsymbol{x}_j))}{\rho_{\mathcal{X}}(\boldsymbol{x}_i, \boldsymbol{x}_j)}.\end{aligned}$$

Proposition 3. [25] Let $\text{Lip}(f) \leq L_f$ and $\text{Lip}(g) \leq L_g$, then
(a) $\text{Lip}(f + g) \leq L_f + L_g$;
(b) $\text{Lip}(f - g) \leq L_f + L_g$;
(c) $\text{Lip}(af) \leq |a| L_f$, where a is a constant.

Proposition 4. Let $\text{Lip}(f_k(\boldsymbol{x})) \leq L, k = 1, \ldots, N$, then, for any $K \leq N$, $\text{Lip}(\sum_{k=1}^{K} f_{[k]}(\boldsymbol{x}))$ is bounded by $K \max_k L$, where $f_{[k]}(\boldsymbol{x})$ denotes the kth smallest element of the set $\{f_k(\boldsymbol{x}), k = 1, \ldots, K\}$.

Proof. $\forall \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X}, k \in \{1, \ldots, N\}$

$$\begin{aligned}\sum_{k=1}^{K} f_{[k]}(\boldsymbol{x}_i) &= \sum_{k=1}^{K} \{f_k(\boldsymbol{x}_j) + f_k(\boldsymbol{x}_j + (\boldsymbol{x}_i - \boldsymbol{x}_j)) - f_k(\boldsymbol{x}_j)\}_{[k]} \\ &\leq \sum_{k=1}^{K} \{f_k(\boldsymbol{x}_j) + L\|\boldsymbol{x}_j + (\boldsymbol{x}_i - \boldsymbol{x}_j) - \boldsymbol{x}_j\|\}_{[k]} \\ &\leq \sum_{k=1}^{K} \{f_k(\boldsymbol{x}_j) + L\|\boldsymbol{x}_i - \boldsymbol{x}_j\|\}_{[k]} \\ &= \sum_{k=1}^{K} f_{[k]}(\boldsymbol{x}_j) + KL\|\boldsymbol{x}_i - \boldsymbol{x}_j\|.\end{aligned}$$

Based on the definition of Lipschitz constant, the proposition is proved. □

Lemma 1. With distance defined in (7), the Lipschitz constant of the classifier specified in (8) is bounded by $L = 2(\sum_s \sqrt{\|M(A_s)\|_F} + \sqrt{\|M(B)\|_F})$, where $\|\cdot\|_F$ denotes the matrix Frobenius norm.

Proof. Let $d_M(\boldsymbol{x}, \boldsymbol{x}_k)$ denote the Mahalanobis distance with metric M, i.e.

$$d_M(\boldsymbol{x}, \boldsymbol{x}_k) = \sqrt{(\boldsymbol{x} - \boldsymbol{x}_k)^T M (\boldsymbol{x} - \boldsymbol{x}_k)},$$

and $d_I(\boldsymbol{x}, \boldsymbol{x}_k)$ denotes the Euclidean distance with the identity matrix I.

The Mahalanobis distance $d_M(\boldsymbol{x}, \boldsymbol{x}_k)$ has the Lipschitz constant of $\sqrt{\|M\|_F}$ as follows:

$$\begin{aligned}\text{Lip}(d_M(\boldsymbol{x}, \boldsymbol{x}_k)) &= \sup_{\boldsymbol{x}_a, \boldsymbol{x}_b \in \mathcal{X}, \boldsymbol{x}_a \neq \boldsymbol{x}_b} \frac{d_M(\boldsymbol{x}_a, \boldsymbol{x}_k) - d_M(\boldsymbol{x}_b, \boldsymbol{x}_k)}{d_I(\boldsymbol{x}_a, \boldsymbol{x}_b)} \\ &\leq \sup_{\boldsymbol{x}_a, \boldsymbol{x}_b \in \mathcal{X}, \boldsymbol{x}_a \neq \boldsymbol{x}_b} \frac{d_M(\boldsymbol{x}_a, \boldsymbol{x}_b)}{d_I(\boldsymbol{x}_a, \boldsymbol{x}_b)} \\ &\leq \sup_{\boldsymbol{x}_a, \boldsymbol{x}_b \in \mathcal{X}, \boldsymbol{x}_a \neq \boldsymbol{x}_b} \frac{d_I(\boldsymbol{x}_a, \boldsymbol{x}_b)\sqrt{\|M\|_F}}{d_I(\boldsymbol{x}_a, \boldsymbol{x}_b)} \\ &= \sqrt{\|M\|_F},\end{aligned}$$

where the first inequality follows the triangle inequality of distance, and the second inequality is based on the Cauchy-Schwarz inequality and the fact that Frobenius norm is compatible with the vector l_2 norm.

According to the definition of distance in (7), we have

$$\begin{aligned}D_M(\boldsymbol{x}, \boldsymbol{x}_k) &= \sum_s \gamma_s d_{M(A_s)}(\boldsymbol{x}, \boldsymbol{x}_k) + \gamma_b d_{M(B)}(\boldsymbol{x}, \boldsymbol{x}_k) \\ &\leq \sum_s d_{M(A_s)}(\boldsymbol{x}, \boldsymbol{x}_k) + d_{M(B)}(\boldsymbol{x}, \boldsymbol{x}_k)\end{aligned}$$

as $\gamma_s, \gamma_b \leq 1$. From Proposition 3, we get that

$$\text{Lip}(D_M(\boldsymbol{x}, \boldsymbol{x}_k)) \leq \sum_s \sqrt{\|M(A_s)\|_F} + \sqrt{\|M(B)\|_F}.$$

Based on the Lipschitz constant of $D_M(\boldsymbol{x}, \boldsymbol{x}_k)$ and the composition property illustrated in Proposition 4,

$$\begin{aligned}&\text{Lip}(\sum_{k=1}^{K} \{D_M(\boldsymbol{x}, \boldsymbol{x}_k), k = 1, \ldots K\}_{[k]}) \\ \leq& K\left\{\sum_s \sqrt{\|M(A_s)\|_F} + \sqrt{\|M(B)\|_F}\right\}.\end{aligned}$$

Finally, based on Proposition 3, $f(\boldsymbol{x})$ in (8) is bounded by $2(\sum_s \sqrt{\|M(A_s)\|_F} + \sqrt{\|M(B)\|_F})$. □

Combining Lemma 1 and Corollary 2 of [24], we can obtain the following Corollary.

Corollary 1. Let metric space $\mathcal{X}$ have doubling dimension $\mathrm{ddim}(\mathcal{X})$ and let $\mathcal{F}$ be the collection of real valued functions over $\mathcal{X}$ with the Lipschitz constant at most L. Then for any $f \in \mathcal{F}$ that classify correctly on all but k examples, we have with probability at least $1-\delta$

$$\begin{aligned} &P\{(\boldsymbol{x},t) : \mathrm{sign}[f(\boldsymbol{x})] \neq t\} \\ \leq &\frac{k}{n} + \sqrt{\frac{2}{n}(c\ln(34en/c)\log_2(578n) + \ln(4/\delta))}, \end{aligned} \tag{9}$$

where n denotes the sample size, $t \in \{-1, 1\}$ denotes the label, and

$$\begin{aligned} c &\leq (16L\,\mathrm{diam}(\mathcal{X}))^{\mathrm{ddim}(\mathcal{X})} \\ &= \Big(32(\sum_s \sqrt{\|M(A_s)\|_F} + \sqrt{\|M(B)\|_F})\,\mathrm{diam}(\mathcal{X})\Big)^{\mathrm{ddim}(\mathcal{X})}. \end{aligned}$$

diam denotes the diameter of the space and ddim denotes doubling dimension; precise definitions can be found in [24].

The above learning bound illustrates that the generalization ability, i.e. the difference between the expected error $P\{(\boldsymbol{x},t) : \mathrm{sign}[f(\boldsymbol{x})] \neq t\}$ and the empirical error k/n, can be improved by reducing the value of $\sum_s \sqrt{\|M(A_s)\|_F} + \sqrt{\|M(B)\|_F}$. Since the square function is monotonically increasing, we would instead reduce $\sum_s \|M(A_s)\|_F + \|M(B)\|_F$. In other words, $\sum_s \|M(A_s)\|_F + \|M(B)\|_F$ would be used as the regularization term to improve the generalization ability of the classifier.

5 Optimization Problem

5.1 Objective Function

In order to obtain low training error and good generalization ability, we propose the following optimization problem, where the objective function consists of a sum of hinge loss and the regularization term $\sum_s \|\boldsymbol{M}(A_s)\|_F + \|\boldsymbol{M}(B)\|_F$:

$$\begin{aligned} &\min_{\Theta,\boldsymbol{\xi}} \frac{1}{N_1}\sum_{(i,j)}\xi_{ij} + \frac{1}{N_2}\sum_{(m,n)}\xi_{mn} + \alpha\sum_s \|M(A_s)\|_F + \alpha\|M(B)\|_F \\ &\text{s.t. } D_M(\boldsymbol{x}_i,\boldsymbol{x}_j) \leq 1 - C + \xi_{ij}, D_M(\boldsymbol{x}_m,\boldsymbol{x}_n) \geq 1 + C + \xi_{mn} \\ &\quad \xi_{ij}, \xi_{mn} \geq 0, M \in \boldsymbol{M}_+ \\ &\quad i,m = 1,\ldots,N, j \rightarrow i, n \nrightarrow m, \end{aligned} \tag{10}$$

where $\Theta = \{M(A_s), M(B), \boldsymbol{o}, \boldsymbol{r}\}$ denotes the set of parameters to be optimized; $j \rightarrow i$ denotes that $\boldsymbol{x}_j$ is $\boldsymbol{x}_i$'s K nearest neighbor comparing against all instances in the same class; $m \nrightarrow n$ denotes that $\boldsymbol{x}_n$ is $\boldsymbol{x}_m$'s K nearest neighbor comparing against all instances in the different class; C is a constant which has the intuition of margin; ξ_{ij} and ξ_{mn} denote the error caused by margin violation; N, N_1, N_2 denote the number of training samples, pairs (i,j), and pairs (m,n) respectively ; α is a trade-off parameter between the margin loss and the regularization terms. This optimization formula is suitable for both binary and multi-class tasks. In the proposed algorithm, we will learn the locations of influential regions $(\boldsymbol{o}_s, r_s)$ and the metrics of influential/background regions $(M(B), M(A_s))$ under the same framework.

5.2 Gradient Descent

With $D_{M(A_s)}$ and $D_{M(B)}$ being the Mahalanobis distances, the optimization problem is convex even when $\boldsymbol{o}, \boldsymbol{r}$ are fixed and only $M(A_s)$ and $M(B)$ are updated. Therefore, we adopt the gradient descent algorithm:

$$\Theta^{t+1} = \Theta^t - \beta\frac{\partial g}{\partial \Theta}|_{\Theta^t},$$

where β is the learning rate, and the superscript t denotes the time step during optimization.

The objective function g is

$$\begin{aligned} g = &\frac{1}{N_1}[D_M(\boldsymbol{x}_i,\boldsymbol{x}_j) - (1-C)]_+ + \alpha\sum_s \|M(A_s)\|_F \\ &+ \frac{1}{N_2}[1 + C - D_M(\boldsymbol{x}_m,\boldsymbol{x}_n)]_+ + \alpha\|M(B)\|_F, \end{aligned}$$

where the distance is

$$\begin{aligned} D_M(\boldsymbol{x}_i,\boldsymbol{x}_j) &= [\gamma_b(\boldsymbol{o}_s, r_s)]_+ D_{M(B)}(\boldsymbol{x}_i,\boldsymbol{x}_j) \\ &+ \sum_s \gamma_s(\boldsymbol{o}_s, r_s) D_{M(A_s)}(\boldsymbol{x}_i,\boldsymbol{x}_j) \end{aligned}$$

and $\gamma_b(\boldsymbol{o}_s, r_s) = 1 - \sum_s \gamma_s(\boldsymbol{o}_s, r_s)$. Here, γ_s is written as $\gamma_s(\boldsymbol{o}_s, r_s)$ to remind us that γ_s is a function of the location parameters $\boldsymbol{o}_s$ and r_s.

The gradient of g with respect to parameters $\boldsymbol{o}, r$ is

$$\begin{aligned} &\frac{\partial g}{\partial \Theta}|_{\Theta^t} = \\ &\frac{1}{N_1}\sum_{(i,j)} \mathbf{1}[D_{M^t}(\boldsymbol{x}_i,\boldsymbol{x}_j) - (1-C) > 0]\frac{\partial D_M(\boldsymbol{x}_i,\boldsymbol{x}_j)}{\partial \Theta}|_{\Theta^t} \\ -&\frac{1}{N_2}\sum_{(m,n)} \mathbf{1}[1 + C - D_{M^t}(\boldsymbol{x}_m,\boldsymbol{x}_n) > 0]\frac{\partial D_M(\boldsymbol{x}_m,\boldsymbol{x}_n)}{\partial \Theta}|_{\Theta^t}. \end{aligned}$$

If the gradient is with respect to $M(B)$ and $M(A^s)$, then the shrinkage term of $\frac{\alpha M(B)}{\|M(B)\|}$ or $\frac{\alpha M(A_s)}{\|M(A_s)\|}$ should be added into the above formula.

Now we will calculate $\frac{\partial D_M(\boldsymbol{x}_i,\boldsymbol{x}_j)}{\partial \Theta}|_{\Theta^t}$ for the parameters $M(A_s)$, $M(B)$, $\boldsymbol{o}_s$, $\boldsymbol{r}_s$ separately:

$$\begin{aligned} &\frac{\partial D(\boldsymbol{x}_i,\boldsymbol{x}_j)}{\partial M(A_s)}|_{\Theta^t} = \frac{\gamma_s(\boldsymbol{o}_s^t, r_s^t)}{2}\times \\ &[(\boldsymbol{x}_i - \boldsymbol{x}_j)^T M^t(A_s)(\boldsymbol{x}_i - \boldsymbol{x}_j)]^{-1/2}(\boldsymbol{x}_i - \boldsymbol{x}_j)(\boldsymbol{x}_i - \boldsymbol{x}_j)^T; \\ &\frac{\partial D(\boldsymbol{x}_i,\boldsymbol{x}_j)}{\partial M(B)}|_{\Theta^t} = \frac{\mathbf{1}[\gamma_b(\boldsymbol{o}_s^t, r_s^t) > 0]\gamma_b(\boldsymbol{o}_s^t, r_s^t)}{2}\times \\ &[(\boldsymbol{x}_i - \boldsymbol{x}_j)^T M^t(B)(\boldsymbol{x}_i - \boldsymbol{x}_j)]^{-1/2}(\boldsymbol{x}_i - \boldsymbol{x}_j)(\boldsymbol{x}_i - \boldsymbol{x}_j)^T; \\ &\frac{\partial D(\boldsymbol{x}_i,\boldsymbol{x}_j)}{\partial \boldsymbol{o}_s}|_{\Theta^t} = \frac{\partial \gamma_s}{\partial \boldsymbol{o}_s} D_{M^t(A_s)}(\boldsymbol{x}_i,\boldsymbol{x}_j) - \\ &\qquad \frac{\partial \gamma_s}{\partial \boldsymbol{o}_s}\mathbf{1}[\gamma_b(\boldsymbol{o}_s^t, r_s^t) > 0] D_{M^t(B)}(\boldsymbol{x}_i,\boldsymbol{x}_j) \end{aligned}$$

where $\frac{\partial \gamma}{\partial \boldsymbol{o}}$ could be obtained as illustrated in Table 2;

$$\begin{aligned} \frac{\partial D(\boldsymbol{x}_i,\boldsymbol{x}_j)}{\partial r_s}|_{\Theta^t} = &\frac{\partial \gamma_s}{\partial r_s} D_{M^t(A_s)}(\boldsymbol{x}_i,\boldsymbol{x}_j) - \\ &\frac{\partial \gamma_s}{\partial r_s}\mathbf{1}[\gamma_b(\boldsymbol{o}_s^t, r_s^t) > 0] D_{M^t(B)}(\boldsymbol{x}_i,\boldsymbol{x}_j), \end{aligned}$$

where $\frac{\partial \gamma}{\partial r}$ could be obtained as illustrated in Table 2.

Initial values are crucial for non-convex optimization problems. We adopt a heuristic method to initialize the parameters as

TABLE 2
Partial gradients of $\frac{\partial\gamma}{\partial \boldsymbol{o}}$ and $\frac{\partial\gamma}{\partial r}$ in different cases. $\overrightarrow{oh}$, $\overrightarrow{ov}$, $\overrightarrow{ou}$ can be found from Fig. 3.

λ_u, λ_v	γ	partial gradients
$\lambda_u < 0, \lambda_v < 0$	0	$\frac{\partial\gamma}{\partial \boldsymbol{o}} = \mathbf{0}$, $\frac{\partial\gamma}{\partial r} = 0$
$\lambda_u < 0, \lambda_v > 1$	1	
$\lambda_u > 1, \lambda_v > 1$	0	
$0 \leq \lambda_u \leq 1, 0 \leq \lambda_v \leq 1$	$\lambda_v - \lambda_u$	$\frac{\partial\gamma}{\partial \boldsymbol{o}} = -\frac{2\Delta^{-1/2}(\boldsymbol{x}_j-\boldsymbol{x}_i)^T(\boldsymbol{x}_i-\boldsymbol{o})(\boldsymbol{x}_j-\boldsymbol{x}_i)}{(\boldsymbol{x}_j-\boldsymbol{x}_i)^T(\boldsymbol{x}_j-\boldsymbol{x}_i)} - 4\Delta^{-1/2}(\boldsymbol{o}-\boldsymbol{x}_i) = 4\Delta^{-1/2}\overrightarrow{oh}$, $\frac{\partial\gamma}{\partial r} = 4\Delta^{-1/2}r$
$\lambda_u < 0, 0 \leq \lambda_v \leq 1$	λ_v	$\frac{\partial\gamma}{\partial \boldsymbol{o}} = \frac{(\boldsymbol{x}_j-\boldsymbol{x}_i)-\Delta^{-1/2}(\boldsymbol{x}_j-\boldsymbol{x}_i)^T(\boldsymbol{x}_i-\boldsymbol{o})(\boldsymbol{x}_j-\boldsymbol{x}_i)}{(\boldsymbol{x}_j-\boldsymbol{x}_i)^T(\boldsymbol{x}_j-\boldsymbol{x}_i)} - 2\Delta^{-1/2}(\boldsymbol{o}-\boldsymbol{x}_i) = 2\Delta^{-1/2}\overrightarrow{ov}$, $\frac{\partial\gamma}{\partial r} = 2\Delta^{-1/2}r$
$0 \leq \lambda_u \leq 1, \lambda_v > 1$	$1 - \lambda_u$	$\frac{\partial\gamma}{\partial \boldsymbol{o}} = \frac{-(\boldsymbol{x}_j-\boldsymbol{x}_i)-\Delta^{-1/2}(\boldsymbol{x}_j-\boldsymbol{x}_i)^T(\boldsymbol{x}_i-\boldsymbol{o})(\boldsymbol{x}_j-\boldsymbol{x}_i)}{(\boldsymbol{x}_j-\boldsymbol{x}_i)^T(\boldsymbol{x}_j-\boldsymbol{x}_i)} - 2\Delta^{-1/2}(\boldsymbol{o}-\boldsymbol{x}_i) = 4\Delta^{-1/2}\overrightarrow{ou}$, $\frac{\partial\gamma}{\partial r} = 2\Delta^{-1/2}r$

follows. 1) Extract local discriminative direction $h(\boldsymbol{x}) \in R^F$ for each training instance $\boldsymbol{x}$, where F indicates the number of features of $\boldsymbol{x}$:

$$h(\boldsymbol{x}_i)[f] = \sum_{k \nrightarrow i} |\boldsymbol{x}_k[f] - \boldsymbol{x}_i[f]| - \sum_{j \rightarrow i} |\boldsymbol{x}_j[f] - \boldsymbol{x}_i[f]|,$$

where $\boldsymbol{x}[f]$ indicates the fth dimension of vector $\boldsymbol{x}$. 2) Perform nonparametric clustering: The Dirichlet process Gaussian mixture model is applied to the augmented feature vector $[\boldsymbol{x}, h(\boldsymbol{x})]$ to group instances into clusters; the number of clusters, and hence the number of influential regions, is automatically decided by the clustering algorithm. 3) Initialize the parameters: Cluster centers are initialized as $\boldsymbol{o}_s$; the 80th percentile of the distance between samples and the cluster center is set as initial value of r_s; the local metric is set as $M(A_s) = I + 0.1 \times \text{diag}(\text{mean}(h(\boldsymbol{x}), \boldsymbol{x} \in \text{cluster } s))$, where diag is an operation which returns a square diagonal matrix with elements of the input vector on the main diagonal. The initialization process is carried out from the largest cluster to the smallest one. If a later influential region overlaps with an earlier one, the later region will be shrunk, or even deleted, until no overlap exists.

6 Experiments

6.1 Toy Example

To visualize the learned parameters, we consider a toy data set for binary classification consisting of 80 instances generated from a two-component Gaussian mixture model. 40 instances in the positive and 40 instances in the negative class are sampled from $\frac{1}{2}N[(-1,0), \frac{1}{2}\boldsymbol{I}] + \frac{1}{2}N[(1,0), \frac{1}{2}\boldsymbol{I}]$ and $\frac{1}{2}N[(-1,2), \frac{1}{2}\boldsymbol{I}] + \frac{1}{2}N[(3,0), \frac{1}{2}\boldsymbol{I}]$ respectively. Parameters in our algorithm are set as follows: α and C in the optimization formula are 0.1 and 0.5 respectively; the number of clusters used for initializing the parameters is 2; the gradient descent algorithm stops after 50 iterations. For illustration purpose, overlap detection has not been conducted on the toy example.

In Figs. 4a-4c, we learn one parameter from $\{M(A), \boldsymbol{o}, r\}$ at each time, fixing the other parameters. Take Fig. 4a (left) as an example. Since $M(A_1) = M(A_2) = 2\boldsymbol{I}$ and $M(B) = \boldsymbol{I}$, the influential regions act as enlarging the local distance. In this case, we see that the centers of $A1$ and $A2$ move to the inter-class region. This phenomenon could be explained as follows. For a line segment that lies in an inter-class region and violates the margin constraint, i.e. $D_M(\boldsymbol{x}_m, \boldsymbol{x}_n) < 1 + C$, the direction of gradient descent is same as that of $\frac{\partial D_M(\boldsymbol{x}_m, \boldsymbol{x}_n)}{\partial \boldsymbol{o}}$. As $D_{M(A)}(\boldsymbol{x}_m, \boldsymbol{x}_n) > D_{M(B)}(\boldsymbol{x}_m, \boldsymbol{x}_n)$, $\frac{\partial D_M(\boldsymbol{x}_m, \boldsymbol{x}_n)}{\partial \boldsymbol{o}}$ has the same direction as $\frac{\partial r}{\partial \boldsymbol{o}}$, which, according to Table 2, is the direction of $\overrightarrow{oh}$, $\overrightarrow{ou}$, $\overrightarrow{ov}$ in Fig. 3 depending on the value of γ. In other words, the margin-violated inter-class line segments will pull the influential regions towards the inter-class region. At the same time, for an intra-class line segment that violates the margin constraint, i.e. $D_M(\boldsymbol{x}_i, \boldsymbol{x}_j) > 1 - C$, the direction of gradient descent is opposite to that of $\frac{\partial D_M(\boldsymbol{x}_m, \boldsymbol{x}_n)}{\partial \boldsymbol{o}}$, and hence opposite to $\overrightarrow{oh}$, $\overrightarrow{ou}$, $\overrightarrow{ov}$. That is, the margin-violated intra-class line segments will push the influential regions away from the intra-class region. In summary, as illustrated in Fig. 4a (left), when the influential regions have the effect of 'enlarging' distance, $\boldsymbol{o}$ move to the inter-class region. Similar reasoning applies to Fig. 4a (right), 4b, and 4c. In Fig. 4d, $M(A), \boldsymbol{o}, r$ are learned simultaneously. As expected, the influential regions focus on inter-class samples by moving towards the inter-class region, increasing the region size, and enlarging the local distance in the direction that is nearly perpendicular to the decision boundary.

The toy example demonstrates that the gradient learning has a clear geometric interpretation.

6.2 Real Data

We compare our algorithm with twelve established metric learning algorithms from three categories: (1) the most cited algorithms, including large margin nearest neighbor (LMNN) [2] and information theoretic metric learning (ITML) [3]; (2) local metric learning algorithms, including multiple-metric large margin nearest neighbor (mmLMNN) [2], parametric local metric learning (PLML) [17], reduced-rank local distance metric learning (R2LML) [18], and local discriminative distance metrics ensemble learning (LDDM) [26]; (3) the state-of-the-art metric learning algorithms, including distance metric learning with eigenvalue optimization (DMLE) [27], sparse compositional metric learning (SCML) [20], stochastic neighbor compression (SNC) [28], regressive virtual metric learning (RVML) [29], geometric mean metric learning (GMML) [30], and supervised distance metric learning through maximization of the Jeffrey divergence (DMLMJ) [31]. LMNN and ITML are implemented using the metric-learn toolbox[2]; mmLMNN, PLML, R2LML, LDDM, DMLE, SCML, SNC, RVML, GMML and DMLMJ are implemented using the authors' code.

We conduct binary classification on 14 data sets and multiple-class classification on 6 data sets, all of which are publicly available from UCI[3] and LibSVM[4]. For binary classification, we use data sets Australian, Breastcancer, Diabetes, Fourclass,

2. https://all-umass.github.io/metric-learn/
3. https://archive.ics.uci.edu/ml/datasets.html
4. https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/binary.html

TABLE 3
Metric learning algorithm results: Mean accuracy and standard deviation are reported with the best ones in bold; 'AVERAGE' denotes the average accuracy of all data sets; '# of BEST' denotes the number of data sets that an algorithm performs the best; 'NAN' indicates the algorithm cannot return a classification result for the data set.

Data sets	LMNN	ITML	mmLMNN	PLML	R2LML	LDDM	DMLE	SCML	SNC	RVML	GMML	DMLMJ	LMLIR
Binary classification													
Australian	78.8±2.6	77.2±1.9	82.5±2.6	80.5±1.1	84.7±1.3	72.8±9.1	82.6±1.5	82.3±1.4	81.8±8.8	83.0±1.6	84.4±1.0	83.9±1.3	**85.1±1.9**
Breastcancer	95.9±0.7	96.4±1.0	96.7±1.0	96.4±0.9	97.0±0.7	66.1±1.8	97.0±1.1	97.0±0.9	96.7±0.7	95.8±1.1	**97.3±0.8**	96.6±0.8	96.4±2.1
Diabetes	69.2±1.4	69.1±1.2	72.2±1.9	68.5±2.0	73.8±1.4	64.4±2.0	72.6±2.0	71.5±2.2	75.3±2.7	71.0±2.6	74.2±2.6	71.5±3.1	**75.9±1.9**
Fourclass	72.1±2.3	72.1±2.2	75.6±1.4	72.4±2.4	76.1±1.9	64.0±2.1	75.6±1.4	75.5±1.4	73.4±8.7	70.5±1.4	76.1±1.9	76.1±1.9	**79.9±0.9**
German	67.9±1.5	67.0±2.1	68.9±1.8	70.0±2.9	72.9±1.8	70.1±1.5	72.0±2.1	70.9±2.7	70.1±3.3	71.7±1.8	71.6±1.1	69.3±2.7	**73.7±1.6**
Haberman	67.9±3.3	68.0±4.1	69.0±2.7	67.1±3.1	71.1±3.4	73.8±3.6	70.8±3.5	69.2±2.5	72.0±5.2	66.7±2.3	71.2±3.4	68.5±3.2	**74.4±3.7**
Heart	76.2±3.8	76.9±3.3	79.4±3.7	75.1±3.2	82.0±3.8	71.6±9.7	77.9±3.1	79.0±3.2	77.0±5.3	77.7±4.1	81.2±2.7	80.6±2.8	**83.1±3.2**
ILPD	67.0±2.1	68.7±2.8	66.8±2.1	67.4±3.0	65.9±2.2	**72.4±1.1**	68.8±2.7	68.0±2.9	68.9±2.7	68.0±2.9	67.1±2.2	68.0±1.6	69.6±2.7
Liverdisorders	61.0±4.8	57.2±4.0	62.0±3.5	62.2±2.5	**66.8±3.7**	56.8±3.8	61.8±2.7	61.7±4.6	63.3±5.2	64.6±3.9	63.8±5.4	60.9±3.8	66.7±3.6
Monk1	88.4±2.6	77.3±1.3	90.3±2.6	96.6±2.7	89.2±1.5	67.9±8.1	**99.9±0.3**	97.5±0.9	96.8±4.8	89.2±2.7	75.0±2.6	87.7±3.8	95.0±7.2
Pima	68.5±1.6	68.0±2.0	72.5±2.7	68.4±2.2	72.3±1.5	64.9±2.6	72.1±2.4	71.1±2.6	74.0±2.6	69.5±1.7	73.0±1.8	71.1±2.8	**74.6±2.0**
Planning	60.4±5.3	62.2±2.3	54.7±3.4	60.8±5.5	63.9±3.4	**72.1±7.8**	60.1±5.5	61.9±5.0	NAN	55.1±7.4	65.2±5.5	64.3±2.9	67.5±6.5
Voting	94.8±0.8	90.8±1.4	95.4±0.9	95.5±1.0	**96.3±1.2**	65.1±10.3	93.1±1.9	95.0±1.3	94.5±1.2	95.8±1.3	95.2±1.9	95.3±1.1	93.2±3.9
WDBC	96.6±1.1	94.9±0.9	**97.4±1.0**	96.4±0.9	96.9±1.7	63.2±3.5	96.7±0.5	97.0±0.9	96.9±0.9	96.6±1.3	96.7±0.8	97.3±1.9	96.6±1.0
AVERAGE	76.0	74.6	77.3	76.9	79.2	67.5	78.6	78.4	NAN	76.7	77.9	77.9	80.8
# of BEST	0	0	1	0	2	2	1	0	0	0	1	0	7
Multiclass classification													
Cleveland	54.6±2.1	56.2±2.2	53.9±2.2	49.0±4.1	57.7±2.1	52.9±2.3	54.0±2.4	54.4±3.5	53.3±3.1	50.9±4.4	**59.1±2.3**	55.3±3.2	57.7±3.5
Glass	70.7±4.8	69.9±4.7	NAN	NAN	70.2±5.5	41.6±9.0	66.2±5.3	71.7±2.9	69.5±6.5	68.1±3.7	69.9±6.0	59.3±5.1	**72.0±5.7**
Iris	86.7±2.9	87.0±3.3	86.5±3.6	82.7±6.9	87.0±4.6	70.0±13.3	86.8±3.6	87.3±3.1	NAN	83.8±4.2	87.5±3.7	85.3±4.8	**87.8±3.8**
Newthyroid	88.6±2.7	90.0±2.3	88.5±3.2	89.0±2.1	90.4±3.2	69.9±3.0	89.2±2.1	89.3±3.3	89.7±2.5	88.3±1.8	89.8±3.4	**91.1±2.1**	90.6±1.9
Tae	50.2±8.2	46.2±7.0	50.2±7.2	50.8±8.3	50.8±6.1	29.2±5.1	49.7±4.4	53.6±5.9	NAN	**55.4±6.9**	51.2±6.3	49.0±6.9	53.6±6.7
Winequality(red)	58.3±1.9	56.1±1.5	NAN	NAN	58.0±1.2	NAN	55.0±1.7	58.9±1.7	58.2±4.0	59.6±2.3	58.2±1.8	49.0±3.9	**60.08±6.5**
AVERAGE	73.0	71.8	NAN	NAN	75.2	62.1	74.1	75.1	NAN	73.8	74.3	72.7	76.9
# of BEST	0	0	0	0	0	0	0	0	0	1	1	1	3

Germannumber, Haberman, Heart, ILPD, Liverdisorders, Monk1, Pima, Planning, Voting, and WDBC; for multiple-class, we use Cleveland, Glass, IRIS, Newthyroid, Tae, and Winequality (red). All data sets are pre-processed by firstly subtracting the mean and dividing by the standard deviation, and then normalizing the L2-norm of each instance to one.

For each data set, 60% instances are randomly selected as training samples and the rest for testing. This process is repeated 10 times and the mean accuracy and the standard deviation are reported. We use 10-fold cross-validation to select the trade-off parameters in the compared algorithms, namely the regularization parameter of LMNN (from $\{0.1, 0.3, 0.5, 0.7, 0.9\}$), γ in ITML (from $\{0.25, 0.5, 1, 2, 4\}$), t in GMML (from $\{0.1, 0.3, 0.5, 0.7, 0.9\}$) and λ in RVML (from $\{10^{-5}, 10^{-4}, \ldots, 10\}$). All other parameters are set as default. For our algorithm, we set the parameters as follows: α and C in the optimization formula are 0.1 and 0.5 respectively; K in the classifier is 10. The number of influential regions in our algorithm is determined via Dirichlet process Gaussian mixture model and it is implemented with PRML toolbox [5].

Results for binary classification and multiclass classification are shown in Table 3. The proposed algorithm achieves the highest average accuracy on both tasks. Out of 20 data sets, LMLIR outperforms all other methods on ten data sets out and none of the other algorithms performs the best in more than two data sets. In cases where our algorithm is not leading, the difference to the optimal method is relatively small. Such encouraging results demonstrate the effectiveness of our proposed method.

7 Conclusions and Future Work

In this short paper, by introducing influential regions, we define a very intuitive distance and propose a novel local metric learning method. The distance can be computed efficiently and encouraging results are obtained on public data sets.

5. https://github.com/PRML/PRMLT

Some directions merit future investigation to extend our work. Original features from data are used in this paper, but we may explore deep features for specified tasks. More advanced optimization techniques and other types of influential regions may also be explored. Domain knowledge can be embedded into the partition of the regions. Tighter learning bounds and resultant penalty terms would also be our future work.

References

[1] E. P. Xing, M. I. Jordan, S. Russell, and A. Y. Ng, "Distance metric learning with application to clustering with side-information," in *Advances in neural information processing systems*, 2002, pp. 505–512.

[2] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *The Journal of Machine Learning Research*, vol. 10, pp. 207–244, 2009.

[3] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, "Information-theoretic metric learning," in *Proceedings of the 24th international conference on Machine learning*. ACM, 2007, pp. 209–216.

[4] R. Jin, S. Wang, and Y. Zhou, "Regularized distance metric learning: Theory and algorithm," in *Advances in neural information processing systems*, 2009, pp. 862–870.

[5] Z.-C. Guo and Y. Ying, "Guaranteed classification via regularized similarity learning," *Neural computation*, vol. 26, no. 3, pp. 497–522, 2014.

[6] Q. Cao, Z.-C. Guo, and Y. Ying, "Generalization bounds for metric and similarity learning," *Machine Learning*, vol. 102, no. 1, pp. 115–132, 2016.

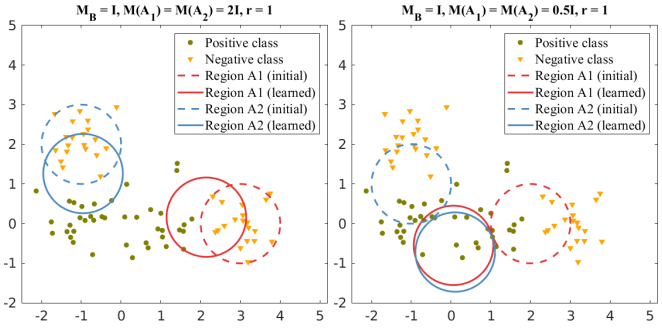
[7] N. Verma and K. Branson, "Sample complexity of learning mahalanobis distance metrics," in *Advances in Neural Information Processing Systems*, 2015, pp. 2584–2592.

[8] J. Hu, J. Lu, and Y.-P. Tan, "Discriminative deep metric learning for face verification in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1875–1882.

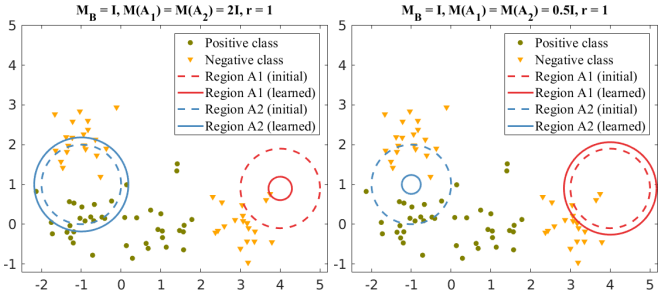
[9] J. Lu, G. Wang, W. Deng, P. Moulin, and J. Zhou, "Multi-manifold deep metric learning for image set classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1137–1145.

[10] W. Liu, D. Xu, I. Tsang, and W. Zhang, "Metric learning for multi-output tasks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

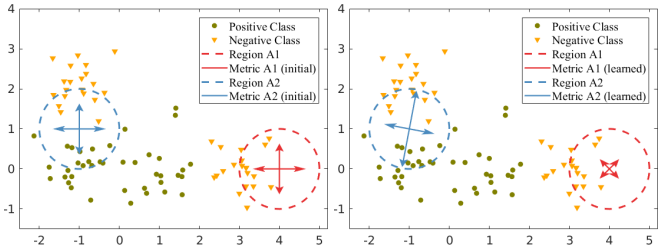
[11] J. Hu, J. Lu, and Y.-P. Tan, "Sharable and individual multi-view metric learning," *IEEE transactions on pattern analysis and machine intelligence*, 2017.



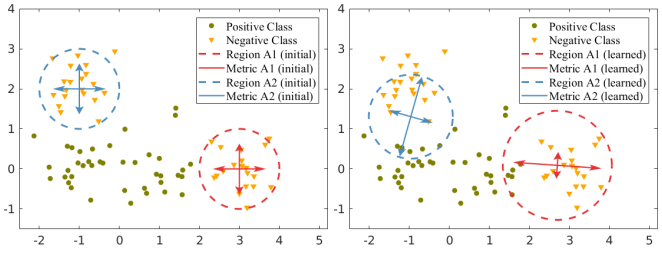
(a) Learn the centers $\boldsymbol{o}$ with $\boldsymbol{M}, r$ being fixed: when the influential regions have the effect of 'enlarging' distance (left), $\boldsymbol{o}$ move to the inter-class region; when the influential regions have the effect of 'shrinking' distance (right), $\boldsymbol{o}$ move to the intra-class region.



(b) Learn the radii r with $\boldsymbol{M}, \boldsymbol{o}$ being fixed: when the influential regions have the effectiveness of 'enlarging' distance (left), the region lying in the intra-class region (i.e. A1) decreases its size and the region lying in inter-class region (i.e. A2) increases its size; when the influential regions have the effectiveness of 'shrinking' distance (right), r_{A1} increases and r_{A2} decreases.



(c) Learn local metrics $\boldsymbol{M}(A_1), \boldsymbol{M}(A_2)$ with $\boldsymbol{o}, r$ being fixed: Starting from initial metrics $2\boldsymbol{I}$ (left), the metric that is learned in the intra-class region shrinks its distance; the metric that is learned in the inter-class region enlarges the distance in the direction to the decision boundary (right).



(d) Learn $\boldsymbol{o}, r, \boldsymbol{M}(A_1), \boldsymbol{M}(A_2)$: Both influential regions move towards the inter-class region and increase their sizes. The learned local metrics enlarge the distance between samples from different classes.

Fig. 4. Illustration of parameter learning using a toy data set. This figure is best viewed in color.

[12] L. Yang, R. Jin, L. Mummert, R. Sukthankar, A. Goode, B. Zheng, S. C. Hoi, and M. Satyanarayanan, "A boosting framework for visuality-preserving distance metric learning and its application to medical image retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 30–44, 2010.

[13] J. Lu, X. Zhou, Y.-P. Tan, Y. Shang, and J. Zhou, "Neighborhood repulsed metric learning for kinship verification," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 2, pp. 331–345, 2014.

[14] Z. Huang, R. Wang, S. Shan, L. Van Gool, and X. Chen, "Cross Euclidean-to-Riemannian metric learning with application to face recognition from video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[15] B. Wang, G. Wang, K. L. Chan, and L. Wang, "Tracklet association by online target-specific metric learning and coherent dynamics estimation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 3, pp. 589–602, 2017.

[16] A. Frome, Y. Singer, F. Sha, and J. Malik, "Learning globally-consistent local distance functions for shape-based image retrieval and classification," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. IEEE, 2007, pp. 1–8.

[17] J. Wang, A. Kalousis, and A. Woznica, "Parametric local metric learning for nearest neighbor classification," in *Advances in Neural Information Processing Systems*, 2012, pp. 1601–1609.

[18] Y. Huang, C. Li, M. Georgiopoulos, and G. C. Anagnostopoulos, "Reduced-rank local distance metric learning," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2013, pp. 224–239.

[19] J. Bohné, Y. Ying, S. Gentric, and M. Pontil, "Large margin local metric learning," in *European Conference on Computer Vision*. Springer, 2014, pp. 679–694.

[20] Y. Shi, A. Bellet, and F. Sha, "Sparse compositional metric learning," in *AAAI*, 2014, pp. 2078–2084.

[21] S. Saxena and J. Verbeek, "Coordinated local metric learning," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2015, pp. 127–135.

[22] J. St Amand and J. Huan, "Sparse compositional local metric learning," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2017, pp. 1097–1104.

[23] Y. Noh, B. Zhang, and D. Lee, "Generative local metric learning for nearest neighbor classification." *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 1, p. 106, 2018.

[24] L.-A. Gottlieb, A. Kontorovich, and R. Krauthgamer, "Efficient classification for metric data," *Information Theory, IEEE Transactions on*, vol. 60, no. 9, pp. 5750–5759, 2014.

[25] N. Weaver and N. Weaver, *Lipschitz algebras*. World Scientific, 1999.

[26] Y. Mu, W. Ding, and D. Tao, "Local discriminative distance metrics ensemble learning," *Pattern Recognition*, vol. 46, no. 8, pp. 2337–2349, 2013.

[27] Y. Ying and P. Li, "Distance metric learning with eigenvalue optimization," *Journal of Machine Learning Research*, vol. 13, no. Jan, pp. 1–26, 2012.

[28] M. Kusner, S. Tyree, K. Weinberger, and K. Agrawal, "Stochastic neighbor compression," in *International Conference on Machine Learning*, 2014, pp. 622–630.

[29] M. Perrot and A. Habrard, "Regressive virtual metric learning," in *Advances in Neural Information Processing Systems*, 2015, pp. 1810–1818.

[30] P. Zadeh, R. Hosseini, and S. Sra, "Geometric mean metric learning," in *International Conference on Machine Learning*, 2016, pp. 2464–2471.

[31] B. Nguyen, C. Morell, and B. De Baets, "Supervised distance metric learning through maximization of the jeffrey divergence," *Pattern Recognition*, vol. 64, pp. 215–225, 2017.