Metric Learning with Lipschitz Continuous Functions

Mingzhi Dong

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of

University College London.

Department of Statistical Science

University College London

December 2, 2018

I, Mingzhi Dong, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

Classification is a fundamental problem in the field of statistical machine learning. In classification, issues of nonlinear separability and multimodality are frequently encountered even in relatively small data sets. Distance-based classifiers, such as the nearest neighbour (NN) classifier which classifies a new instance by computing distances between this instance and the training instances, have been found useful to deal with nonlinear separability and multimodality. However, the performance of distance-based classifiers heavily depends on the underlying distance metric, so it is valuable to study metric learning, which enables the algorithms to automatically learn a suitable metric from available data.

In this thesis, I discuss the topic of metric learning with Lipschitz continuous functions. The classifiers are restricted to have certain Lipschitz continuous properties, so that the performance guarantee of classifiers, which could be described by probably approximately correct (PAC) learning bounds, would be obtained.

In Chapter 2, I propose a framework in which the metric would be learned with the criterion of large margin ratio. Both inter-class margin and intra-class dispersion are considered in the criterion, so as to enhance the generalisation ability of classifiers. Some well-known metric learning algorithms can be shown as special cases of the proposed framework.

In Chapter 3, I suggest that multiple local metrics would be learned to deal with multimodality problems. I define an intuitive distance with local metrics and influential regions, and subsequently propose a novel local metric learning method for distance-based classification. The 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.

In Chapter 4, metric learning with instance extraction (MLIE) is discussed. A big drawback of the NN classifier is that it needs to store all training instances, hence it suffers from problems of storage and computation. Therefore, I propose an algorithm to extract a small number of useful instances, which would reduce the costs of storage as well as the computation costs during the test stage. Furthermore, the proposed instance extraction method could be understood as an elegant way to do local linear classification, i.e. simultaneously learn the positions of local areas and the linear classifiers inside the local areas.

In Chapter 5, based on an algorithm-dependent PAC bound, another algorithm of MLIE is proposed. Besides the Lipschitz continuous requirement with respect to the parameter, the Lipschitz continuous requirement with respect to the gradient of parameter will also be considered. Therefore, smooth classifiers and smooth loss functions are proposed in this chapter.

The classifiers proposed in Chapter 2 and Chapter 3 have bounded values of $\mathrm{lip}(h \leftarrow \boldsymbol{x})$ with a PAC bound, where $\mathrm{lip}(h \leftarrow \boldsymbol{x})$ denotes the Lipschitz constant of the function with respect to the input space $\mathcal{X}$. The classifiers proposed in Chapter 4 enjoys the bounded value of $\mathrm{lip}(h \leftarrow \boldsymbol{\theta})$ with a tighter PAC bound, where $\mathrm{lip}(h \leftarrow \boldsymbol{\theta})$ denotes the Lipschitz constant of the function with respect to the input space Θ. In Chapter 5, to consider the property of the optimisation algorithm simultaneously, an algorithm-dependent PAC bound based on Lipschitz smoothness is derived.

Acknowledgements

I would like to express my sincere gratitude to my supervisor Dr. Jing-Hao Xue. During my Ph.D. period, he has been providing guidance for my research and without his detailed help and support, this Ph.D thesis would not have been possible. I feel very lucky to be a Ph.D. student of Dr. Xue and I would like to thank him for all his invaluable suggestions which help me overcome numerous difficulties. He has always been a great supervisor, advisor and friend of mine.

I also would like to thank my second supervisor Dr. Afzal Siddiqui for his comments and encouragement on the research work.

My sincere thanks also go to Prof. Li Shang, for his suggestions on the research directions and discussions on research topics.

I would like to thank Prof. Yang Wu, for his discussions on research work and his warm host of my stay in Japan.

I also greatly appreciate the helpful discussions with my collaborators, Kunkun Pang, Yujiang Wang, Xiaochen Yang, Rui Zhu and Mengyuan Chen.

I gratefully acknowledge the funding sources during my PhD study. I was funded by the Overseas Research Scholarship of UCL and the CSC scholarship offered by Chinese government.

I thank my friends, Liang Yin, Yanhui Yang, Zhenhua Zhang, Xiaoou Lu, Yurong Ling, for all their help and support.

Lastly, I would like to express my sincere gratitude to my parents and elder brother, for all their love and support throughout my life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Over the past few decades, *classification* has always been one of the most fundamental problems in the field of statistical machine learning. In this chapter, the thesis starts by reviewing the definition of classification in Section 1.1. The issues with respect to *learnability* are discussed in Section 1.2, which illustrates whether or not an algorithm is able to learn the optimal classifier inside a hypothesis set $\mathcal{H}$. In Section 1.2.1, the definitions of PAC learnability and agnostic PAC learnability are introduced. In Section 1.2.2, uniform convergence, which is a sufficient condition for (agnostic) PAC learnability is reviewed. In Section 1.2.3, some examples of PAC learning bounds based on uniform convergence are illustrated. In Section 1.2.4, we explain how to control the generalisation ability of a learnable algorithm by adding the regularisation terms based on union convergence bounds. Then, the definition and some examples of metric learning are illustrated in Section 1.3. After that, the definitions and some concepts with respect to Lipschitz functions are reviewed in Section 1.4. Finally, the structure of the thesis is presented in Section 1.5.

1.1 Classification

Learning is the process of summarising general rules from given examples. Statistical machine learning, as defined by Arthur Samuel, is a "field of study that gives computers the ability to learn without being explicitly programmed" [60]. Based on the feedback information provided, machine learning tasks are mainly divided into three categories [54]:

- Supervised Learning. The system would supply the algorithm with the desired outputs of some input instances. Then the algorithm learns a general rule that maps inputs to outputs.

- Unsupervised Learning. No desired outputs are given and the algorithm should find the rule from the input data itself.

- Reinforcement Learning. In a dynamic environment, the algorithm learns to take sequential actions so as to maximise the reward of achieving a goal.

In this thesis, I will primarily focus on a supervised learning problem: classification.

The problem of *classification* is about predicting to which set of categories a new observation belongs, based on a given set of instances whose category information is known. In classification problems, an input ($\boldsymbol{x}_i \in \mathcal{X}$) and its corresponding output/label ($y_i \in \mathcal{Y}$) form a training pair ($\boldsymbol{z}_i = (\boldsymbol{x}_i, y_i) \in \mathcal{Z}$). The set which consists of all training pairs is called the *training set* $\boldsymbol{z}^n$, where $\boldsymbol{z}^n = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n\}$ and n denotes the number of training instances. During the *training process*, a learner $\mathcal{Z}^n \to \mathcal{G}$ seeks an optimal function $g \in \mathcal{G}$ based on all training pairs, where $\mathcal{G}$ is the set containing all candidate functions of $\mathcal{X} \to \mathcal{Y}$. After that, during the *test process*, a new input instance $\boldsymbol{x}$ is mapped to the output space via $g(\boldsymbol{x})$.

Throughout the thesis, unless specified otherwise, the set of categories is assumed to be the binary set $\{-1, 1\}$. Meanwhile, I assume $\mathcal{X} \subseteq \mathbb{R}^D$, where D denotes the dimension of the input space. In binary classification, the function g is usually based on another function $h : \mathcal{X} \to \mathbb{R}$ which maps $\boldsymbol{x}$ to a real value, where $h \in \mathcal{H}$ and $\mathcal{H}$ is called the *hypothesis set*. The relationship between g and h is defined as follows:

$$g(\boldsymbol{x}) = \mathrm{sign}[h(\boldsymbol{x})] = \begin{cases} 1 & \text{if } h(\boldsymbol{x}) \geq 0 \\ -1 & \text{if } h(\boldsymbol{x}) < 0 \end{cases},$$

where $\mathrm{sign}[\cdot]$ denotes the sign function and returns the sign of a real number.

The risk/loss/error of a classifier h for a training pair $\boldsymbol{z}$ could be measured by a risk function $r(\boldsymbol{z}, h)$ or be equivalently written as a loss function $l(h(\boldsymbol{x}); y)$. Then $R_n(\boldsymbol{z}^n, h) := \frac{1}{n}\sum_i r(\boldsymbol{z}_i, h) := \frac{1}{n}\sum_i l(h(\boldsymbol{x}_i); y_i)$ is called the *training error* or *empirical risk*, which indicates the training loss given the classifier h for a set of training instances $\boldsymbol{x}^n$. Similarly, $R(h) := \mathbb{E}_{\boldsymbol{z}'} r(\boldsymbol{z}', h) := \mathbb{E}_{\boldsymbol{z}'} l(h(\boldsymbol{x}'); y')$ is called the *test error* or *expected risk*, which indicates the expected value of test loss given a test input pair $\boldsymbol{z}' = (\boldsymbol{x}', y')$ into the classifier h. The gap between the training error and the test error, i.e. $R(h) - R_n(\boldsymbol{z}^n, h)$, is called the *generalisation gap*.

1.2 Learnability

1.2.1 PAC and Agnostic PAC Learnability

Definition 1. [57, 65] A hypothesis class $\mathcal{H}$ is *Probably Approximately Correct (PAC) learnable* if there exist a function $n_{\mathcal{H}}^{L} : (0, 1)^2 \to \mathbb{N}$ and a learning algorithm with the following property: For every $\epsilon, \delta \in (0, 1)$, for every distribution $\mathcal{D}_{\mathcal{X}}$ over $\mathcal{X}$, and for every target function $g \in \mathcal{G}$, if there exists an $h^* \in \mathcal{H}$ which returns the same classification result as g, then when running the learning algorithm on $n \geq n_{\mathcal{H}}^{L}(\epsilon, \delta)$ independent and identically distributed (i.i.d.) instances generated by $\mathcal{D}_{\mathcal{X}}$ and labelled by g, the algorithm returns a hypothesis $\hat{h}$, such that, with probability at least $1 - \delta$, $R(\hat{h}) \leq \epsilon$, which can be equivalently written as

$$\mathbb{P}_{\boldsymbol{x}^n}\left(R(\hat{h}) \leq \epsilon\right) \geq 1 - \delta,$$

or

$$\mathbb{P}_{\boldsymbol{x}^n}\left(\mathbb{E}_{\boldsymbol{x}'}\left[l\big(\hat{h}(\boldsymbol{x}'); g(\boldsymbol{x}')\big)\right] \leq \epsilon\right) \geq 1 - \delta,$$

where the probability is over $\boldsymbol{x}_n$ and $\hat{h}$ is a random variable related to $\boldsymbol{x}_n$.

In the definition of PAC learnability, training and test instances should come from the same distribution, but this distribution is unknown to the learner and could be any distribution $\mathcal{D}$ over $\mathcal{X}$. PAC contains two kinds of approximations: 1) *Approximately* correct: ϵ denotes the difference between the output classifier and the optimal one, which indicates that we cannot expect a learner to learn a concept exactly; and 2) *Probably* correct: $1 - \delta$ denotes how likely the event of $R(\hat{h}) \leq \epsilon$

happens, which indicates that we cannot always expect a close approximation to happen. An expectation for a good classifier is that with high probability $(1-\delta)$ it will learn a close approximation ($R(\hat{h}) \leq \epsilon$) to the target.

The function $n_{\mathcal{H}}^{L} : (0,1)^2 \to \mathbb{N}$ is a function of ϵ and δ. It determines how many training pairs are required to guarantee a $(\epsilon, 1-\delta)$-solution, that is, the *sample complexity* of learning $\mathcal{H}$.

Two strong assumptions are imposed in the definition of PAC learnability. First, the target function g indicates y_i is determined given $\boldsymbol{x}_i$. However, label noise exists in real cases. In other words, for the same $\boldsymbol{x}_i$, there is a positive probability for both $y_i = 1$ and $y_i = -1$ to happen. Second, the existence of an optimal hypothesis, that is 'an $h^* \in \mathcal{H}$ which returns the same classification result as g', may require $\mathcal{H}$ to be very large. To solve these problems, agnostic PAC learnability has been proposed.

Definition 2. [57, 23] A hypothesis class $\mathcal{H}$ is *agnostic PAC learnable* or has *agnostic PAC learnability* if there exist a function $n_{\mathcal{H}}^{AL} : (0,1)^2 \to \mathbb{N}$ and a learning algorithm with the following property: For every $\epsilon, \delta \in (0,1)$ and for every distribution $\mathcal{D}_{\mathcal{Z}}$ over $\mathcal{Z}$, when running the learning algorithm on $n \geq n_{\mathcal{H}}^{AL}(\epsilon, \delta)$ i.i.d. instances generated by $\mathcal{D}_{\mathcal{Z}}$, the algorithm returns a hypothesis $\hat{h}$ which satisfies the following *agnostic PAC learning bound*: with probability at least $1-\delta$,

$$R(\hat{h}) - \min_{h \in \mathcal{H}} R(h) \leq \epsilon.$$

The above agnostic PAC learning bound can be equivalently written as

$$\mathbb{P}_{\boldsymbol{z}^n}\Big(R(\hat{h}) - \min_{h \in \mathcal{H}} R(h) \leq \epsilon\Big) \geq 1-\delta,$$

or more explicitly

$$\mathbb{P}_{\boldsymbol{z}^n}\Big(\mathbb{E}_{\boldsymbol{z}'}\big[l\big(\hat{h}(\boldsymbol{x}'); y\big)\big] - \min_{h \in \mathcal{H}} \mathbb{E}_{\boldsymbol{z}'}\big[l\big(h(\boldsymbol{x}'); y\big)\big] \leq \epsilon\Big) \geq 1-\delta.$$

where the probability is over $\boldsymbol{z}^n$ and $\hat{h}$ is a random variable related to $\boldsymbol{z}^n$.

Unless specified otherwise, the discussion of learnability in the thesis would be restricted to agnostic PAC learnability. In the following discussion, PAC learnability, PAC learnable and PAC learning bound will represent agnostic PAC learnability, agnostic PAC learnable and agnostic PAC learning bound respectively.

1.2.2 Uniform Convergence

Uniform convergence is a widely used sufficient condition for (agnostic) PAC learnability. In this section, the definition of uniform convergence is introduced first. Then, the relationship between uniform convergence and learnability is illustrated.

Definition 3. [57] A hypothesis class $\mathcal{H}$ has the *uniform convergence* property if there exists a function $n_{\mathcal{H}}^{UC} : (0,1)^2 \rightarrow \mathbb{N}$ with the following property: For every $\epsilon, \delta \in (0,1)$ and for every probability distribution $\mathcal{D}_{\mathcal{Z}}$ over $\mathcal{Z}$, if $\boldsymbol{z}^n$ is a sample of $n \geq n_{\mathcal{H}}^{UC}(\epsilon, \delta)$ i.i.d. instances drawn from $D_{\mathcal{Z}}$, then the following *uniform convergence bound* holds: with probability at least $1-\delta$,

$$\max_{h \in \mathcal{H}} |R(h) - R_n(\boldsymbol{z}^n, h)| \leq \epsilon,$$

or

$$\forall h \in \mathcal{H}, |R(h) - R_n(\boldsymbol{z}^n, h)| \leq \epsilon,$$

which can be equivalently written as

$$\mathbb{P}_{\boldsymbol{z}^n}\Big(\max_{h \in \mathcal{H}} |R(h) - R_n(\boldsymbol{z}^n, h)| \leq \epsilon\Big) \geq 1 - \delta.$$

Lemma 1. [57] If a hypothesis set $\mathcal{H}$ has the uniform convergence property with a function $n_{\mathcal{H}}^{UC}$, then $\mathcal{H}$ is agnostic PAC learnable with the sample complexity function $n_{\mathcal{H}}^{AL}(\epsilon, \delta) \leq n_{\mathcal{H}}^{UC}(\epsilon/2, \delta)$. Furthermore, in this case, $\mathrm{ERM}_{\mathcal{H}}$ is a successful agnostic PAC learner for $\mathcal{H}$, where $\mathrm{ERM}_{\mathcal{H}}$ denotes the empirical risk minimisation strategy inside the set of $\mathcal{H}$ that $\hat{h} \in \mathrm{argmin}_{h \in \mathcal{H}} R_n(\boldsymbol{z}^n, h)$.

Proof. For a sample $\boldsymbol{z}^n$, the relationship between $\max_{h \in \mathcal{H}} |R(h) - R_n(\boldsymbol{z}^n, h)| \leq \epsilon/2$ and $R(\hat{h}) \leq \min_{h \in \mathcal{H}} R(h) + \epsilon$ will be illustrated as follows, where $\hat{h}$ is obtained with $\mathrm{ERM}_{\mathcal{H}}$:

$$\begin{aligned}
&\forall \boldsymbol{z}^n, \quad \max_{h\in\mathcal{H}} |R(h) - R_n(\boldsymbol{z}^n, h)| \le \epsilon/2 \\
&\qquad \Rightarrow \forall h \in \mathcal{H}, R(\hat{h}) \le_{(a)} R_n(\boldsymbol{z}^n, \hat{h}) + \epsilon/2 = \min_{h\in\mathcal{H}} R_n(\boldsymbol{z}^n, h) + \epsilon/2 \\
&\qquad \le_{(b)} R_n(\boldsymbol{z}^n, h) + \epsilon/2 \le_{(c)} R(h) + \epsilon,
\end{aligned}$$

where inequality (a) is due to $\max_{h\in\mathcal{H}} |R(h) - R_n(\boldsymbol{z}^n, h)| \le \epsilon/2$; inequality (b) is due to $\hat{h}$ is obtained by $\mathrm{ERM}_{\mathcal{H}}$, so that $R_n(\boldsymbol{z}^n, \hat{h}) = \min_{h\in\mathcal{H}} R_n(\boldsymbol{z}^n, h) \le R_n(\boldsymbol{z}^n, h)$; inequality (c) is again due to $\max_{h\in\mathcal{H}} |R(h) - R_n(\boldsymbol{z}^n, h)| \le \epsilon/2$. Therefore

$$1 - \delta \le \mathbb{P}_{\boldsymbol{z}_n}\Big(\max_{h\in\mathcal{H}} |R(\hat{h}) - R_n(\boldsymbol{z}^n, h)| \le \epsilon/2\Big) \le \mathbb{P}_{\boldsymbol{z}^n}\Big(R(\hat{h}) \le \min_{h\in\mathcal{H}} R(h) + \epsilon\Big).$$

□

Since uniform convergence is a sufficient condition for PAC learnability, in the following discussion, we will refer to the uniform convergence bound as a PAC learning bound.

1.2.3 PAC Learning Bounds based on Uniform Convergence

As illustrated in lemma 1, $\mathcal{H}$ is (agnostic) PAC learnable with the learner $\mathrm{ERM}_{\mathcal{H}}$ as long as it has the uniform convergence property. We will now use the uniform convergence property to obtain PAC learning bounds in the following cases.

1.2.3.1 Finite Hypothesis Set Bounds

We can use the Hoeffding's inequality, one of concentration inequalities, to obtain the PAC learning bounds of finite hypothesis sets directly. Let $\mathcal{H} = \{h_1, \ldots, h_K\}$ be a hypothesis set with K functions and suppose the risk function $r(\boldsymbol{z}, h)$ is bounded by the interval $[0, C_r]$,

$$\begin{aligned}
&\mathbb{P}_{\boldsymbol{z}^n}[\max_{h\in\mathcal{H}} |R(h) - R_n(\boldsymbol{z}^n, h)| > \epsilon] \\
=&\mathbb{P}_{\boldsymbol{z}^n}[\exists h \in \{h_1, \dots, h_K\}, |R(h) - R_n(\boldsymbol{z}^n, h)| > \epsilon] \\
\leq&\sum_{i=1}^{K} \mathbb{P}_{\boldsymbol{z}^n}[|R(h_i) - R_n(\boldsymbol{z}^n, h_i)| > \epsilon] \\
\leq&2K\exp(-\frac{2n\epsilon^2}{C_r^2}).
\end{aligned}$$

where the first inequality is based on the probability of the union of events and the second inequality is based on the Hoeffding's inequality, as illustrated in Appendix 1.6.1. Set $\delta = 2K\exp(-\frac{2n\epsilon^2}{C_r^2})$. The obtained PAC learning bound is equivalently written as follows: with probability at least $1 - \delta$

$$\forall h \in \mathcal{H}, R(h) \leq R_n(\boldsymbol{z}^n, h) + C_r\sqrt{\frac{\ln 2K + \ln(1/\delta)}{2n}}.$$

1.2.3.2 Vapnik-Chervonenkis Bounds

In practical algorithms, the hypothesis sets have infinite elements. The Vapnik-Chervonenkis (VC) dimension [66] is a complexity measure for infinite hypothesis sets.

Definition 4. [66] The *VC dimension* of a class $\mathcal{H}$ is the largest integer V such that

$$S_{\mathcal{H}}(V) = 2^V,$$

and

$$S_{\mathcal{H}}(V) = \max_{\boldsymbol{x}_1,\dots,\boldsymbol{x}_V} \text{card}\Big(\{(\text{sign}[h(\boldsymbol{x}_1)], \dots, \text{sign}[h(\boldsymbol{x}_V)]) : h \in \mathcal{H}\}\Big),$$

where $\text{card}(\cdot)$ denotes the cardinality of a set.

In other words, the VC dimension of a hypothesis set $\mathcal{H}$ is the largest number of training instances that $\mathcal{H}$ can correctly classify whatever the labels of these instances are. With the VC dimension V of $\mathcal{H}$, the VC bound could be represented as follows [66]: with probability at least $1 - \delta$,

$$\forall h \in \mathcal{H}, R(h) \leq R_n(h) + 2\sqrt{2\frac{V \log \frac{2en}{V} + 2\log\frac{2}{\delta}}{N}}.$$

1.2.3.3 Rademacher Complexity Bounds

For general hypothesis sets, the VC dimension is relatively hard to compute. A practically common complexity measure is the Rademacher complexity.

Definition 5. [46] Let $\boldsymbol{\epsilon}^n = \{\epsilon_1, \dots \epsilon_n\}$ be i.i.d. ±1-valued random variables with $P(\epsilon_i = +1) = P(\epsilon_i = -1) = \frac{1}{2}$. $\boldsymbol{z}^n = \{\boldsymbol{z}_1, \dots, \boldsymbol{z}_n\}$ are i.i.d. samples. The *empirical Rademacher complexity* is defined as

$$\hat{\text{Rad}}_n(\mathcal{H}) = \mathbb{E}_{\boldsymbol{\epsilon}^n}\left[\max_{h\in\mathcal{H}} \frac{1}{n}\sum_i \epsilon_i h(\boldsymbol{z}_i)\middle|\boldsymbol{z}^n\right];$$

and the *Rademacher complexity* is defined as

$$\text{Rad}(\mathcal{H}) = \mathbb{E}_{\boldsymbol{z}^n}\left[\hat{\text{Rad}}_n(\mathcal{H})\right].$$

Theorem 1. [46] With probability at least $1 - \delta$ the following bounds hold:

$$\forall h \in \mathcal{H}, R(h) \leq R_n(\boldsymbol{z}^n, h) + 2\,\text{Rad}(\phi \circ \mathcal{H}) + \sqrt{\frac{\ln\frac{1}{\delta}}{2n}},$$

and

$$\forall h \in \mathcal{H}, R(h) \leq R_n(\boldsymbol{z}^n, h) + 2\hat{\text{Rad}}_n(\phi \circ \mathcal{H}) + 3\sqrt{\frac{\ln\frac{2}{\delta}}{2n}},$$

where $\phi : \mathbb{R} \to \mathbb{R}$ denotes the loss function $l(h(\boldsymbol{x}); y)$; $\circ$ denotes the composition of functions.

1.2.4 Regularisation as a Practical Strategy

Based on Lemma 1, if a hypothesis set $\mathcal{H}$ has the uniform convergence property, then $\text{ERM}_{\mathcal{H}}$ is a successful agnostic PAC learner. In addition, we could control the generalisation ability of the hypothesis set $\mathcal{H}$ through some regularisation terms derived from the uniform convergence bound.

$\mathrm{ERM}_{\mathcal{H}}$ has the following optimisation function:

$$\begin{aligned} \min_{h} \quad & R_n(\boldsymbol{z}_n, h) \\ s.t. \quad & h \in \mathcal{H}. \end{aligned}$$

For parametric functions, it is equivalent to

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & R_n(\boldsymbol{z}_n, h_{\boldsymbol{\theta}}) \\ s.t. \quad & \boldsymbol{\theta} \in \Theta. \end{aligned}$$

After we obtain the uniform convergence PAC bound, suppose the generalisation gap is related to the parameter $\boldsymbol{\theta}$ and there exists a function $\mathcal{P} : \Theta \to \mathbb{R}$ such that controlling $\mathcal{P}(\boldsymbol{\theta})$ could efficiently control the upper bound of the gap, then the optimisation function could be set as follows:

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & R_n(\boldsymbol{z}_n, h_{\boldsymbol{\theta}}) \\ s.t. \quad & \mathcal{P}(\boldsymbol{\theta}) \leq C, \end{aligned} \tag{1.1}$$

where C is a constant and the restriction $\mathcal{P}(\boldsymbol{\theta}) \leq C$ is imposed in order for the hypothesis set to have a relatively good generalisation ability. A smaller $\mathcal{P}(\boldsymbol{\theta})$ will result in a smaller generalisation gap. Problem (1.1) is a (agnostic) learnable algorithm inside the hypothesis set of $\mathcal{H}_\theta = \{h_{\boldsymbol{\theta}}; \mathcal{P}(\boldsymbol{\theta}) \leq C\}$ and it can be equivalently written as follows[1]:

$$\min_{\boldsymbol{\theta}} \quad R_n(\boldsymbol{z}_n, h_{\boldsymbol{\theta}}) + \lambda \mathcal{P}(\boldsymbol{\theta}), \tag{1.2}$$

where λ is a trade-off parameter between the empirical risk $R_n(\boldsymbol{z}_n, h_{\boldsymbol{\theta}})$ and the generalisation ability $\mathcal{P}(\boldsymbol{\theta})$. A detailed discussion on the equivalence between the two optimisation problems is provided in Appendix 1.6.2.

Based on the discussion above, adding a regularisation term in (1.1) is a practical way to control the generalisation ability of the learner $\mathrm{ERM}_{\mathcal{H}}$. One critical issue here is to determine the form of $\mathcal{P}(\boldsymbol{\theta})$ based on a PAC learning bounds.

[1] 'Equivalent' here denotes the equivalent necessary condition for the optimal $\boldsymbol{\theta}$.

1.3 Metric Learning Classifiers

A large number of classifiers have been proposed over the past few decades. Among those, the nearest neighbour (NN) classifier is one of the oldest and simplest methods. For a new instance, NN calculates distances to all training instances and returns the label of its nearest neighbour. The performance of NN depends on the distance or similarity metrics and handcrafting these metrics is generally difficult. Therefore, metric learning has been proposed and it enables the algorithms to learn a metric from the data automatically.

1.3.1 Nearest Neighbour

The nearest neighbour (NN) classifier, based on non-parametric estimation, is one of the most intuitive classifiers. In NN classifier, it remembers all the training instances and labels $\{\boldsymbol{z}_i = (\boldsymbol{x}_i, y_i), i = 1, \dots, n\}$. At the same time, the space $\mathcal{X}$ is endowed with a distance metric $\rho_{\mathcal{X}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Given a test instance $\boldsymbol{x}$, NN would assign the test instance with the same label as its nearest neighbour $\boldsymbol{x}_k$ in the training set:

$$y = y_k, \qquad \text{where } k = \operatorname*{argmin}_i \rho_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}_i), i = 1 \dots n.$$

The conventional NN is a type of the so-called ‘lazy learning’ because it does not learn any parameter.

1.3.2 Metric Learning

We start by reviewing the basic terminology of metric [76].

Definition 6. A mapping function $\rho : \mathcal{X} \times \mathcal{X} \to [0, \infty)$ is called a *metric* if for all vectors $\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_k \in \mathcal{X}$, it satisfies the following properties:

- $\rho(\boldsymbol{x}_i, \boldsymbol{x}_j) + \rho(\boldsymbol{x}_j, \boldsymbol{x}_k) \geq \rho(\boldsymbol{x}_i, \boldsymbol{x}_k)$ (triangle inequality);
- $\rho(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0$ (non-negativity);
- $\rho(\boldsymbol{x}_i, \boldsymbol{x}_j) = \rho(\boldsymbol{x}_j, \boldsymbol{x}_i)$ (symmetry);
- $\rho(\boldsymbol{x}_i, \boldsymbol{x}_j) = 0 \Leftrightarrow \boldsymbol{x}_i = \boldsymbol{x}_j$ (identity of indiscernibles).

If a mapping only satisfies the first three properties, that is the distance between two distinct points could be zero, it is called *pseudometric*. In classification tasks, we mainly deal with pseudometric and refer to pseudometric as metric in the following discussion for simplicity [2].

A frequently adopted metric is the Mahalanobis distance:

$$d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{M} (\boldsymbol{x}_i - \boldsymbol{x}_j)}, \quad \boldsymbol{M} \in \boldsymbol{M}_+,$$

where $\boldsymbol{M}_+$ denotes the set of positive semi-definite matrices. The Mahalanobis distance is equivalent to computing the Euclidean distance after a linear transformation $\boldsymbol{x}' = \boldsymbol{L}\boldsymbol{x}$:

$$d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \|\boldsymbol{L}(\boldsymbol{x}_i - \boldsymbol{x}_j)\|_2 = d(\boldsymbol{x}'_i, \boldsymbol{x}'_j), \quad \boldsymbol{M} = \boldsymbol{L}^T \boldsymbol{L},$$

where $d(\boldsymbol{x}'_i, \boldsymbol{x}'_j)$ denotes the Euclidean distance between $\boldsymbol{x}'_i$ and $\boldsymbol{x}'_j$.

Learning the metric with a convex optimisation problem was first proposed in [78], where the metric is found through the following optimisation problem with a convex objective function:

$$\begin{array}{ll} \max\limits_{\boldsymbol{M}} & \sum_{ij}(1 - y_{ij}) d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ s.t. & \sum_{ij} y_{ij} d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \leq 1 \\ & \boldsymbol{M} \in \boldsymbol{M}_+, \end{array} \tag{1.3}$$

where $y_{ij} = 1$ if $y_i = y_j$ and $y_{ij} = 0$ otherwise.

The algorithm aims to learn a metric $\boldsymbol{M}$ such that the distances between the instance pairs with the same label would be equal to or less than 1 ($\sum_{ij} y_{ij} d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \leq 1$) and the distances between instance pairs with different labels would be as large as possible ($\max_{\boldsymbol{M}} \sum_{ij}(1-y_{ij}) d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$). In the optimisation problem, the squared Mahalanobis distance is used to guarantee the constraint to be a convex one. This metric learning strategy has improved the performance of NN significantly [78, 76].

[2] In local metric learning problems, a function ρ without triangle inequality or even symmetry property may be called a 'metric' following the local metric learning references, such as [70, 5].

After that, in [56], the hinge loss and Frobenius regularisation have been introduced into the research of metric learning:

$$\begin{aligned} \min_{\boldsymbol{W},\boldsymbol{\xi}} \quad & \|\boldsymbol{M}\|_F^2 + \lambda \textstyle\sum_{ijk} \xi_{ijk} \\ s.t. \quad & d_{\boldsymbol{M}}^2(\boldsymbol{x}_i, \boldsymbol{x}_k) - d_{\boldsymbol{M}}^2(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 1 - \xi_{ijk} \\ & \xi_{ijk} \geq 0, \boldsymbol{M} = \boldsymbol{A}^T \boldsymbol{W} \boldsymbol{A}, \boldsymbol{W} \geq 0 \\ & i = 1, \ldots, n, j \rightarrow i, k \nrightarrow i, \end{aligned} \tag{1.4}$$

where $d_{\boldsymbol{M}}^2(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{M} (\boldsymbol{x}_i - \boldsymbol{x}_j) = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{A}^T \boldsymbol{W} \boldsymbol{A} (\boldsymbol{x}_i - \boldsymbol{x}_j)$, $\boldsymbol{A}$ is a fixed matrix determined via prior knowledge, $\boldsymbol{W}$ is a diagonal matrix which will be optimised and $\boldsymbol{W} \geq 0$ denotes all elements of $\boldsymbol{W}$ should be greater than or equal to 0; $\boldsymbol{\xi} = \{\xi_{ijk}\}$ and ξ_{ijk} denotes the value of $\max[d_{\boldsymbol{M}}^2(\boldsymbol{x}_i, \boldsymbol{x}_j) + 1 - d_{\boldsymbol{M}}^2(\boldsymbol{x}_i, \boldsymbol{x}_k), 0]$, which indicates that if $d_{\boldsymbol{M}}^2(\boldsymbol{x}_i, \boldsymbol{x}_k)$ is not larger than $d_{\boldsymbol{M}}^2(\boldsymbol{x}_i, \boldsymbol{x}_j) + 1$, there will be a loss of $d_{\boldsymbol{M}}^2(\boldsymbol{x}_i, \boldsymbol{x}_j) + 1 - d_{\boldsymbol{M}}^2(\boldsymbol{x}_i, \boldsymbol{x}_k)$; $\|\boldsymbol{M}\|_F^2$ is the square of the Frobenius norm of $\boldsymbol{M}$; λ is a constant which controls the trade-off between the two terms of the objective function; $j \rightarrow i$ indicates $\boldsymbol{x}_j$ is the *target neighbour* of $\boldsymbol{x}_i$ and $k \nrightarrow i$ indicates $\boldsymbol{x}_k$ is $\boldsymbol{x}_i$'s *impostor neighbour*.

A heuristic strategy for choosing target neighbours is to pick $\boldsymbol{x}_i$'s C nearest neighbours from the same class measured via the Euclidean metric, that is,

$$j \in \Big\{ b \mid d(\boldsymbol{x}_b, \boldsymbol{x}_i) \leq \boldsymbol{U}_{(C)}, \boldsymbol{U} = \{ d(\boldsymbol{x}_a, \boldsymbol{x}_i) \mid \boldsymbol{x}_a \in \boldsymbol{x}^n, y_a = y_i, \boldsymbol{x}_a \neq \boldsymbol{x}_i \} \Big\},$$

where $\boldsymbol{x}^n$ denotes the set of training instances, each element of the set $\boldsymbol{U}$ is the distance between $\boldsymbol{x}_i$ and a training instance with the same label, $\boldsymbol{U}_{(C)}$ denotes the Cth smallest element of $\boldsymbol{U}$. Similarly, a heuristic strategy for choosing impostor neighbours is to pick $\boldsymbol{x}_i$'s C nearest neighbours from the different class measured via the Euclidean metric, that is,

$$k \in \Big\{ b \mid d(\boldsymbol{x}_b, \boldsymbol{x}_i) \leq \boldsymbol{V}_{(C)}, \boldsymbol{V} = \{ d(\boldsymbol{x}_a, \boldsymbol{x}_i) \mid \boldsymbol{x}_a \in \boldsymbol{x}^n, y_a \neq y_i \} \Big\},$$

where each element of the set $\boldsymbol{V}$ is the distance between $\boldsymbol{x}_i$ and a training instance with the different label.

Another similar and widely cited metric learning algorithm is the large margin nearest neighbour (LMNN) [76] and it solves the following optimisation problem:

$$\begin{array}{ll} \min\limits_{\boldsymbol{M},\boldsymbol{\xi}} & \sum_{ij} d^2_{\boldsymbol{M}}(\boldsymbol{x}_i - \boldsymbol{x}_j) + \lambda \sum_{ijk} \xi_{ijk} \\ s.t. & d^2_{\boldsymbol{M}}(\boldsymbol{x}_i - \boldsymbol{x}_k) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i - \boldsymbol{x}_j) \geq 1 - \xi_{ijk} \\ & \qquad \xi_{ijk} \geq 0, \boldsymbol{M} \in \boldsymbol{M}_+ \\ & \qquad i = 1, \ldots, n, j \rightarrow i, k \nrightarrow i, \end{array} \tag{1.5}$$

where $\boldsymbol{M}$ is the metric to be optimised. The fundamental difference between (1.5) and (1.4) is the regularisation term. In (1.4), the authors adopted the matrix Frobenius norm. In (1.5), the authors adopted

$$\sum_{ij} d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

to bring target neighbours closer. From the above discussion, we can see that the regularisation term is critical for metric learning.

1.4 Lipschitz Functions

Most of the hypothesis set $\mathcal{H}$ enjoys the Lipschitz continuous property. In this section, the definition of Lipschitz functions is reviewed and two kinds of Lipschitz constant, which are frequently used in the thesis, are defined. After that, we discuss some properties relating to classification, such as how we can construct more sophisticated Lipschitz functions from basic Lipschitz functions.

1.4.1 Definition of Lipschitz Functions

Definition 7. [74] Let $(\mathcal{U}, \rho_{\mathcal{U}})$, $(\mathcal{V}, \rho_{\mathcal{V}})$ be two metric spaces. A function $h : \mathcal{U} \rightarrow \mathcal{V}$ is called *Lipschitz continuous* if $\exists L < \infty, \forall \boldsymbol{u}_1, \boldsymbol{u}_2 \in \mathcal{U}$,

$$\rho_{\mathcal{V}}(h(\boldsymbol{u}_1), h(\boldsymbol{u}_2)) \leq L \rho_{\mathcal{U}}(\boldsymbol{u}_1; \boldsymbol{u}_2).$$

The *Lipschitz constant of a Lipschitz function* h is

$$\begin{aligned}\operatorname{lip}(h;\mathcal{V}\leftarrow\mathcal{U}) =& \min\{L\in\mathbb{R}|\forall \boldsymbol{u}_1,\boldsymbol{u}_2\in\mathcal{U},\boldsymbol{u}_1\neq\boldsymbol{u}_2,\\ & \rho_{\mathcal{V}}(h(\boldsymbol{u}_1),h(\boldsymbol{u}_2))\leq L\rho_{\mathcal{U}}(\boldsymbol{u}_1,\boldsymbol{u}_2)\}\\ =& \max_{\boldsymbol{u}_1,\boldsymbol{u}_2\in\mathcal{U}:\boldsymbol{u}_1\neq\boldsymbol{u}_2}\frac{\rho_{\mathcal{V}}(h(\boldsymbol{u}_1),h(\boldsymbol{u}_2))}{\rho_{\mathcal{U}}(\boldsymbol{u}_1,\boldsymbol{u}_2)}.\end{aligned}$$

$\operatorname{lip}(h;\mathcal{V}\leftarrow\mathcal{U})$ is written as $\operatorname{lip}(h\leftarrow\boldsymbol{u})$ if $\mathcal{V}$ and $\mathcal{U}$ are clear from the context. The function h is called a *L-Lipschitz function* if its Lipschitz constant is L.

The definition states that Lipschitz functions are a family of functions, where a 'small' change in $\boldsymbol{u}$ will not cause a very large change in $h(\boldsymbol{u})$.

In classification tasks, with the classifier $h(\boldsymbol{x};\boldsymbol{\theta})$, the following two kinds of Lipschitz constant are defined which differ in the input space:

(1) Treat $\mathcal{X}$ as the input space and consider $\operatorname{lip}(h\leftarrow\boldsymbol{x})$.

A metric ρ is defined as

$$\rho_{\mathcal{H}(\boldsymbol{x})}(h(\boldsymbol{x}_1;\cdot),h(\boldsymbol{x}_2;\cdot)) = \max_{\boldsymbol{\theta}\in\Theta}|h(\boldsymbol{x}_1;\boldsymbol{\theta})-h(\boldsymbol{x}_2;\boldsymbol{\theta})|,$$

and $\operatorname{lip}(h\leftarrow\boldsymbol{x})$ is

$$\begin{aligned}\operatorname{lip}(h\leftarrow\boldsymbol{x}) =& \max_{\boldsymbol{x}_1,\boldsymbol{x}_2\in\mathcal{X},\boldsymbol{x}_1\neq\boldsymbol{x}_2}\frac{\rho_{\mathcal{H}(\boldsymbol{x})}\Big(h(\boldsymbol{x}_1;\cdot),h(\boldsymbol{x}_2;\cdot)\Big)}{\rho_{\mathcal{X}}(\boldsymbol{x}_1,\boldsymbol{x}_2)}\\ =& \max_{\boldsymbol{x}_1,\boldsymbol{x}_2\in\mathcal{X},\boldsymbol{x}_1\neq\boldsymbol{x}_2,\boldsymbol{\theta}\in\Theta}\frac{|h(\boldsymbol{x}_1;\boldsymbol{\theta})-h(\boldsymbol{x}_2;\boldsymbol{\theta})|}{\|\boldsymbol{x}_1-\boldsymbol{x}_2\|}.\end{aligned}$$

(2) Treat Θ as the input space and consider $\operatorname{lip}(h\leftarrow\boldsymbol{\theta})$.

A metric $\rho_{\mathcal{H}(\boldsymbol{\theta})}$ in the parametric function space is defined as

$$\rho_{\mathcal{H}(\boldsymbol{\theta})}(h(\cdot;\boldsymbol{\theta}_1),h(\cdot;\boldsymbol{\theta}_2)) = \max_{\boldsymbol{x}\in\mathcal{X}}|h(\boldsymbol{x};\boldsymbol{\theta}_1)-h(\boldsymbol{x};\boldsymbol{\theta}_2)|.$$

and $\mathrm{lip}(h \leftarrow \boldsymbol{\theta})$ is

$$\begin{aligned}\mathrm{lip}(h \leftarrow \boldsymbol{\theta}) &= \max_{\boldsymbol{\theta}_1,\boldsymbol{\theta}_2\in\Theta,\boldsymbol{\theta}_1\neq\boldsymbol{\theta}_2} \frac{\rho_{\mathcal{H}(\boldsymbol{\theta})}\Big(h(\cdot;\cdot\boldsymbol{\theta}_1), h(\cdot;\cdot,\boldsymbol{\theta}_2)\Big)}{\rho_\Theta(\boldsymbol{\theta}_1,\boldsymbol{\theta}_2)} \\ &= \max_{\boldsymbol{\theta}_1,\boldsymbol{\theta}_2\in\Theta,\boldsymbol{\theta}_1\neq\boldsymbol{\theta}_2,\boldsymbol{x}\in\mathcal{X}} \frac{|h(\boldsymbol{x};\boldsymbol{\theta}_1) - h(\boldsymbol{x};\boldsymbol{\theta}_2)|}{\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|},\end{aligned}$$

where $\|\cdot\|$ is restricted to the L2-norm for vectors or the Frobenius norm for matrices in this thesis.

1.4.2 Bounding the Lipschitz Constant

The Lipschitz constant could be obtained based on the gradient of differentiable functions.

Theorem 2. [28] Let $\mathcal{U} \in \mathbb{R}^n$ be open, $h : \mathcal{U} \rightarrow \mathbb{R}$ be differentiable and the line segment $[\boldsymbol{u}_1, \boldsymbol{u}_2] \in \mathcal{U}$, where $[\boldsymbol{u}_1, \boldsymbol{u}_2] = \{\boldsymbol{v} \mid \boldsymbol{v} = \boldsymbol{u}_1 + t(\boldsymbol{u}_2 - \boldsymbol{u}_1), t \in [0, 1]\}$ joins $\boldsymbol{u}_1$ to $\boldsymbol{u}_2$. Based on the *Mean Value Theorem*, there exists a $\boldsymbol{u} \in [\boldsymbol{u}_1, \boldsymbol{u}_2]$

$$f(\boldsymbol{u}_2) - f(\boldsymbol{u}_1) = f'(\boldsymbol{u})^T(\boldsymbol{u}_2 - \boldsymbol{u}_1).$$

Applying the mean value theorem, we could bound the Lipschitz constant via the gradient.

Corollary 1. Let $\mathcal{U} \in \mathbb{R}^n$ be open and convex, function $h : \mathcal{U} \rightarrow \mathbb{R}$ be differentiable inside $\mathcal{U}$, then the following inequality holds:

$$\mathrm{lip}(h \leftarrow \boldsymbol{u}) = \max_{\boldsymbol{u}_1,\boldsymbol{u}_2\in U,\boldsymbol{u}_1\neq\boldsymbol{u}_2} \frac{|h(\boldsymbol{u}_2) - h(\boldsymbol{u}_1)|}{\|\boldsymbol{u}_2 - \boldsymbol{u}_1\|} \leq \max_{u\in\mathcal{U}} \|h'(\boldsymbol{u})\|.$$

Proof. Since $\mathcal{U}$ is convex, $\forall \boldsymbol{u}_1, \boldsymbol{u}_2 \in U, \boldsymbol{u}_1 \neq \boldsymbol{u}_2$, the line segment $[\boldsymbol{u}_1, \boldsymbol{u}_2] = \{\boldsymbol{v} \mid \boldsymbol{v} = \boldsymbol{u}_1 + t(\boldsymbol{u}_2 - \boldsymbol{u}_1), t \in [0, 1]\} \in \mathcal{U}$.

$$|h(\boldsymbol{u}_2)-h(\boldsymbol{u}_1)| =_{(a)} |h'(\boldsymbol{u})^T(\boldsymbol{u}_2-\boldsymbol{u}_1)| \leq_{(b)} \|h'(\boldsymbol{u})\|\|\boldsymbol{u}_2-\boldsymbol{u}_1\| \leq_{(c)} \max_{u\in\mathcal{U}} \|h'(\boldsymbol{u})\|\|\boldsymbol{u}_2-\boldsymbol{u}_1\|,$$

where equality (a) is due to Theorem 2; inequality (b) is due to the Cauchy-Schwarz inequality; inequality (c) is due to $\|h'(\boldsymbol{u})\| \leq \max_{u\in\mathcal{U}} \|h'(\boldsymbol{u})\|$. □

Lipschitz constant can also be obtained based on the basic ones using the following lemma.

Lemma 2. [44, 74] Let $h_1, h_2 \in \mathrm{lip}(h \leftarrow \boldsymbol{u})$. Then
(a) $\mathrm{lip}(ah_1 \leftarrow \boldsymbol{u}) \leq |a| \, \mathrm{lip}(h_1 \leftarrow \boldsymbol{u})$, where a is a constant;
(b) $\mathrm{lip}(h_1 + h_2 \leftarrow \boldsymbol{u}) \leq \mathrm{lip}(h_1 \leftarrow \boldsymbol{u}) + \mathrm{lip}(h_2 \leftarrow \boldsymbol{u})$,
$\mathrm{lip}(h_1 - h_2 \leftarrow \boldsymbol{u}) \leq \mathrm{lip}(h_1 \leftarrow \boldsymbol{u}) + \mathrm{lip}(h_2 \leftarrow \boldsymbol{u})$;
(c) $\mathrm{lip}(\min(h_1, h_2) \leftarrow \boldsymbol{u}) \leq \max\{\mathrm{lip}(h_1 \leftarrow \boldsymbol{u}), \mathrm{lip}(h_2 \leftarrow \boldsymbol{u})\}$,
$\mathrm{lip}(\max(h_1, h_2) \leftarrow \boldsymbol{u}) \leq \max\{\mathrm{lip}(h_1 \leftarrow \boldsymbol{u}), \mathrm{lip}(h_2 \leftarrow \boldsymbol{u})\}$,
where $\max(h_1, h_2)$ or $\min(h_1, h_2)$ denotes the pointwise maximum or minimum of functions h_1 and h_2;
(d) $\mathrm{lip}(h_2 \circ h_1 \leftarrow \boldsymbol{u}) \leq \mathrm{lip}(h_2 \leftarrow h_1) \, \mathrm{lip}(h_1 \leftarrow \boldsymbol{u})$, where $\circ$ denotes the composition of functions.

This lemma illustrates that after the operations of multiplication by constant, addition, subtraction, minimisation, maximisation and function composition, the functions are still Lipschitz continuous.

Lemma 3. [44, 74] Let $\mathrm{lip}(h_1 \leftarrow \boldsymbol{u})$ and $\mathrm{lip}(h_2 \leftarrow \boldsymbol{u})$ be finite and h_1, h_2 are bounded real-valued functions. Then the product of two functions[3] $h_1 h_2$ is again Lipschitz continuous and

$$\mathrm{lip}(h_1 h_2 \leftarrow \boldsymbol{u}) \leq \|h_1\|_\infty \, \mathrm{lip}(h_2 \leftarrow \boldsymbol{u}) + \|h_2\|_\infty \, \mathrm{lip}(h_1 \leftarrow \boldsymbol{u}),$$

where $\|h\|_\infty = \max_{\boldsymbol{u}} h(\boldsymbol{u})$.

This lemma illustrates that after the operation of function multiplication, the result is a Lipschitz function if the basic Lipschitz functions are bounded.

1.5 Structure of the Thesis

The structure of the thesis is illustrated in Figure 1.1. The thesis is organised as follows. All classifiers used in the thesis are restricted to have certain kinds

[3] The product of two functions is defined as $(h_1 h_2)(\boldsymbol{u}) = h_1(\boldsymbol{u}) h_2(\boldsymbol{u})$.

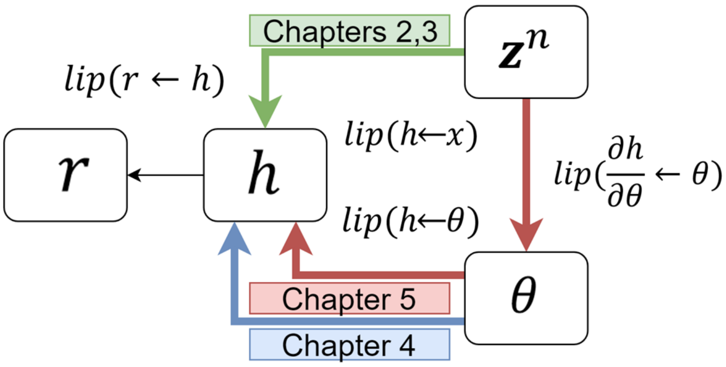

Figure 1.1: The structure of the thesis. h denotes a distance-based classifier; $\boldsymbol{z}^n$ denotes the set of training instances; $\boldsymbol{\theta}$ denotes the parameters of the classifier; r denotes a risk function. The relationship between these concepts can be linked with certain kinds of Lipschitz properties. In the thesis, the loss function with bounded $\mathrm{lip}(r \leftarrow h)$ is selected. In Chapters 2 and 3, the classifiers have bounded $\mathrm{lip}(h \leftarrow \boldsymbol{x})$; In Chapter 4, the classifier has bounded $\mathrm{lip}(h \leftarrow \boldsymbol{\theta})$; In Chapter 5, the classifier has bounded $\mathrm{lip}(h \leftarrow \boldsymbol{\theta})$ and $\mathrm{lip}(\frac{\partial h}{\partial \boldsymbol{\theta}} \leftarrow \boldsymbol{\theta})$.

of Lipschitz continuous properties. In Chapters 2 and 3, functions with bounded $\mathrm{lip}(h \leftarrow \boldsymbol{x})$ are used for metric learning with large margin ratio and local metrics, respectively. In Chapter 4, functions with bounded $\mathrm{lip}(h \leftarrow \boldsymbol{\theta})$ are used for metric learning with instance extraction. In Chapter 5, functions with bounded $\mathrm{lip}(h \leftarrow \boldsymbol{\theta})$ and $\mathrm{lip}(\frac{\partial h}{\partial w} \leftarrow \boldsymbol{\theta})$ are used and the resultant PAC bound takes into account some terms appeared in the optimisation algorithm. Generally speaking, the restrictions on the functions used become increasingly stronger from chapter to chapter. The main contributions of this thesis are covered in Chapters 2–5 led to the following four submissions or working papers:

- Mingzhi Dong, Xiaochen Yang, Yang Wu, Jing-Hao Xue. Metric Learning via Maximizing the Lipschitz Margin Ratio. Working paper (based on Chapter 2).

- Mingzhi Dong, Yujiang Wang, Xiaochen Yang and Jing-Hao Xue. Learning Local Metrics and Influential Regions for Classification. IEEE Transactions on Pattern Analysis and Machine Intelligence, under revision (based on Chapter 3).

- Mingzhi Dong, Rui Zhu, Yujiang Wang and Jing-Hao Xue. Metric Learning with Instance Extraction. Working paper (based on Chapter 4).

- Mingzhi Dong, Xiaochen Yang, Yujiang Wang and Jing-Hao Xue. Smooth Metric Learning with Instance Extraction. Working paper (based on Chapter 5).

1.6 Appendix

1.6.1 Some Useful Properties of Probability and Set

Lemma 4. Let $\boldsymbol{u}^n = \{\boldsymbol{u}_i, i = 1, \dots, n\}$ denote the set of n independent random variables and $\boldsymbol{u}_i$ is bounded by the interval $[a_i, b_i]$. Let $\overline{\boldsymbol{u}} = \frac{\sum_{i=1}^n \boldsymbol{u}_i}{n}$, then *Hoeffding's inequality* indicates

$$\mathbb{P}_{\boldsymbol{u}^n}(\overline{\boldsymbol{u}} - \mathbb{E}_{\boldsymbol{u}^n}[\overline{\boldsymbol{u}}] \geq \epsilon) \leq \exp\Big(-\frac{2n^2\epsilon^2}{\sum_{i=1}^n (b_i - a_i)^2}\Big);$$

$$\mathbb{P}_{\boldsymbol{u}^n}(|\overline{\boldsymbol{u}} - \mathbb{E}_{\boldsymbol{u}^n}[\overline{\boldsymbol{u}}]| \geq \epsilon) \leq 2\exp\Big(-\frac{2n^2\epsilon^2}{\sum_{i=1}^n (b_i - a_i)^2}\Big).$$

Proposition 1. *Probability of the Union of Sets*:

$$\mathbb{P}(A \cup B) \leq \mathbb{P}(A) + \mathbb{P}(B).$$

1.6.2 Equivalence between Optimisation Problems (1.1) and (1.2)

Let us review the two optimisation problems.

Problem 1:

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & R_n(\boldsymbol{z}_n, h_{\boldsymbol{\theta}}) \\ s.t. \quad & \mathcal{P}(\boldsymbol{\theta}) \leq C; \end{aligned}$$

Problem 2:

$$\min_{\boldsymbol{\theta}} \quad R_n(\boldsymbol{z}_n, h_{\boldsymbol{\theta}}) + \lambda\mathcal{P}(\boldsymbol{\theta}).$$

The Lagrange function of Problem 1 is

$$\mathcal{L}(\theta, u) = R_n(\boldsymbol{z}_n, h_{\boldsymbol{\theta}}) + u(\mathcal{P}(\boldsymbol{\theta}) - C), \quad u \geq 0,$$

where u is the Lagrangian multiplier.

For Problem 1, the (KKT) necessary conditions imply

$$\begin{aligned} &\text{Condition 1} \quad \frac{\partial R_n(\boldsymbol{z}_n, h_{\boldsymbol{\theta}})}{\partial \boldsymbol{\theta}} + u\frac{\partial \mathcal{P}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = 0; \\ &\text{Condition 2} \quad u(\mathcal{P}(\boldsymbol{\theta}) - C) = 0. \end{aligned}$$

For Problem 2, the necessary condition implies

$$\frac{\partial R_n(\boldsymbol{z}_n, h_{\boldsymbol{\theta}})}{\partial \boldsymbol{\theta}} + \lambda\frac{\partial \mathcal{P}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = 0.$$

Suppose θ_1^* and μ^* satisfy the necessary condition of Problem 1. Setting $\lambda = \mu^*$, we can see θ_1^* satisfies for the necessary condition of Problem 2. Suppose θ_2^* satisfies the necessary condition of Problem 2. Setting $\mu = \lambda$ and $C = \mathcal{P}(\boldsymbol{\theta}_2^*)$, we can see Condition 1 and Condition 2 of Problem 1 are satisfied, so θ_2^* satisfies the necessary condition of Problem 1 as well. Based on the above results, the necessary conditions of Problem 1 and Problem 2 are equivalent.

Meanwhile, when the regularisation term in Problem 2 is a convex function, the equivalent Problem 1 constrains $\boldsymbol{\theta}$ inside the set of $\{\boldsymbol{\theta}|\mathcal{P}(\boldsymbol{\theta}) \leq C\}$, which is a convex set [7].

Chapter 2

Metric Learning with Margin Ratio

2.1 Introduction

Classification is a fundamental research question in the field of machine learning. For distance-based classifiers, it is crucial to appropriately measure the distance between instances. The nearest neighbour classifier, a classical distance-based classifier, classifies a new instance into the class of the training instance with the shortest distance.

In practice, it is often difficult to handcraft a well-suited and adaptive distance metric. To mitigate this issue, metric learning has been proposed to enable learning a metric automatically from the data available. Metric learning with a convex objective function was first proposed in the pioneering work of [78]. The large margin intuition was introduced into the research of metric learning by the seminal researches of "large margin metric learning" (LMML) [56] and "large margin nearest neighbor" (LMNN) [76]. Besides the large margin approach, other inspiring metric learning strategies have been developed, such as nonlinear metrics [34, 24], localised strategies [12, 71, 49] and scalable/efficient algorithms [58, 53]. Metric learning has also been adopted by many other learning tasks, such as semi-supervised learning [82], unsupervised-learning [32], multi-task/cross-domain learning [41, 72], AUC optimisation [30] and distributed approaches [37].

On top of the methodological and applied advancement of metric learning, some theoretical progress has also been made recently, in particular on deriving

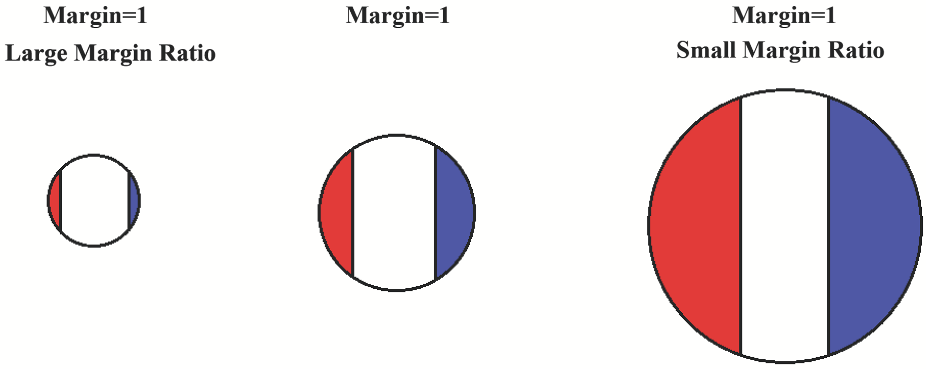

Figure 2.1: An illustration of the margin ratio. Each ball indicates a metric space. The red area indicates the area of positive class instances; the blue area indicates the area of negative class instances. Although the margins between the two classes in different metric spaces are the same, it is clear that the difficulties of classification are distinct between them.

different types of generalisation bounds for metric learning [33, 20, 67, 8]. These developments have theoretically justified the performance of metric learning algorithms. However, they generally lack a geometrical link with the classification margin, not as interpretable as one may expect (e.g. like the clear relationship between the margin and $1/|w|$ in the support vector machine (SVM)).

Besides the inter-class margin, the intra-class dispersion is also crucial to classification [13, 11, 31]. The intra-class dispersion is especially important for metric learning, because different metrics may lead to similar inter-class margins and quite different intra-class dispersion. As illustrated in Figure 2.1, although the margins in these metric spaces are exactly the same, the classification becomes more difficult as the margin ratio decreases. Therefore, the seminal work of [78] and many later work made efforts to consider the inter-class margin and the intra-class dispersion at the same time.

In this chapter, we aim to propose a new concept, the Lipschitz margin ratio, to integrate both inter-class and intra-class properties, and through maximising the Lipschitz margin ratio we aim to propose a new metric learning framework to enable the enhancement of the generalisation ability of a classifier. These two novelties are our main contributions to be made in this work.

To achieve these two aims and present our contributions in a well-structured way, we organise the rest of this chapter as follows. Firstly, in Section 2.2 we discuss the relationship between the distance-based classification, metric learning and Lipschitz functions. We show that a Lipschitz extension, which is a distance-based function, can be regarded as a generalised nearest neighbour model, enjoying great representation ability. Then, in Section 2.3 we introduce the Lipschitz margin ratio, and we point out that its associated learning bound indicates the desirability of maximising the Lipschitz margin ratio, for enhancing the generalisation ability of Lipschitz extensions. Consequently in Section 2.4, we propose a new metric learning framework through maximising the Lipschitz margin ratio. Moreover, we prove that many well-known metric learning algorithms are special cases of the proposed framework. Then for illustrative purposes, we implement the framework to learn the squared Mahalanobis metric. The method is presented in Section 2.4.3, and the experiment results are reported in Section 2.5, which demonstrate the superiority of the proposed method. Finally, we draw conclusions and discuss future work in Section 2.6. For the convenience of readers, some proofs are deferred to the Appendix 2.7.

2.2 Lipschitz Functions and Distance-based Classifiers

2.2.1 Definition of Lipschitz Functions

To start with, we will review the definitions of Lipschitz functions, the Lipschitz constant and the Lipschitz set.

Definition 8. [74] Let $(\mathcal{X}, \rho_{\mathcal{X}})$ be a metric space. A function $f : \mathcal{X} \to \mathbb{R}$ is called *Lipschitz continuous* (with respect to input $\boldsymbol{x}$) [1] if $\exists C < \infty, \forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{X}$,

$$|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)| \leq C\rho_{\mathcal{X}}(\boldsymbol{x}_1, \boldsymbol{x}_2).$$

[1] Without further clarification, the Lipschitz constant considered in this chapter is restricted to be with respect to the input $\boldsymbol{x}$.

The *Lipschitz constant* $\mathrm{lip}(f \leftarrow \boldsymbol{x})$ of a Lipschitz function f is

$$\begin{aligned} &\mathrm{lip}(f \leftarrow \boldsymbol{x}) \\ =&\min\{C \in \mathbb{R} | \forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{X}, |f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)| \leq C\rho_{\mathcal{X}}(\boldsymbol{x}_1, \boldsymbol{x}_2)\} \\ =&\max_{\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{X} : \boldsymbol{x}_1 \neq \boldsymbol{x}_2} \frac{|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)|}{\rho_{\mathcal{X}}(\boldsymbol{x}_1, \boldsymbol{x}_2)}, \end{aligned}$$

and function f is also called a *L-Lipschitz function* if its Lipschitz constant is bounded by L. Meanwhile, all L-Lipschitz functions construct the *L-Lipschitz set*

$$L\text{-}Lip(\mathcal{X}) = \{f : \mathcal{X} \rightarrow \mathbb{R}; \mathrm{lip}(f \leftarrow \boldsymbol{x}) \leq L\}.$$

From the definitions, we can observe that the Lipschitz constant is fundamentally connected with the metric $\rho_{\mathcal{X}}$; and that the Lipschitz functions have specified a family of functions whose change of output values can be bounded by the distances in the input space.

2.2.2 Lipschitz Extensions and Distance-based Classifiers

Distance-based classifiers are the classifiers that are based on certain kinds of distance metrics. Most of distance-based classifiers stem from the nearest neighbours (NN) classifier. To decide the class label of a new instance, the NN classifier compares the distances between the new instance and the training instances.

In binary classification tasks, a Lipschitz function is commonly used as the classification function f and the instance $\boldsymbol{x}$ is then classified according to the sign of $f(\boldsymbol{x})$. Using Theorem 3, we shall present a family of Lipschitz functions, called Lipschitz extensions. We shall also show that Lipschitz extensions present a distance-based classifier, and that a special case of Lipschitz extensions returns exactly the same classification result as the NN classifier.

Theorem 3. [43, 77, 74, 42] (McShane-Whitney Extension Theorem) Given a function u defined on a finite subset $A = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, there exist a family of functions which coincide with u on $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, are defined on the whole space $\mathcal{X}$, and have the same Lipschitz constant as u. Additionally, it is possible to explicitly construct functions $U_\alpha(\boldsymbol{x})$ with Lipschitz constant u in the following form and they are called *L-Lipschitz extensions* of u:

$$U_\alpha(\boldsymbol{x}) = \alpha U_1(\boldsymbol{x}) + (1-\alpha)U_2(\boldsymbol{x}),$$

where $\alpha \in [0,1]$,

$$\begin{aligned} U_1(\boldsymbol{x}) &= \overline{u}(\boldsymbol{x}) = \min_{\boldsymbol{a}\in A}\{u(\boldsymbol{a}) + L\rho(\boldsymbol{x},\boldsymbol{a})\}, \\ U_2(\boldsymbol{x}) &= \underline{u}(\boldsymbol{x}) = \max_{\boldsymbol{a}\in A}\{u(\boldsymbol{a}) - L\rho(\boldsymbol{x},\boldsymbol{a})\}. \end{aligned}$$

Theorem 3 can be readily validated by calculating the values of $U_1(\boldsymbol{x})$ and $U_2(\boldsymbol{x})$ on the finite points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$. The bound of the Lipschitz constant of $\overline{u}(\boldsymbol{x})$ and $\underline{u}(\boldsymbol{x})$ can be proved on the basis of the Lemmas in Appendix.

Theorem 3 clearly shows that Lipschitz extensions are based on certain kind of distance function ρ and hence are distance-based functions. Moreover, we can illustrate the relationship between Lipschitz extension functions and empirical risk as follows.

Assume A is the set of training instances of a classification task $A = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$. If there are no $\boldsymbol{x}_i, \boldsymbol{x}_j$ such that $\rho(\boldsymbol{x}_i, \boldsymbol{x}_j) = 0$ while their labels $t_i \neq t_j$ (i.e. no overlap between training instances from different classes), setting $u(\boldsymbol{x}_i) = t_i$ would result in zero empirical risk, and $u(\boldsymbol{x}_i)$ would be a Lipschitz function with Lipschitz constant L_0,

$$L_0 = \max_{i,j} \frac{|t_i - t_j|}{\rho(\boldsymbol{x}_i, \boldsymbol{x}_j)},$$

where the existence of such a function u, i.e. the Lipschitz extensions, is guaranteed by Theorem 3.

That is, when doing classification, if we set L of Lipschitz extension to be larger than L_0, zero empirical risk could be obtained. In other words, as distance-based functions, Lipschitz extensions enjoy excellent representation ability for classification tasks.

Moreover, if we set α as $1/2$, Lipschitz extensions will have exactly the same classification results as the NN classifier:

Proposition 2. [42] The function $U_{1/2}(\boldsymbol{x})$ defined above has the same sign, i.e. has the same classification results, as that of the NN classifier.

2.3 Lipschitz Margin Ratio

In the previous section, we show that Lipschitz extensions can be viewed as a distance-based classifier, and its representation ability is so strong that zero empirical error can be obtained under mild conditions. In this section, we shall propose the Lipschitz margin ratio to control the model complexity of the Lipschitz functions and hence improve its generalisation ability. To start with, we propose an intuitive way to understand the Lipschitz margin and the Lipschitz margin ratio. Then, learning bounds of the Lipschitz margin ratio will be presented.

2.3.1 Lipschitz Margin

We define the training set of class k as $\boldsymbol{S}_k = \{\boldsymbol{x}_i | t_i = k, \boldsymbol{x}_i \in \boldsymbol{S}\}$, where $k \in \{1, -1\}$; the decision boundary of classification function f as $\boldsymbol{H}_f = \{\boldsymbol{h} | \boldsymbol{h} \in \mathcal{X}, f(\boldsymbol{h}) = 0\}$. The margin used in [42] is equivalent to the Lipschitz margin defined below.

Definition 9. The *Lipschitz margin* is the distance between the training sets $\boldsymbol{S}_1$ and $\boldsymbol{S}_{-1}$:

$$\text{L-Margin} = D(\boldsymbol{S}_1, \boldsymbol{S}_{-1}) = \min_{\boldsymbol{x}_i \in \boldsymbol{S}_{-1}, \boldsymbol{x}_j \in \boldsymbol{S}_1} \rho(\boldsymbol{x}_i, \boldsymbol{x}_j). \tag{2.1}$$

The relationship between the Lipschitz margin and the Lipschitz constant is established as follows.

Proposition 3. For any L-Lipschitz function f satisfying $\forall \boldsymbol{x}_i \in \boldsymbol{S}_1, f(\boldsymbol{x}_i) \geq 1$ and $\forall \boldsymbol{x}_j \in \boldsymbol{S}_{-1}, f(\boldsymbol{x}_j) \leq -1$,

$$\text{L-Margin} \geq \frac{2}{L}. \tag{2.2}$$

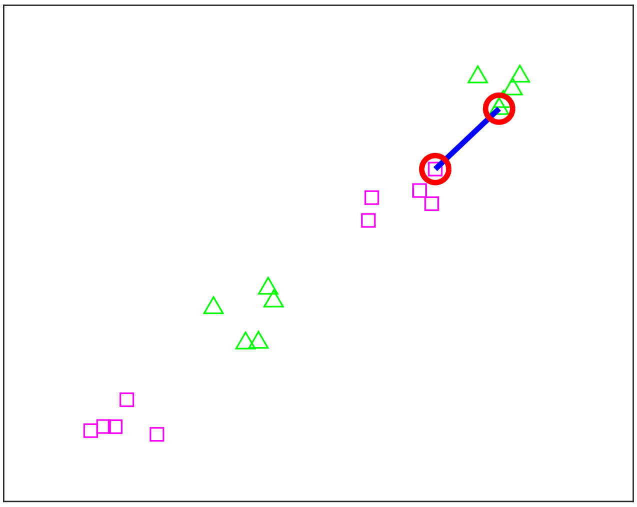

Figure 2.2: An illustration of the Lipschitz margin. Green triangles are instances from the positive class, and purple squares are from the negative class. Data points with red circles around them are the nearest instances from different classes. The length of the blue line indicates the value of the Lipschitz margin.

Proof. Let $\boldsymbol{x}_n$ and $\boldsymbol{x}_m$ denote the nearest instances from different classes, i.e.

$$\rho(\boldsymbol{x}_n, \boldsymbol{x}_m) = D(\boldsymbol{S}_1, \boldsymbol{S}_{-1}) = \min_{\boldsymbol{x}_i \in \boldsymbol{S}_{-1}, \boldsymbol{x}_j \in \boldsymbol{S}_1} \rho(\boldsymbol{x}_i, \boldsymbol{x}_j).$$

It is straightforward to see

$$\begin{aligned} \frac{2}{L} &\leq \frac{2}{|f(\boldsymbol{x}_n) - f(\boldsymbol{x}_m)|/\rho(\boldsymbol{x}_n, \boldsymbol{x}_m)} \\ &\leq \rho(\boldsymbol{x}_n, \boldsymbol{x}_m) \\ &= D(\boldsymbol{S}_1, \boldsymbol{S}_{-1}), \end{aligned}$$

where the first inequality follows from the definition of the Lipschitz constant; and the second inequality is for the reason that $\forall \boldsymbol{x}_i \in \boldsymbol{S}_1, f(\boldsymbol{x}_i) \geq 1$ and $\forall \boldsymbol{x}_j \in \boldsymbol{S}_{-1}, f(\boldsymbol{x}_j) \leq -1$, then $|f(\boldsymbol{x}_i) - f(\boldsymbol{x}_j)| \geq 2$. □

The proposition shows that the Lipschitz margin can be lower bounded by the multiplicative inverse Lipschitz constant.

The Lipschitz margin is closely related to the margin adopted in SVM (the distance between the hyperplane $\boldsymbol{H}_f$ and the training instances $\boldsymbol{S}$),

$$D(\boldsymbol{S}, \boldsymbol{H}_f) = \min_{\boldsymbol{x}_i \in \boldsymbol{S}, \boldsymbol{h} \in \boldsymbol{H}_f} \rho(\boldsymbol{x}_i, \boldsymbol{h}),$$

As illustrated in Figure 2.2, the Lipschitz margin is also suitable for the classification of non-linearly separable classes. The relationship between these two types of margins are described via the following proposition.

Proposition 4. In the Euclidean space, let f be any continuous function which correctly classifies all the training instances, i.e. $\forall \boldsymbol{x}_i \in \boldsymbol{S}, t_i f(\boldsymbol{x}_i) \geq 1$, then

$$D(\boldsymbol{S}_1, \boldsymbol{S}_{-1}) \geq 2D(\boldsymbol{S}, \boldsymbol{H}_f).$$

Proof. In the Euclidean space,

$$D(\boldsymbol{S}_1, \boldsymbol{S}_{-1}) = \min_{\boldsymbol{x}_i \in \boldsymbol{S}_{-1}, \boldsymbol{x}_j \in \boldsymbol{S}_{+1}} \rho_E(\boldsymbol{x}_i, \boldsymbol{x}_j),$$

$$D(\boldsymbol{S}, \boldsymbol{H}_f) = \min_{\boldsymbol{x}_i \in \boldsymbol{S}, \boldsymbol{h} \in \boldsymbol{H}_f} \rho_E(\boldsymbol{x}_i, \boldsymbol{h}),$$

and $\rho_E(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T(\boldsymbol{x}_i - \boldsymbol{x}_j)}$ is the Euclidean distance.

Let $\boldsymbol{x}_n$ and $\boldsymbol{x}_m$ denote the nearest instances from different classes, i.e.

$$\rho_E(\boldsymbol{x}_n, \boldsymbol{x}_m) = D(\boldsymbol{S}_1, \boldsymbol{S}_{-1}) = \min_{\boldsymbol{x}_i \in \boldsymbol{S}_{-1}, \boldsymbol{x}_j \in \boldsymbol{S}_{+1}} \rho_E(\boldsymbol{x}_i, \boldsymbol{x}_j),$$

where $\boldsymbol{x}_n \in \boldsymbol{S}_{-1}, \boldsymbol{x}_m \in \boldsymbol{S}_{+1}$.

We define a connected set $\boldsymbol{Z} = \{a\boldsymbol{x}_n + (1-a)\boldsymbol{x}_m | 0 \leq a \leq 1\}$, which indicates the line segment between $\boldsymbol{x}_n$ and $\boldsymbol{x}_m$. Because $f(\boldsymbol{x}_n) \leq -1$, $f(\boldsymbol{x}_m) \geq 1$ and for any continuous function f, it maps connected sets into connected sets, there exists $\boldsymbol{z} \in \boldsymbol{Z}$, such that $f(\boldsymbol{z}) = 0$. According to the definition of $\boldsymbol{H}_f$, we can see $\boldsymbol{z} \in \boldsymbol{H}_f$.

Therefore,

$$\begin{aligned} D(\boldsymbol{S}, \boldsymbol{H}_f) &= \min_{\boldsymbol{x}_i \in \boldsymbol{S}, \boldsymbol{h} \in \boldsymbol{H}_f} \rho_E(\boldsymbol{x}_i, \boldsymbol{h}) \\ &\leq \min_{\boldsymbol{x}_i \in \boldsymbol{S}} \rho_E(\boldsymbol{x}_i, \boldsymbol{z}) \\ &\leq \frac{\rho_E(\boldsymbol{x}_n, \boldsymbol{z}) + \rho_E(\boldsymbol{x}_m, \boldsymbol{z})}{2} \\ &= \frac{\rho_E(\boldsymbol{x}_n, \boldsymbol{x}_m)}{2} \\ &= \frac{D(\boldsymbol{S}_1, \boldsymbol{S}_{-1})}{2}, \end{aligned}$$

where the second equality follows from the connectedness property of $\boldsymbol{Z}$. □

2.3.2 Lipschitz Margin Ratio

The Lipschitz margin discussed above effectively depicts the inter-class relationship. However, as we mentioned before, when we learn the metrics, different metrics will result in different intra-class dispersion and it is also important to consider intra-class properties. Hence we propose the Lipschitz margin ratio to incorporate both the inter-class and intra-class properties into metric learning.

We start with defining the diameter of a metric space:

Definition 10. [74] The *diameter* of a metric space $(\mathcal{X}, \rho)$ is defined as

$$\mathrm{diam}(\mathcal{X}, \rho) = \sup_{\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X}} \rho(\boldsymbol{x}_i, \boldsymbol{x}_j).$$

The Lipschitz margin ratio is then defined as the ratio between the margin and $\mathrm{diam}(\mathcal{X})$ (i.e. the diameter) or $\mathrm{diam}(\boldsymbol{S}_1) + \mathrm{diam}(\boldsymbol{S}_{-1})$ (i.e. the sum of intra-class dispersion), as follows.

Definition 11. The *Diameter Lipschitz Margin Ratio* (L-Ratio^{Diam}) and the *Intra-Class Dispersion Lipschitz Margin Ratio* (L-Ratio^{Intra}) in a metric space $(\mathcal{X}, \rho)$ are defined as

$$\begin{aligned}\text{L-Ratio}^{Diam} &= \frac{D(\boldsymbol{S}_1, \boldsymbol{S}_{-1})}{\operatorname{diam}(\mathcal{X}, \rho)} \\ &= \frac{\min\limits_{\boldsymbol{x}_i \in \boldsymbol{S}_{-1}, \boldsymbol{x}_j \in \boldsymbol{S}_1} \rho(\boldsymbol{x}_i, \boldsymbol{x}_j)}{\max\limits_{\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X}} \rho(\boldsymbol{x}_i, \boldsymbol{x}_j)},\end{aligned}$$

$$\begin{aligned}\text{L-Ratio}^{Intra} &= \frac{D(\boldsymbol{S}_1, \boldsymbol{S}_{-1})}{\operatorname{diam}(\boldsymbol{S}_1, \rho) + \operatorname{diam}(\boldsymbol{S}_{-1}, \rho)} \\ &= \frac{\min\limits_{\boldsymbol{x}_i \in \boldsymbol{S}_{-1}, \boldsymbol{x}_j \in \boldsymbol{S}_1} \rho(\boldsymbol{x}_i, \boldsymbol{x}_j)}{\max\limits_{\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{S}_1} \rho(\boldsymbol{x}_i, \boldsymbol{x}_j) + \max\limits_{\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{S}_{-1}} \rho(\boldsymbol{x}_i, \boldsymbol{x}_j)}.\end{aligned}$$

The relationship between L-Ratio^{Diam} and L-Ratio^{Intra} can be established via the following proposition.

Proposition 5. In a metric space $(\mathcal{X}, \rho)$,

$$\operatorname{diam}(\mathcal{X}, \rho) \leq \operatorname{diam}(\boldsymbol{S}_1, \rho) + \operatorname{diam}(\boldsymbol{S}_{-1}, \rho) + D(\boldsymbol{S}_{-1}, \boldsymbol{S}_1)$$

and

$$\frac{1}{\text{L-Ratio}^{Diam}} \leq \frac{1}{\text{L-Ratio}^{Intra}} + 1.$$

Proof. : See Appendix 2.7.1 □

In this inequality, $\operatorname{diam}(\boldsymbol{S}_1, \rho)$ and $\operatorname{diam}(\boldsymbol{S}_{-1}, \rho)$ indicate the maximum intra-class distances, and $D(\boldsymbol{S}_1, \boldsymbol{S}_{-1})$ indicates the inter-class margin. Therefore, this inverse margin ratio regularisation will push the learner to select a metric ρ which pulls the instances from the same class closer (small $\sum_{t=1,-1} \operatorname{diam}(\boldsymbol{S}_t, \rho)$) and enlarges the margin between the instances from different classes (large $D(\boldsymbol{S}_1, \boldsymbol{S}_{-1})$). In a very simple (linearly separable one-dimensional) case, as illustrated in Figure 2.3, $\operatorname{diam}(\mathcal{X}, \rho)$ can be decomposed into intra-class dispersion ($\operatorname{diam}(\boldsymbol{S}_{-1}, \rho)$, $\operatorname{diam}(\boldsymbol{S}_{-1}, \rho)$) and inter-class margin ($D(\boldsymbol{S}_1, \boldsymbol{S}_{-1})$) directly.

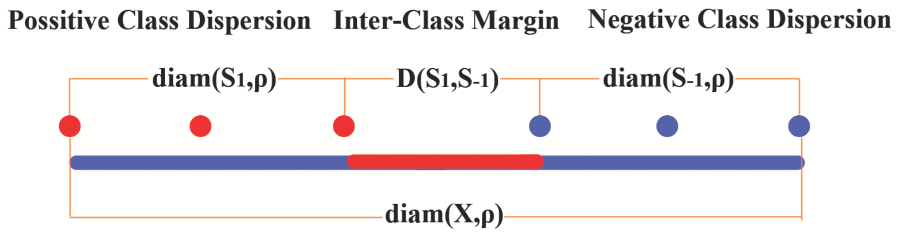

Figure 2.3: An illustration of the relationship between the margin ratio and the intra-/inter-class properties. A linearly separable one-dimensional case is used as an example. The red solid circles indicate the positive class instances; the blue solid circles indicate the negative class instances.

Then we can bound the Lipschitz margin ratio using the Lipschitz constant and the diameter of metric space:

Proposition 6. For any L-Lipschitz function f satisfying $\forall \boldsymbol{x}_i \in \boldsymbol{S}_1, f(\boldsymbol{x}_i) \geq 1$ and $\forall \boldsymbol{x}_j \in \boldsymbol{S}_{-1}, f(\boldsymbol{x}_j) \leq -1$,

$$\text{L-Ratio}^{Diam} \geq \frac{2}{L\,\text{diam}(\mathcal{X}, \rho)},$$

$$\text{L-Ratio}^{Intra} \geq \frac{2}{L\,\text{diam}(\boldsymbol{S}_1, \rho) + L\,\text{diam}(\boldsymbol{S}_{-1}, \rho)}.$$

Proof. The inequalities can be obtained by substituting the result of Proposition 3. □

Based on this proposition, although it is not possible to calculate the exact value of the Lipschitz margin ratio in most cases, we can use $\frac{1}{L\,\text{diam}(\mathcal{X},\rho)}$ or $\frac{1}{L\,\text{diam}(\boldsymbol{S}_1,\rho)+L\,\text{diam}(\boldsymbol{S}_{-1},\rho)}$ as a surrogate. For example, in the objective function of metric learning by maximising Lipschitz margin ratio, we can maximise $\frac{1}{L\,\text{diam}(\mathcal{X},\rho)}$ or $\frac{1}{L\,\text{diam}(\boldsymbol{S}_1,\rho)+L\,\text{diam}(\boldsymbol{S}_{-1},\rho)}$ or equivalently minimise $L\,\text{diam}(\mathcal{X}, \rho)$ or $L(\text{diam}(\boldsymbol{S}_1, \rho) + \text{diam}(\boldsymbol{S}_{-1}, \rho))$.

Furthermore, in some cases we may be more interested in the local properties rather than the global ones (see also Section 4.2). In those cases we can define the *local* Lipschitz margin ratio as follows.

Definition 12. The *local Lipschitz margin ratio* with subset $\boldsymbol{S}^l \subseteq \boldsymbol{S}$ and metric $\rho^l \in \mathcal{D}$ is defined as

$$\text{Local-Ratio}^{Diam} = \frac{\text{L-Margin}}{\operatorname{diam}(\boldsymbol{S}^l, \rho^l)} = \frac{D(\boldsymbol{S}_1^l, \boldsymbol{S}_{-1}^l)}{\operatorname{diam}(\boldsymbol{S}^l, \rho^l)},$$

$$\begin{aligned}\text{Local-Ratio}^{Intra} &= \frac{\text{L-Margin}}{\operatorname{diam}(\boldsymbol{S}_1, \rho) + \operatorname{diam}(\boldsymbol{S}_{-1}, \rho)} \\ &= \frac{D(\boldsymbol{S}_1^l, \boldsymbol{S}_{-1}^l)}{\operatorname{diam}(\boldsymbol{S}_1^l, \rho^l) + \operatorname{diam}(\boldsymbol{S}_{-1}^l, \rho^l)},\end{aligned}$$

where $\boldsymbol{S}_k^l = \{\boldsymbol{x}_i | t_i = k, \boldsymbol{x}_i \in \boldsymbol{S}^l\}$ indicates the local training set of class k and $k \in \{1, -1\}$.

2.3.3 Learning Bounds of the Lipschitz Margin Ratio

In the section above, we have defined the Lipschitz margin ratio, which is a measure of model complexity. In this section, we shall establish the effectiveness of the Lipschitz margin ratio through showing the relationship between its lower bound and the generalisation ability.

Definition 13. [18] For a metric space $(\mathcal{X}, \rho)$, let λ be the smallest number such that every ball in $\mathcal{X}$ can be covered by λ balls of half the radius. Then λ is called the *doubling constant* of $\mathcal{X}$ and the *doubling dimension* of $\mathcal{X}$ is $\operatorname{ddim}(\mathcal{X}) = \log_2 \lambda$.

As presented in [18], a low Euclidean dimension implies a low doubling dimension (Euclidean metrics of dimension d have doubling dimension $O(d)$); a low doubling dimension is more general than a low Euclidean dimension and can be utilised to measure the 'dimension' of a general metric space.

Definition 14. We say that $\mathcal{F}$ *γ-shatters* $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, if there exists $s_1, \ldots, s_n$, such that, for every $\epsilon \in \{\pm 1\}^n$, there exists $f \in \mathcal{F}$ such that $\forall t \in \{1, \ldots, n\}$

$$\epsilon_t (f_\epsilon(\boldsymbol{x}_t) - s_t) \geq \gamma$$

Fat-shattering dimension is defined as follows

$$\begin{aligned} fat_{\gamma}(\mathcal{F}) = & \max\{n; \exists \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathcal{X}, \\ & s.t.\ \mathcal{F}\ \gamma \text{ -shatters } \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}. \end{aligned}$$

Theorem 4. [18] Let $\mathcal{F}$ be the collection of real valued functions over $\mathcal{X}$ with the Lipschitz constant at most L. Define $D = \mathrm{fat}_{1/16}(\mathcal{F})$ and let P be some probability distribution on $\mathcal{X} \times \{-1, 1\}$. Suppose that $(x_i, t_i), i = 1, \ldots, n$ are drawn from $\mathcal{X} \times \{-1, 1\}$ independently according to P. Then for every $f \in \mathcal{F}$ that classifies a sample of size n correctly, we have with probability at least $1 - \delta$

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

Furthermore, if f is correct on all but k examples, we have with probability at least $1 - \delta$

$$\begin{aligned} & P\{(\boldsymbol{x}, t) : \mathrm{sign}[f(\boldsymbol{x})] \neq t\} \\ & \leq \frac{k}{n} + \sqrt{\frac{2}{n}(D \log_2(34en/D) \log_2(578n) + \log_2(4/\delta))}. \end{aligned} \tag{2.3}$$

Proposition 7. In classification problems, $\forall \boldsymbol{x}_i \in \boldsymbol{S}_1$ and $\forall \boldsymbol{x}_j \in \boldsymbol{S}_{-1}$, $L = \max_{i,j} \frac{2}{\rho(\boldsymbol{x}_i, \boldsymbol{x}_j)}$, where i and j indicate the index of positive and negative class instances respectively. Then $D = \mathrm{fat}_{1/16}(\mathcal{F})$ can be bounded by the surrogate of Lipschitz Margin Ratio as follows:

$$\begin{aligned} D & \leq \Big(16L\, \mathrm{diam}(\mathcal{X}, \rho)\Big)^{\mathrm{ddim}(\mathcal{X})} \\ & \leq \Big(16L(\mathrm{diam}(\boldsymbol{S}_1, \rho) + \mathrm{diam}(\boldsymbol{S}_{-1}, \rho)) + 32\Big)^{\mathrm{ddim}(\mathcal{X})}. \end{aligned} \tag{2.4}$$

Proof. The first inequality has been proved in [18]. We prove the second inequality here. Because $L = \max_{i,j} \frac{2}{\rho(\boldsymbol{x}_i, \boldsymbol{x}_j)} = \frac{2}{D(\boldsymbol{S}_{-1}, \boldsymbol{S}_1)}$, we have

$$LD(\boldsymbol{S}_{-1}, \boldsymbol{S}_1) = 2.$$

It follows that

$$\begin{aligned} L\,\mathrm{diam}(\mathcal{X},\rho) &\leq L(\mathrm{diam}(\boldsymbol{S}_1,\rho)+\mathrm{diam}(\boldsymbol{S}_{-1},\rho)+D(\boldsymbol{S}_{-1},\boldsymbol{S}_1)) \\ &= L((\mathrm{diam}(\boldsymbol{S}_1,\rho)+\mathrm{diam}(\boldsymbol{S}_{-1},\rho))+2, \end{aligned}$$

where the first inequality is based on Proposition 5. Meanwhile, because $\mathrm{ddim}(\mathcal{X}) \geq 1$, the second inequality holds. □

Corollary 2. Under the condition that $n \geq \frac{D}{34e}$, the following bounds for the surrogate margin ratios holds. If f is correct on all but k examples, we have with probability at least $1-\delta$

$$\begin{aligned} &P\{(\boldsymbol{x},t) : \mathrm{sign}[f(\boldsymbol{x})] \neq t\} \leq \frac{k}{n} + \\ &\sqrt{\frac{2}{n}((16C)^{\mathrm{ddim}(\mathcal{X})}\log_2(34en/(16C)^{\mathrm{ddim}(\mathcal{X})})\log_2(578n)+\log_2(4/\delta))}, \end{aligned} \tag{2.5}$$

where $C = L\,\mathrm{diam}(\mathcal{X},\rho)$ or $C = L(\mathrm{diam}(\boldsymbol{S}_1,\rho)+\mathrm{diam}(\boldsymbol{S}_{-1},\rho))+2$.

Proof. Substitute the inequalities of Proposition 7 into Theorem 4. □

The above learning bound illustrates the relationship between the generalisation error (i.e. the difference between the expected error $P\{(\boldsymbol{x},t) : \mathrm{sign}[f(\boldsymbol{x})] \neq t\}$ and the empirical error $\frac{k}{n}$) and the surrogate inverse Lipschitz margin ratio $L\,\mathrm{diam}(\mathcal{X},\rho)$ or $L(\mathrm{diam}(\boldsymbol{S}_1,\rho)+\mathrm{diam}(\boldsymbol{S}_{-1},\rho))$. Therefore, reducing the value of surrogate inverse Lipschitz margin ratio would help reduce the gap between the empirical error and the expected error, which implies an improvement in the generalisation ability of the model. In other words, the learning bound indicates that minimising inverse Lipschitz margin ratio would be an effective way to enhance the generalisation ability and control model complexity.

2.4 Metric Learning via Maximising the Lipschitz Margin Ratio

From previous sections, we have seen that Lipschitz functions have the following desirable properties relevant to metric learning:

- (Close relationship with metrics) The definitions of the Lipschitz constant, Lipschitz functions and Lipschitz extensions have natural relationship with metrics.

- (Strong representation ability) Lipschitz functions, in particular Lipschitz extensions, could obtain small empirical risks, and hence illustrate the representational capability of Lipschitz functions.

- (Good generalisation ability) Complexity of Lipschitz functions could be controlled by penalising the Lipschitz margin ratio.

Therefore, it is reasonable for us to conduct metric learning with the Lipschitz functions and control the model complexity by maximising (the lower bound of) the Lipschitz margin ratio.

2.4.1 Learning Framework

Similarly to other structure risk minimisation approaches, we minimise the empirical risk and maximise (the lower bound of) the Lipschitz margin ratio in the proposed framework. To estimate (the lower bound of) the Lipschitz margin ratio, we may either

- use training instances to estimate the Lipschitz constant $\mathrm{lip}(f \leftarrow \boldsymbol{x})$ and the diameters $\mathrm{diam}(\mathcal{X}, \rho)$, and obtain $\hat{L}$ and $\hat{\mathrm{diam}}$; or

- adopt the upper bounds of $\mathrm{lip}(f \leftarrow \boldsymbol{x})$ and $\mathrm{diam}(\mathcal{X}, \rho)$ by applying the properties of the classifier f and metric space $(\mathcal{X}, \rho)$, and obtain L^s and diam^s.

The optimisation problem could be formulated as follows:

$$\begin{aligned} \min_{\boldsymbol{\xi}, \boldsymbol{a}, \rho} \quad & 1/\text{L-Ratio} + \alpha \textstyle\sum_{i=1}^{N} \xi_i \\ s.t. \quad & t_i f(\boldsymbol{x}_i; \boldsymbol{a}, \rho) \geq 1 - \xi_i \\ & \xi_i \geq 0 \\ & i = 1, \ldots, N, \end{aligned} \tag{2.6}$$

where N indicates the number of training instances; $\boldsymbol{a}$ denotes the parameters of the classification function f; $\boldsymbol{\xi} = \{\xi_i\}$ is the hinge loss; $\alpha > 0$ is a trade-off parameter which balances the empirical risk term $\sum_{i=1}^{N}\xi_i$ and the generalisation ability term $1/$L-Ratio. $\mathrm{lip}(f \leftarrow \boldsymbol{x})$ and $\mathrm{diam}(\mathcal{X},\rho)$, $\mathrm{diam}(\boldsymbol{S}_1,\rho)$ and $\mathrm{diam}(\boldsymbol{S}_{-1},\rho)$ from the L-Ratio term, will be replaced by either the empirically estimated values $\hat{L}$ and $\widehat{\mathrm{diam}}$ or the theoretical upper bounds L^s and diam^s.

Empirical estimates of $\hat{L}$ and $\widehat{\mathrm{diam}}$ can be added as constraints

$$\frac{f(\boldsymbol{x}_i;\boldsymbol{a},\rho)-f(\boldsymbol{x}_j;\boldsymbol{a},\rho)}{\rho(\boldsymbol{x}_i,\boldsymbol{x}_j)} \le \hat{L},$$

$$\rho(\boldsymbol{x}_i,\boldsymbol{x}_j) \le \widehat{\mathrm{diam}}(\mathcal{X},\rho), \quad \text{where } x_i \in \boldsymbol{S}, x_j \in \boldsymbol{S},$$
$$\rho(\boldsymbol{x}_i,\boldsymbol{x}_j) \le \widehat{\mathrm{diam}}(\boldsymbol{S}_1,\rho), \quad \text{where } x_i \in \boldsymbol{S}_1, x_j \in \boldsymbol{S}_1,$$
$$\rho(\boldsymbol{x}_i,\boldsymbol{x}_j) \le \widehat{\mathrm{diam}}(\boldsymbol{S}_{-1},\rho), \quad \text{where } x_i \in \boldsymbol{S}_{-1}, x_j \in \boldsymbol{S}_{-1}.$$

Then the objective function of minimising $1/$L-Ratio^{Diam} becomes

$$\min_{\boldsymbol{\xi},\boldsymbol{a},\rho,\hat{L},\widehat{\mathrm{diam}}} \quad \hat{L}\widehat{\mathrm{diam}}(\mathcal{X},\rho)+\alpha\textstyle\sum_{i=1}^{N}\xi_i, \tag{2.7}$$

where the penalty term $\hat{L}\widehat{\mathrm{diam}}(\mathcal{X},\rho)$ tries to maximise the inter-class margin (via minimising $\hat{L}$) and minimise the overall diameter (via minimising $\widehat{\mathrm{diam}}(\mathcal{X},\rho)$).

The objective function to minimise $1/$L-Ratio^{Intra} becomes

$$\min_{\boldsymbol{\xi},\boldsymbol{a},\rho,\hat{L},\widehat{\mathrm{diam}}} \hat{L}(\widehat{\mathrm{diam}}(\boldsymbol{S}_1,\rho)+\widehat{\mathrm{diam}}(\boldsymbol{S}_{-1},\rho))+\alpha\sum_{i=1}^{N}\xi_i,$$

or we can minimise an upper bound of $1/$L-Ratio^{Intra} as

$$\min_{\boldsymbol{\xi},\boldsymbol{a},\rho,\hat{L},\widehat{\mathrm{diam}}} 2\hat{L}\max(\widehat{\mathrm{diam}}(\boldsymbol{S}_1,\rho),\widehat{\mathrm{diam}}(\boldsymbol{S}_{-1},\rho))+\alpha\sum_{i=1}^{N}\xi_i, \tag{2.8}$$

where the penalty term $L(\widehat{\mathrm{diam}}(\boldsymbol{S}_1,\rho)+\mathrm{diam}(\boldsymbol{S}_{-1},\rho))$ or $\hat{L}\max(\widehat{\mathrm{diam}}(\boldsymbol{S}_1,\rho),\widehat{\mathrm{diam}}(\boldsymbol{S}_{-1},\rho))$ tries to maximise the inter-class margin (via minimising $\hat{L}$) and minimise the intra-class dispersion (via minimising $\widehat{\mathrm{diam}}(\boldsymbol{S}_1,\rho)+\widehat{\mathrm{diam}}(\boldsymbol{S}_{-1},\rho)$ or $\max(\widehat{\mathrm{diam}}(\boldsymbol{S}_1,\rho),\widehat{\mathrm{diam}}(\boldsymbol{S}_{-1},\rho))$) at the same time.

2.4.2 Relationship with other Metric Learning Methods

Some widely adopted metric learning algorithms can be shown as special cases of the proposed framework.

As presented in Appendix 2.7.3, based on our framework, the regularisation term of Large Margin Metric Learning (LMML) [56] could be interpreted as an upper bound of $1/\text{L-Ratio}^{Diam}$ margin ratio; and this framework could suggest a reasonable strategy for choosing the target neighbours and the imposter neighbours in LMML. Also as discussed in Appendix 2.7.4, we can see that the regularisation term of LMNN [76] could be interpreted as an upper bound of $1/\text{Local-Ratio}^{Intra}$.

2.4.3 Applying the Framework for Learning the Squared Mahalanobis Metric

We now apply the proposed framework to learn the squared Mahalanobis metric,

$$\rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{M} (\boldsymbol{x}_i - \boldsymbol{x}_j), \boldsymbol{M} \in \boldsymbol{M}_+,$$

where $\boldsymbol{M}_+$ denotes the set of positive semi-definite matrices. A Lipschitz extension function is selected as the classifier:

$$\begin{aligned} f(\boldsymbol{x}; \boldsymbol{a}, \rho) =& U_{1/2}(\boldsymbol{x}) \\ =& \frac{1}{2} \min_{i=1,\dots,N} (a_i + L\rho_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}_i)) + \\ & \frac{1}{2} \max_{i=1,\dots,N} (a_i - L\rho_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}_i)). \end{aligned} \tag{2.9}$$

In binary classification tasks, let $t_i \in \{-1, +1\}$ indicate the label of x_i, $i = 1, \dots, N$.

Based on the framework of (2.6) and (2.7), firstly we propose an optimisation formula which penalises the L-Ratio^{Diam}:

$$\begin{array}{cc}\min\limits_{\boldsymbol{a},\boldsymbol{\xi},\boldsymbol{M},\hat{\mathrm{diam}},\hat{L}} & \hat{L}\hat{\mathrm{diam}} + \alpha \sum_{i=1}^{N} \xi_i \\ s.t. & \frac{|a_i - a_j|}{\rho_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j)} \leq \hat{L} \\ & \rho_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j) \leq \hat{\mathrm{diam}} \\ & t_i a_i = 1 - \xi_i \\ & \xi_i \geq 0, \boldsymbol{M} \in \boldsymbol{M}_+ \\ & \boldsymbol{x}_i \in \boldsymbol{S}, \boldsymbol{x}_j \in \boldsymbol{S}. \end{array} \tag{2.10}$$

At first glance, the optimisation problem seems quite complex. However, based on the smoothness assumption, balanced class assumption ($|\boldsymbol{S}_1| = |\boldsymbol{S}_2|$) and some equivalent transformations, as illustrated in Appendix 2.7.5, the following optimisation problem can be obtained:

$$\begin{array}{cc}\min\limits_{\boldsymbol{\xi},\boldsymbol{M'},d} & cd + \sum \xi_{ij} \\ s.t. & \rho_{\boldsymbol{M'}}(\boldsymbol{x}_i,\boldsymbol{x}_j) \geq 2 - \xi_{ij} \\ & \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ are instance pairs with different labels} \\ & \rho_{\boldsymbol{M'}}(\boldsymbol{x}_m,\boldsymbol{x}_n) \leq d \\ & \xi_{ij} \geq 0, \boldsymbol{M'} \in \boldsymbol{M}_+ \\ & \boldsymbol{x}_m, \boldsymbol{x}_n \in \boldsymbol{S}. \end{array} \tag{2.11}$$

Intuitively speaking, the first set of inequality constraints indicate that the distances between samples from different classes should be large; and the third set of inequality constraints indicate that the estimated diameter should be small.

Based on the framework in (2.6) and (2.8), we can also propose an optimisation formula which penalises the upper bound of $1/\text{L-Ratio}^{Intra}$:

$$\begin{aligned}
\min_{\boldsymbol{a},\boldsymbol{\xi},\boldsymbol{M},\hat{\text{diam}},\hat{L}} \quad & \hat{L}\hat{\text{diam}} + \alpha \textstyle\sum_{i=1}^{N} \xi_i \\
s.t. \quad & \tfrac{|a_i - a_j|}{\rho_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j)} \leq \hat{L} \\
& \rho_{\boldsymbol{M}}(\boldsymbol{x}_m,\boldsymbol{x}_n) \leq \hat{\text{diam}} \\
& \boldsymbol{x}_m \text{ and } \boldsymbol{x}_n \text{ are instance pairs with the same label} \\
& t_i a_i = 1 - \xi_i \\
& \xi_i \geq 0, \boldsymbol{M} \in \boldsymbol{M}_+ \\
& \boldsymbol{x}_i, \boldsymbol{x}_j \in \boldsymbol{S}.
\end{aligned} \tag{2.12}$$

The only difference between (2.10) and (2.12) lies on the selected instance pairs to estimate $\hat{\text{diam}}$: (2.10) utilises all instance pairs to estimate the diameter of all the training instances, while (2.12) utilises the instances pairs with the same label to estimate the maximum intra-class dispersion. Similarly to the transformations from (2.10) to (2.11), the following optimisation problem can be obtained:

$$\begin{aligned}
\min_{\boldsymbol{\xi},\boldsymbol{M'},d} \quad & cd + \textstyle\sum \xi_{ij} \\
s.t. \quad & \rho_{\boldsymbol{M'}}(\boldsymbol{x}_i,\boldsymbol{x}_j) \geq 2 - \xi_i - \xi_j \\
& \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ are instance pairs with different labels} \\
& \rho_{\boldsymbol{M'}}(\boldsymbol{x}_m,\boldsymbol{x}_n) \leq d \\
& \boldsymbol{x}_m \text{ and } \boldsymbol{x}_n \text{ are instance pairs with the same label} \\
& \xi_i \geq 0, \boldsymbol{M'} \in \boldsymbol{M}_+.
\end{aligned} \tag{2.13}$$

In order to solve (2.11) and (2.13) more efficiently, alternating direction methods of multipliers (ADMM) have been adopted (see Algorithm 1), and the detailed derivation of the ADMM algorithm is presented in Appendix 2.7.6.

2.5 Experiments

To evaluate the performance of our proposed methods, we compare them with four widely adopted distance-based algorithms: nearest neighbor (NN), large margin nearest neighbor (LMNN) [76], maximally collapsing metric learning

Algorithm 1 ADMM for (2.11)

```
Input:
$\boldsymbol{A}_1, \boldsymbol{A}_2$
Initialise:
$\boldsymbol{M} = \boldsymbol{I}, \boldsymbol{m}_1 = \boldsymbol{m}_2 = \text{vector}(\boldsymbol{M}), \boldsymbol{p} = 2 - \boldsymbol{A}_1\boldsymbol{m}_1,$
$\boldsymbol{q} = 2 - \boldsymbol{A}_2\boldsymbol{m}_2, \boldsymbol{\alpha}_{1,2,3,4} = \boldsymbol{0}$
  while not converged do
    1. Update $\boldsymbol{p}_{ij}^{t+1}$ using (2.18)
    2. Update $\boldsymbol{q}_{ij}^{t+1}$ using bisection search for $t^*$ and Equation 2.19
    3. Update $\boldsymbol{m}_1^{t+1}$ using (2.20)
    4. Update $\boldsymbol{m}_2^{t+1}$ using (2.21)
    5. Update $\boldsymbol{m}^{t+1}$ using (2.22)
    6. Update the Lagrangian multipliers $\boldsymbol{\alpha}_1^{t+1}$, $\boldsymbol{\alpha}_2^{t+1}$, $\boldsymbol{\alpha}_3^{t+1}$, $\boldsymbol{\alpha}_4^{t+1}$ using (2.23)
  end while
Output: $M$
```

(MCML) [15] and neighborhood Components Analysis (NCA) [17]. Under our framework, we have implemented Lip^D (based on the diameter Lipschitz margin ratio), Lip^I (based on the intra-class Lipschitz margin ratio), Lip^D(P) (ADMM-based fast Lip^D), Lip^I(P) (ADMM-based fast Lip^I).

Our proposed Lip^D, Lip^I are implemented using the cvx toolbox[2] in MATLAB with the solver of SeDuMi [63]. The C in our algorithm is fixed at 1 and the λ in the ADMM algorithm is fixed at 1. The LMNN, MCML and NCA are from the dimension reduction toolbox[3].

In the experiments, we focus on the most representative task, binary classification. Eight publicly available data sets from the websites of UCI[4] and LibSVM[5] are adopted to evaluate the performance, namely Statlog/LibSVM Australian Credit Approval (Australian), UCI/LibSVM Original Breast Cancer Wisconsin (Cancer), UCI/LibSVM Pima Indians Diabetes (Diabetes), UCI Echocardiogram (Echo), UCI Fertility (Fertility), LibSVM Fourclass (Fourclass), UCI Haberman's Survival (Haberman) and UCI Congressional Voting Records (Voting). For each data set, 60% instances are randomly selected as training samples, the rest as

[2] http://cvxr.com/

[3] https://lvdmaaten.github.io/drtoolbox/

[4] https://archive.ics.uci.edu/ml/datasets.html

[5] https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/binary.html

Table 2.1: Experiment results of large margin ratio metric learning. Mean accuracy (percentage) and standard deviations are reported with the best ones in bold and underlined.

data sets	Lip^D	Lip^D(P)	Lip^I	Lip^I(P)
Australian	79.64 ± 2.27	80.04 ± 1.92	$\mathbf{80.90} \pm 1.74$	80.29 ± 2.15
Cancer	95.30 ± 1.12	94.84 ± 0.95	95.27 ± 1.01	94.84 ± 0.95
Diabetes	69.42 ± 2.03	69.38 ± 1.59	69.64 ± 2.62	68.80 ± 1.29
Echo	68.00 ± 5.49	$\mathbf{69.00} \pm 6.30$	68.67 ± 8.64	68.67 ± 5.92
Fertility	79.02 ± 4.48	81.46 ± 5.04	78.05 ± 6.60	80.98 ± 3.78
Fourclass	$\mathbf{99.91} \pm 0.14$	$\mathbf{99.91} \pm 0.14$	99.86 ± 0.15	99.88 ± 0.15
Haberman	$\mathbf{66.42} \pm 2.20$	66.26 ± 3.12	$\mathbf{66.42} \pm 2.27$	65.77 ± 2.83
Voting	93.37 ± 2.29	92.40 ± 1.90	$\mathbf{93.83} \pm 1.26$	92.40 ± 1.90
# of best	2	2	**3**	0

data sets	NN	LMNN	MCML	NCA
Australian	79.89 ± 1.31	79.96 ± 2.61	79.89 ± 2.30	79.89 ± 1.18
Cancer	95.61 ± 0.68	$\mathbf{95.41} \pm 0.66$	95.37 ± 1.14	94.95 ± 1.17
Diabetes	69.46 ± 1.22	69.90 ± 1.79	$\mathbf{70.03} \pm 1.34$	68.44 ± 2.69
Echo	65.36 ± 2.43	62.00 ± 10.56	66.33 ± 2.92	66.33 ± 4.97
Fertility	83.21 ± 2.79	$\mathbf{84.39} \pm 2.36$	83.17 ± 5.69	83.66 ± 2.31
Fourclass	99.87 ± 1.14	99.68 ± 0.42	99.88 ± 0.20	99.68 ± 0.62
Haberman	66.25 ± 1.74	66.26 ± 3.12	$\mathbf{66.42} \pm 2.24$	63.66 ± 3.93
Voting	92.85 ± 0.79	93.31 ± 0.72	92.40 ± 1.66	93.37 ± 1.50
# of best	0	2	2	0

test samples. This process is repeated 10 times and the mean accuracy is reported.

As shown in Table 2.1, the proposed algorithms Lip achieve the best mean accuracy on four data sets and equally best with MCML on one data set. The Lip outperforms 1-NN and NCA on seven data sets and LMNN and MCML on five data sets. The only dataset that the Lip performs worse than all other methods is Fertility, in which our method potentially suffers from within-class outliers and hence has a large intra-class dispersion. Apart from this data set, LMNN or MCML outperforms the Lip by only a small performance gap, less than 0.5%. Such encouraging results demonstrate the effectiveness of the proposed framework.

2.6 Conclusion

In this chapter, we have presented that the representation ability of Lipschitz functions is very strong and the complexity of the Lipschitz functions in a metric space can be controlled by penalising the Lipschitz margin ratio. Based on these desirable properties, we have proposed a new metric learning framework via maximising the Lipschitz margin ratio. An application of this framework for learning the squared Mahalanobis metric has been implemented and the experiment results are encouraging.

The diameter Lipschitz margin ratio or the intra-class Lipschitz margin ratio in the optimisation function is equivalent to an adaptive regularisation. In other words, since we encourage samples to stay close within the same class, samples which locate near the class boundary are valued more than those in the centre. Therefore, the performance of our method may deteriorate under the existence of outliers and this problem has been reported on the data set Fertility. We aim to develop more robust methods in our future work.

The local property within a data set could vary dramatically, and hence it is worthwhile to develop an algorithm based on local Lipschitz margin ratio. One option is to follow the idea of LMNN, learning a general metric but considering different local Lipschitz margin ratio; or we can learn a separate metric on each local area.

2.7 Appendix

2.7.1 Proof on Proposition 5

Proof. In any metric space $(\mathcal{X}, \rho)$, let $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ denote the training instances which satisfy

$$\rho(\boldsymbol{x}_a, \boldsymbol{x}_b) = \mathrm{diam}(\boldsymbol{S}, \rho) = \operatorname*{argmax}_{\boldsymbol{x}_a, \boldsymbol{x}_b \in \boldsymbol{S}} \rho(\boldsymbol{x}_a, \boldsymbol{x}_b).$$

(1) If $t_a = t_b$,

$$\begin{aligned}
\mathrm{diam}(\boldsymbol{S}, \rho) &= \rho(\boldsymbol{x}_a, \boldsymbol{x}_b) \\
&= \mathrm{diam}(\boldsymbol{S}_{t_a}, \rho) \\
&\leq \mathrm{diam}(\boldsymbol{S}_1, \rho) + \mathrm{diam}(\boldsymbol{S}_{-1}, \rho) + D(\boldsymbol{S}_{-1}, \boldsymbol{S}_1).
\end{aligned}$$

(2) If $t_a \neq t_b$, let $\boldsymbol{x}_n$ and $\boldsymbol{x}_m$ denote the nearest instances from different classes, i.e.

$$\rho(\boldsymbol{x}_n, \boldsymbol{x}_m) = D(\boldsymbol{S}_1, \boldsymbol{S}_{-1}) = \min_{\boldsymbol{x}_i \in \boldsymbol{S}_{-1}, \boldsymbol{x}_j \in \boldsymbol{S}_{+1}} \rho(\boldsymbol{x}_i, \boldsymbol{x}_j),$$

where $\boldsymbol{x}_n \in \boldsymbol{S}_{t_a}, \boldsymbol{x}_m \in \boldsymbol{S}_{t_b}$. We can see

$$\begin{aligned}
\operatorname{diam}(\mathcal{X}, \rho) &= \rho(\boldsymbol{x}_a, \boldsymbol{x}_b) \\
&\leq \rho(\boldsymbol{x}_a, \boldsymbol{x}_n) + \rho(\boldsymbol{x}_n, \boldsymbol{x}_m) + \rho(\boldsymbol{x}_m, \boldsymbol{x}_b) \\
&\leq \operatorname{diam}(\boldsymbol{S}_1, \rho) + D(\boldsymbol{S}_{-1}, \boldsymbol{S}_1) + \operatorname{diam}(\boldsymbol{S}_{-1}, \rho).
\end{aligned}$$

Take the definition of L-Ratio^{Diam} and L-Ratio^{Intra}:

$$\begin{aligned}
\frac{1}{\text{L-Ratio}^{Diam}} &= \frac{\operatorname{diam}(\mathcal{X}, \rho)}{D(\boldsymbol{S}_1, \boldsymbol{S}_{-1})} \\
&\leq \frac{\operatorname{diam}(\boldsymbol{S}_1, \rho) + D(\boldsymbol{S}_{-1}, \boldsymbol{S}_1) + \operatorname{diam}(\boldsymbol{S}_{-1}, \rho)}{D(\boldsymbol{S}_{-1}, \boldsymbol{S}_1)} \\
&= \frac{\operatorname{diam}(\boldsymbol{S}_1, \rho) + \operatorname{diam}(\boldsymbol{S}_{-1}, \rho)}{D(\boldsymbol{S}_{-1}, \boldsymbol{S}_1)} + 1 \\
&= \frac{1}{\text{L-Ratio}^{Intra}} + 1.
\end{aligned}$$

□

2.7.2 Properties of Lipschitz Functions

Lipschitz constant can also be obtained based on the basic ones using the following lemma.

Lemma 5. [44, 74] Let $h_1, h_2 \in \operatorname{lip}(h \leftarrow \boldsymbol{u})$. Then

(a) $\operatorname{lip}(h_1 + h_2 \leftarrow \boldsymbol{u}) \leq \operatorname{lip}(h_1 \leftarrow \boldsymbol{u}) + \operatorname{lip}(h_2 \leftarrow \boldsymbol{u})$,

$\operatorname{lip}(h_1 - h_2 \leftarrow \boldsymbol{u}) \leq \operatorname{lip}(h_1 \leftarrow \boldsymbol{u}) + \operatorname{lip}(h_2 \leftarrow \boldsymbol{u})$;

(b) $\operatorname{lip}(ah_1 \leftarrow \boldsymbol{u}) \leq |a| \operatorname{lip}(h_1 \leftarrow \boldsymbol{u})$, where a is a constant;

(c) $\operatorname{lip}(\min(h_1, h_2) \leftarrow \boldsymbol{u}) \leq \max\{\operatorname{lip}(h_1 \leftarrow \boldsymbol{u}), \operatorname{lip}(h_2 \leftarrow \boldsymbol{u})\}$,

$\operatorname{lip}(\max(h_1, h_2) \leftarrow \boldsymbol{u}) \leq \max\{\operatorname{lip}(h_1 \leftarrow \boldsymbol{u}), \operatorname{lip}(h_2 \leftarrow \boldsymbol{u})\}$,

where $\max(h_1, h_2)$ or $\min(h_1, h_2)$ denotes the pointwise maximum or minimum of functions h_1 and h_2;

(d) $\mathrm{lip}(h_2 \circ h_1 \leftarrow \boldsymbol{u}) \leq \mathrm{lip}(h_2 \leftarrow h_1)\,\mathrm{lip}(h_1 \leftarrow \boldsymbol{u})$, where $\circ$ denotes the composition of functions.

This lemma illustrates that after the operations of multiplication by constant, addition, subtraction, minimisation, maximisation and function composition, the functions are still Lipschitz continuous.

Lemma 6. [44, 74] Let $\mathrm{lip}(h_1 \leftarrow \boldsymbol{u})$ and $\mathrm{lip}(h_2 \leftarrow \boldsymbol{u})$ be finite and h_1, h_2 are bounded real-valued functions. Then the product $h_1 h_2$[6] is again Lipschitz continuous and

$$\mathrm{lip}(h_1 h_2 \leftarrow \boldsymbol{u}) \leq \|h_1\|_\infty \,\mathrm{lip}(h_2 \leftarrow \boldsymbol{u}) + \|h_2\|_\infty \,\mathrm{lip}(h_1 \leftarrow \boldsymbol{u}),$$

where $\|h\|_\infty = \max_{\boldsymbol{u}} h(\boldsymbol{u})$.

This lemma illustrates that after the operation of function multiplication, the result is a Lipschitz function if the basic Lipschitz functions are bounded.

2.7.3 Relationship between Lipschitz Margin Ratio and LMML [56]

The large margin metric learning (LMML) algorithm [56] has a close relationship with the proposed framework (2.6). Based on our proposed framework, the regularisation term of LMML could be interpreted as an upper bound of the inverse Lipschitz margin ratio. At the same time, the proposed framework could suggest a reasonable strategy for choosing the target neighbours and the imposter neighbours in LMML.

LMML uses the Mahalanobis metric $D_{\boldsymbol{M}}$, and the classification function of NN is equivalent to the following $f(\boldsymbol{x})$:

$$\begin{aligned} f(\boldsymbol{x}) &= D_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{S}_{-1}) - D_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{S}_1) \\ &= \min_a \{\rho_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}_a)\} - \min_b \{\rho_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}_b)\}, \end{aligned} \tag{2.14}$$

where $\boldsymbol{x}_a \in \boldsymbol{S}_{-1}$, $\boldsymbol{x}_b \in \boldsymbol{S}_1$.

[6] The product of two functions f, g is the componentwise multiplication: $(fg)(x) = f(x)g(x)$.

Then LMML adopts an upper bound of $1/\text{L-Ratio}^{Diam} \leq \operatorname{lip}(f \leftarrow \boldsymbol{x}) \operatorname{diam}(\mathcal{X}, D_{\boldsymbol{M}})$ as the regularisation term. Because $\operatorname{lip}(\rho_M(\boldsymbol{x}, \boldsymbol{x}_a) \leftarrow \boldsymbol{x}) = 1$, according to Lemma 5(c), $\operatorname{lip}(\min_a\{\rho_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}_a) \leftarrow \boldsymbol{x}\}) \leq 1$. Then according to Lemma 5(a), $\operatorname{lip}(f \leftarrow \boldsymbol{x})$ is bounded by 2 and

$$\begin{aligned}
&\operatorname{lip}(f \leftarrow \boldsymbol{x}) \max_{n,m} (\boldsymbol{x}_n - \boldsymbol{x}_m)^T \boldsymbol{M} (\boldsymbol{x}_n - \boldsymbol{x}_m) \\
&= \operatorname{lip}(f \leftarrow \boldsymbol{x}) \max_{n,m} \|(\boldsymbol{x}_n - \boldsymbol{x}_m)^T \boldsymbol{M} (\boldsymbol{x}_n - \boldsymbol{x}_m)\|_2 \\
&\leq \operatorname{lip}(f \leftarrow \boldsymbol{x}) \max_{n,m} \|\boldsymbol{x}_n - \boldsymbol{x}_m\|_2^2 \|\boldsymbol{M}\|_F \\
&\leq C \|\boldsymbol{M}\|_F,
\end{aligned}$$

where $C = 2 \max_{n,m} \|\boldsymbol{x}_n - \boldsymbol{x}_m\|_2^2$ and $\boldsymbol{x}_n, \boldsymbol{x}_m \in \mathcal{X}$. The first inequality holds because the matrix Frobenius norm is consistent with the vector L_2-norm. Therefore, the Frobenius norm or the squared Frobenius norm may be used as the regularisation term.

Based on the above discussion, in this special case, the proposed framework (2.6) could be represented as

$$\begin{aligned}
\min_{\boldsymbol{M}, \boldsymbol{\xi}} \quad & \|\boldsymbol{M}\|_F^2 + \alpha \textstyle\sum_{i=1}^N \xi_i^o \\
s.t. \quad & t_i f(\boldsymbol{x}_i; \boldsymbol{a}) \geq 1 - \xi_i^o \\
& \xi_i^o \geq 0, \boldsymbol{M} \in \boldsymbol{M}_+.
\end{aligned} \tag{2.15}$$

Then, the constraints of $\rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_k) - \rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 1 - \xi_i, j \rightarrow i, k \nrightarrow i$ in the optimisation problem of LMML serve as a heuristic approximation of $t_i f(\boldsymbol{x}_i; \boldsymbol{a}) \geq 1 - \xi_i$.

In fact, by choosing the target neighbour $\boldsymbol{x}_j$ of $\boldsymbol{x}_i$ as the nearest neighbour within the same class measured via the Euclidean metric and the imposter neighbours $\boldsymbol{x}_k$ as all the instances within the different class, i.e. $j = \operatorname{argmin}_u \rho_{\boldsymbol{M}=\boldsymbol{I}}(\boldsymbol{x}_i, \boldsymbol{x}_u)$ and $k \in \{u | \boldsymbol{x}_u \in \boldsymbol{S}_{-t_i}\}$, $\min_k\{\rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_k)\} - \rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$

would be an upper bound of $t_i f(\boldsymbol{x}_i)$. This is because

$$\begin{aligned} t_i f(\boldsymbol{x}_i) &= D_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{S}_{-t_i}) - D_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{S}_{t_i}) \\ &= \min_k \{\rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_k)\} - D_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{S}_{t_i}) \\ &\geq \min_k \{\rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_k)\} - \rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j), \end{aligned}$$

where the last inequality holds since $\boldsymbol{x}_j$ is $\boldsymbol{x}_i$'s nearest neighbour within the same class measured via the Euclidean metric and cannot be guaranteed to be the neighbour with in the same class with metric $\boldsymbol{M}$, but $-D_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{S}_{t_i}) \geq -\rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ always holds.

Let $t_i g(\boldsymbol{x}_i) = \min_k \{\rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_k)\} - \rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$, then the hinge loss of $t_i g(\boldsymbol{x}_i)$, i.e. $\max[1 - t_i g(\boldsymbol{x}_i), 0]$, is the upper bound of the hinge loss of $t_i f(\boldsymbol{x}_i)$, i.e. $\max[1 - t_i f(\boldsymbol{x}_i), 0]$, because

$$\begin{aligned} t_i g(\boldsymbol{x}_i) &= \min_k \{\rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_k)\} - \rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \leq t_i f(\boldsymbol{x}_i) \\ \Rightarrow & 1 - t_i g(\boldsymbol{x}_i) \geq 1 - t_i f(\boldsymbol{x}_i) \\ \Rightarrow & \max[1 - t_i g(\boldsymbol{x}_i), 0] \geq \max[1 - t_i f(\boldsymbol{x}_i), 0]. \end{aligned}$$

Therefore, the hinge loss ξ_i obtained by the following optimisation problem is the upper bound of ξ_i^o in (2.15):

$$\begin{aligned} \min_{\boldsymbol{M}, \boldsymbol{\xi}} \quad & \|\boldsymbol{M}\|_F^2 + \alpha \textstyle\sum_{i=1}^N \xi_i \\ s.t. \quad & t_i g(\boldsymbol{x}_i; \boldsymbol{a}) \geq 1 - \xi_i \\ & \xi_i \geq 0, \boldsymbol{M} \in \boldsymbol{M}_+. \end{aligned}$$

The above optimisation problem is equivalent to the following one:

$$\begin{aligned} \min_{\boldsymbol{M}, \boldsymbol{\xi}} \quad & \|\boldsymbol{M}\|_F^2 + \alpha \textstyle\sum_{i=1}^N \xi_i \\ s.t. \quad & \rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_k) - \rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 1 - \xi_i \\ & \xi_i \geq 0, \boldsymbol{M} \in \boldsymbol{M}_+, \end{aligned}$$

where $\boldsymbol{x}_j$ is $\boldsymbol{x}_i$'s nearest neighbour within the same class measured via the Euclidean metric and $\boldsymbol{x}_k$ are all the instances within the different class. This is a special case of the optimisation problem of LMML. Instead of using a heuristic approximation of the empirical risk, this setting of the target neighbour and the imposter neighbours could guarantee that ξ_i is the upper bound of ξ_i^o.

2.7.4 Relationship between Lipschitz Margin Ratio and LMNN [76]

The large margin nearest neighbor (LMNN) [76] also has a close relationship with the proposed framework. Similarly to that for LMML, the proposed framework could provide a reasonable strategy for choosing the target neighbours and the imposter neighbours in LMNN. In the following discussion, let $\boldsymbol{x}_j$ be $\boldsymbol{x}_i$'s nearest neighbour within the same class measured via the Euclidean metric and let $\{\boldsymbol{x}_k\}$ be the set of all instances within the different class of $\boldsymbol{x}_i$. We shall show that the regularisation term of LMNN could be interpreted as an upper bound of $1/\text{Local-Ratio}^{Intra}$ and ξ_i is also an upper bound of the empirical loss of $\boldsymbol{x}_i$.

LMNN uses the Mahalanobis metric $\rho_{\boldsymbol{M}}$, and the classification function is the same as that of LMML (2.14).

When the local margin of $\boldsymbol{x}_i$ with metric $\rho_{\boldsymbol{M}}$ is considered, the ideal subset $\boldsymbol{S}^l$ around $\boldsymbol{x}_i$ is $\{\boldsymbol{x}_i, \boldsymbol{x}_m, \boldsymbol{x}_n\}$, where $\boldsymbol{x}_m$ is x_i's nearest neighbour within the same class measured via the metric $\rho_{\boldsymbol{M}}$ and $\boldsymbol{x}_n$ is x_i's nearest neighbour within the different class measured via the metric $\rho_{\boldsymbol{M}}$. This subset is important for $\boldsymbol{x}_i$ because it determines the classification function of $\boldsymbol{x}_i$. Based on Definition 12, the local inverse Lipschitz margin ratio could be expressed as

$$\frac{\operatorname{diam}(\boldsymbol{S}^l, \rho_{\boldsymbol{M}})}{\text{L-Margin}},$$

and based on Proposition 5, it could be bounded as

$$\begin{aligned}\frac{1}{\text{Local-Ratio}^{Intra}} &= \frac{\operatorname{diam}(\boldsymbol{S}_1^l, \rho^l) + \operatorname{diam}(\boldsymbol{S}_{-1}^l, \rho^l)}{\text{L-Margin}} \\ &\le \frac{1}{2}\operatorname{lip}(f \leftarrow \boldsymbol{x})\{\operatorname{diam}(\boldsymbol{S}_{t_i}^l, \rho_{\boldsymbol{M}}) + \operatorname{diam}(\boldsymbol{S}_{-t_i}^l, \rho_{\boldsymbol{M}})\} \\ &= \frac{1}{2}\operatorname{lip}(f \leftarrow \boldsymbol{x})\rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_m),\end{aligned}$$

where the last equality holds because $\boldsymbol{S}^l = \{\boldsymbol{x}_i, \boldsymbol{x}_m, \boldsymbol{x}_n\}$, so $\boldsymbol{S}_{t_i}^l = \{\boldsymbol{x}_i, \boldsymbol{x}_m\}$, $\boldsymbol{S}_{-t_i}^l = \{\boldsymbol{x}_n\}$ and $\operatorname{diam}(\boldsymbol{S}_{t_i}^l, \rho_{\boldsymbol{M}}) = \rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_m)$, $\operatorname{diam}(\boldsymbol{S}_{-t_i}^l, \rho_{\boldsymbol{M}}) = 0$. Because $\operatorname{lip}(f \leftarrow \boldsymbol{x}) \le 2$, we can see

$$\frac{1}{\text{Local-Ratio}^{Intra}} \le \rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_m) \le \rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j),$$

where the second inequality holds because $\boldsymbol{x}_j$ is defined as $\boldsymbol{x}_i$'s nearest neighbour within the same class measured via the Euclidean metric and $\boldsymbol{x}_m$ may not be the same as $\boldsymbol{x}_j$, thus

$$\begin{aligned}\rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_m) &= D_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{S}_{t_i}) \\ &= \min_{\boldsymbol{x}_u \in \boldsymbol{S}_{t_i}} \rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_u) \\ &\le \rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j), \forall \boldsymbol{x}_j \in \boldsymbol{S}_{t_i}.\end{aligned}$$

Therefore, it is reasonable to penalise the sum of the upper bound of the local inverse Lipschitz margin ratios via

$$\sum_i \rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j).$$

Similarly to the discussion of LMML, the strategy of choosing target and imposter neighbours could guarantee that ξ_i is the upper bound of the empirical risk of $\boldsymbol{x}_i$.

The optimisation problem based on the proposed framework (2.6) could be rewritten as

$$\begin{aligned}\min_{\boldsymbol{M},\boldsymbol{\xi}} \quad & \textstyle\sum_i \rho_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j) + \alpha \sum_i \xi_i \\ s.t. \quad & \rho_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j) - \rho_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_k) \geq 1 - \xi_i \\ & \xi_i \geq 0, \boldsymbol{M} \in \boldsymbol{M}_+,\end{aligned} \tag{2.16}$$

where $\boldsymbol{x}_j$ is $\boldsymbol{x}_i$'s nearest neighbour within the same class measured via Euclidean metric and $\boldsymbol{x}_k$ are all the instances within the different class of $\boldsymbol{x}_i$. This is an optimisation problem of LMNN with a special strategy for choosing the target neighbour and imposter neighbour. This strategy could guarantee that ξ_i is the upper bound of the empirical risk.

2.7.5 From (2.10) to (2.11)

To start with, we assume that the intra class area is relatively smooth and $\hat{L}$ is always determined by instance pairs with different labels, then the optimisation problem (2.10) can be written as

$$\begin{aligned}\min_{\boldsymbol{a},\boldsymbol{\xi},\boldsymbol{M},\hat{\mathrm{diam}},\hat{L}} \quad & \hat{L}\hat{\mathrm{diam}} + \alpha \textstyle\sum_{n=1}^{N} \xi_i \\ s.t. \quad & \tfrac{|a_i - a_j|}{\rho_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j)} \leq \hat{L} \\ & \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ are instance pairs} \\ & \text{with different labels.} \\ & \rho_{\boldsymbol{M}}(\boldsymbol{x}_m,\boldsymbol{x}_n) \leq \hat{\mathrm{diam}} \\ & t_m a_m = 1 - \xi_m \\ & \xi_i \geq 0, \boldsymbol{M} \in \boldsymbol{M}_+ \\ & \boldsymbol{x}_m, \boldsymbol{x}_n \in \boldsymbol{S}.\end{aligned} \tag{2.17}$$

For the squared Mahalanobis metric, we have the following property:

$$\forall C,\ C\rho_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j) = \rho_{C\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j),$$

where C is any constant.

Based on this property, the optimisation problem (2.17) is equivalent to the following one:

$$\begin{aligned} \min_{\boldsymbol{a},\boldsymbol{\xi},\boldsymbol{M},\hat{L},\hat{\mathrm{diam}}} \quad & \hat{L}\hat{\mathrm{diam}} + \alpha \textstyle\sum_{n=1}^{N} \xi_i \\ s.t. \quad & |a_i - a_j| \leq \rho_{\hat{L}\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ & \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ are instance pairs with different labels} \\ & \rho_{\hat{L}\boldsymbol{M}}(\boldsymbol{x}_m, \boldsymbol{x}_n) \leq \hat{L}\hat{\mathrm{diam}} \\ & t_m a_m = 1 - \xi_m \\ & \xi_i \geq 0, \boldsymbol{M} \in \boldsymbol{M}_+ \\ & \boldsymbol{x}_m, \boldsymbol{x}_n \in \boldsymbol{S}. \end{aligned}$$

Take $t_m a_m = 1 - \xi_m$ into the first constraint, because x_i and x_j are from different classes, we have

$$|a_i - a_j| = |1 - \xi_i - (\xi_j - 1)| = |2 - \xi_i - \xi_j|.$$

Therefore, the objective function becomes

$$\begin{aligned} \min_{\boldsymbol{\xi},\boldsymbol{M},\hat{L},\hat{\mathrm{diam}}} \quad & \hat{L}\hat{\mathrm{diam}} + \alpha \textstyle\sum_{n=1}^{N} \xi_n \\ s.t. \quad & \rho_{\hat{L}\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq |2 - \xi_i - \xi_j| \\ & \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ are instance pairs with different labels} \\ & \rho_{\hat{L}\boldsymbol{M}}(\boldsymbol{x}_m, \boldsymbol{x}_n) \leq \hat{L}\hat{\mathrm{diam}} \\ & \xi_i \geq 0, \boldsymbol{M} \in \boldsymbol{M}_+ \\ & \boldsymbol{x}_m, \boldsymbol{x}_n \in \boldsymbol{S}, \end{aligned}$$

which is equivalent to the following optimisation problem:

$$\begin{aligned}
\min_{\boldsymbol{\xi},\boldsymbol{M},\hat{L},\hat{\mathrm{diam}}} \quad & \hat{L}\hat{\mathrm{diam}} + \alpha \textstyle\sum_{n=1}^{N} \xi_n \\
s.t. \quad & \rho_{\hat{L}\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j) \geq 2 - \xi_i - \xi_j \\
& \rho_{\hat{L}\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j) \geq \xi_i + \xi_j - 2 \\
& \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ are instance pairs with different labels} \\
& \rho_{\hat{L}\boldsymbol{M}}(\boldsymbol{x}_m,\boldsymbol{x}_n) \leq \hat{L}\hat{\mathrm{diam}} \\
& \xi_i \geq 0, \boldsymbol{M} \in \boldsymbol{M}_+ \\
& \boldsymbol{x}_m, \boldsymbol{x}_n \in \boldsymbol{S}.
\end{aligned}$$

To simplify the notation, we denote $\xi_{ij} = \xi_i + \xi_j$. With the assumption of balanced class, i.e. $|\boldsymbol{S}_1| = |\boldsymbol{S}_2| = \frac{N}{2}$, we have $\sum_{t_i \neq t_j} \xi_{ij} = N \sum_{n=1}^{N} \xi_n$. Let $d = \hat{L}\hat{\mathrm{diam}}$, $\boldsymbol{M'} = \hat{L}\boldsymbol{M}$, and $c = \frac{1}{\alpha N}$. This turns the optimisation problem into:

$$\begin{aligned}
\min_{\boldsymbol{\xi},\boldsymbol{M'},d} \quad & cd + \textstyle\sum_{i,j=1}^{N} \xi_{ij} \\
s.t. \quad & \rho_{\boldsymbol{M'}}(\boldsymbol{x}_i,\boldsymbol{x}_j) \geq 2 - \xi_{ij} \\
& \rho_{\boldsymbol{M'}}(\boldsymbol{x}_i,\boldsymbol{x}_j) \geq \xi_{ij} - 2 \\
& \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ are instance pairs with different labels} \\
& \rho_{\boldsymbol{M'}}(\boldsymbol{x}_m,\boldsymbol{x}_n) \leq d \\
& \xi_{ij} \geq 0, \boldsymbol{M'} \in \boldsymbol{M}_+ \\
& \boldsymbol{x}_m, \boldsymbol{x}_n \in \boldsymbol{S}.
\end{aligned}$$

The constraints with respect to ξ_{ij} are $(i)\xi_{ij} \geq 2 - \rho_{\boldsymbol{M'}}(\boldsymbol{x}_i,\boldsymbol{x}_j)$, $(ii)\xi_{ij} \leq 2 + \rho_{\boldsymbol{M'}}(\boldsymbol{x}_i,\boldsymbol{x}_j)$ and $(iii)\xi_{ij} \geq 0$. The objective function is to minimise ξ_{ij}, based on the objective function, constraints (iii), constraints (i) and the fact $\rho_{\boldsymbol{M'}}(\boldsymbol{x}_i,\boldsymbol{x}_j) \geq 0$, the maximal value of ξ_{ij} would be smaller or equal to 2. Thus constraints (ii) would always be satisfied. Thus constraints (ii) could be deleted and the optimisation problem could be formulated as (2.11).

2.7.6 ADMM Algorithm for (2.11) and (2.13)

The only difference between (2.11) and (2.13) lies on the selected instance pairs to estimate $\hat{\mathrm{diam}}$. For simplicity, only the derivation process of ADMM for (2.11) is illustrated here.

To start with, (2.11) is as follows

$$\begin{aligned} \min_{\boldsymbol{\xi},\boldsymbol{M'},d} \quad & cd + \textstyle\sum_{i,j=1}^{N} \xi_{ij} \\ s.t. \quad & \rho_{\boldsymbol{M'}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 2 - \xi_{ij} \quad \text{for } t_i \neq t_j \\ & \rho_{\boldsymbol{M'}}(\boldsymbol{x}_m, \boldsymbol{x}_n) \leq d \\ & \xi_{ij} \geq 0, \boldsymbol{M'} \in \boldsymbol{M}_{+}. \end{aligned}$$

Apply the definition of the squared Mahalanobis directly into the constraint:

$$\begin{aligned} \min_{\boldsymbol{\xi},\boldsymbol{M'},d} \quad & cd + \textstyle\sum_{i,j=1}^{N} \xi_{ij} \\ s.t. \quad & (\boldsymbol{x}_i - \boldsymbol{x}_j)(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \otimes \boldsymbol{M'} \geq 2 - \xi_{ij} \quad \text{for } t_i \neq t_j \\ & (\boldsymbol{x}_m - \boldsymbol{x}_n)(\boldsymbol{x}_m - \boldsymbol{x}_n)^T \otimes \boldsymbol{M'} \geq d \\ & \xi_{ij} \geq 0, \boldsymbol{M'} \in \boldsymbol{M}_{+}, \end{aligned}$$

where we define $\boldsymbol{A} \otimes \boldsymbol{B} = \sum_{i,j} \boldsymbol{A}_{ij} \cdot \boldsymbol{B}_{ij}$.

We now stack the columns of $\boldsymbol{M'}$ into a vector and call this vector $\boldsymbol{m}$. Similarly, we take the vectorization of $(\boldsymbol{x}_i - \boldsymbol{x}_j)(\boldsymbol{x}_i - \boldsymbol{x}_j)^T$ and $(\boldsymbol{x}_m - \boldsymbol{x}_n)(\boldsymbol{x}_m - \boldsymbol{x}_n)^T$, take their transpose and name them as $\boldsymbol{A}_{1,ij}$ and $\boldsymbol{A}_{2,mn}$, respectively. The optimisation problem is then equivalent to

$$\begin{aligned} \min_{\boldsymbol{\xi},\boldsymbol{M'},d} \quad & cd + \textstyle\sum_{i,j=1}^{N} \xi_{ij} \\ s.t. \quad & \xi_{ij} \geq 2 - \boldsymbol{A}_{1,ij}\boldsymbol{m} \quad \text{for } t_i \neq t_j \\ & d \geq \boldsymbol{A}_{2,mn}\boldsymbol{m} \\ & \xi_{ij} \geq 0, \boldsymbol{M'} \in \boldsymbol{M}_{+}, \end{aligned}$$

where

$$\begin{aligned} \boldsymbol{m} &= \text{vector}(\boldsymbol{M'}) \in \mathbb{R}^{(p \times p) \times 1}, \\ \boldsymbol{A}_{1,ij} &= [\text{vector}((\boldsymbol{x}_i - \boldsymbol{x}_j)(\boldsymbol{x}_i - \boldsymbol{x}_j)^T)]^T, \\ \boldsymbol{A}_{2,mn} &= [\text{vector}((\boldsymbol{x}_m - \boldsymbol{x}_n)(\boldsymbol{x}_m - \boldsymbol{x}_n)^T)]^T, \end{aligned}$$

$p = \dim(\boldsymbol{M'})$ and $\boldsymbol{v} = \text{vector}(\boldsymbol{V})$ reshapes any matrix $\boldsymbol{V} \in \mathbb{R}^{a \times b}$ into a vector

$\boldsymbol{v} \in \mathbb{R}^{(a\times b)\times 1}$.

Transform this problem into the consensus form [51]:

$$\begin{aligned}
\min_{\boldsymbol{\xi},\boldsymbol{M'},d} \quad & c\max_{i,j}(q_{ij}) + \textstyle\sum_{i,j=1}^{N} \max_{i,j}(0,p_{ij}) + \tilde{I}_{\boldsymbol{M}_+}(\boldsymbol{M'}) \\
s.t. \quad & \boldsymbol{p} = 2 - \boldsymbol{A}_1\boldsymbol{m}_1, \quad \boldsymbol{p} \in \mathbb{R}^{(N_1\times N_2)\times 1} \\
& \boldsymbol{q} = \boldsymbol{A}_2\boldsymbol{m}_2, \quad \boldsymbol{q} \in \mathbb{R}^{(N\times N)\times 1} \\
& \boldsymbol{m}_1 = \boldsymbol{m}_2 = \boldsymbol{m}, \quad \boldsymbol{m}_1, \boldsymbol{m}_2, \boldsymbol{m} \in \mathbb{R}^{(p\times p)\times 1},
\end{aligned}$$

where $\boldsymbol{A}_1 \in \mathbb{R}^{(N_1\times N_2)\times(p\times p)}$ consists of $(N_1 \times N_2)$ blocks of $\boldsymbol{A}_{1,ij}$ and $\boldsymbol{A}_2 \in \mathbb{R}^{(N\times N)\times(p\times p)}$ consists of $(N\times N)$ blocks of $\boldsymbol{A}_{2,mn}$. Here N_1 and N_2 are the number of instances in class 1 and 2 respectively. $\tilde{I}_C(x) = \begin{cases} 0, \ x \in C \\ \infty, \ x \notin C \end{cases}$.

The Augmented Lagrangian function of the above optimisation problem becomes

$$\begin{aligned}
& L_\mu(\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \boldsymbol{\alpha}_3, \boldsymbol{\alpha}_4, \boldsymbol{p}, \boldsymbol{q}, \boldsymbol{m}_1, \boldsymbol{m}_2, \boldsymbol{M'}) \\
=& c\max_{i,j}(q_{ij}) + \sum_{i,j=1}^{N} \max_{i,j}(0,p_{ij}) + \tilde{I}_{\boldsymbol{M}_+}(\boldsymbol{M'}) + \\
& \boldsymbol{\alpha}_1^T(\boldsymbol{m}_1 - \boldsymbol{m}) + \boldsymbol{\alpha}_2^T(\boldsymbol{m}_2 - \boldsymbol{m}) + \\
& \boldsymbol{\alpha}_3^T(\boldsymbol{p} + \boldsymbol{A}_1\boldsymbol{m}_1 - 2) + \boldsymbol{\alpha}_4^T(\boldsymbol{q} - \boldsymbol{A}_2\boldsymbol{m}_2) + \\
& \frac{\mu}{2}||\boldsymbol{m}_1 - \boldsymbol{m}||_2^2 + \frac{\mu}{2}||\boldsymbol{m}_2 - \boldsymbol{m}||_2^2 + \\
& \frac{\mu}{2}||\boldsymbol{p} + \boldsymbol{A}_1\boldsymbol{m}_1 - 2||_2^2 + \frac{\mu}{2}||\boldsymbol{q} - \boldsymbol{A}_2\boldsymbol{m}_2||_2^2,
\end{aligned}$$

where $\boldsymbol{\alpha}_1 \in \mathbb{R}^{(p\times p)\times 1}$, $\boldsymbol{\alpha}_2 \in \mathbb{R}^{(p\times p)\times 1}$, $\boldsymbol{\alpha}_3 \in \mathbb{R}^{(N_1\times N_2)\times 1}$, $\boldsymbol{\alpha}_4 \in \mathbb{R}^{(N\times N)\times 1}$ are the Lagrangian multipliers and $\mu \in \mathbb{R}^1$ is the regularisation parameter.

We apply the Alternating Direction Method of Multipliers algorithm (ADMM) to solve this problem. Specifically, we minimise $\boldsymbol{p}, \boldsymbol{q}, \boldsymbol{m}_1, \boldsymbol{m}_2, \boldsymbol{M'}$ respectively by fixing other variables and then update $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \boldsymbol{\alpha}_3, \boldsymbol{\alpha}_4$.

(1) Update p_{ij}

$$\operatorname{argmin}_{p_{ij}} L_\mu = \operatorname{argmin}_{p_{ij}} \left\{ \max(0,p_{ij}) + \boldsymbol{\alpha}_3^T p_{ij} + \frac{\mu}{2}||p_{ij} + \boldsymbol{A}_{1,ij}\boldsymbol{m}_1 - 2||_2^2 \right\}$$

According to the proposition in [81],

$$S_\lambda(\omega) = \text{argmin}_x \left\{ \lambda \max(0, x) + \frac{1}{2}||x - \omega||_2^2 \right\}$$

has the solution

$$S_\lambda(\omega) = \begin{cases} \omega - \lambda & \text{if } \omega > \lambda \\ 0 & \text{if } 0 \leq \omega \leq \lambda \\ \omega & \text{if } \omega < 0. \end{cases}$$

Our minimisation function can thus be formulated as

$$\begin{aligned} \text{argmin}_{p_{ij}} L_\mu &= \text{argmin}_{p_{ij}} \left\{ \max(0, p_{ij}) + \frac{\mu}{2}||p_{ij} - (2 - \boldsymbol{A}_{1,ij}\boldsymbol{m}_1 - \frac{\boldsymbol{\alpha}_{3,ij}}{\mu})||_2^2 \right\} \\ &= S_{\frac{1}{\mu}}(2 - \boldsymbol{A}_{1,ij}\boldsymbol{m}_1 - \frac{\boldsymbol{\alpha}_{3,ij}}{\mu}) \end{aligned}$$

Hence we have

$$p_{ij}^{t+1} = \begin{cases} 2 - \boldsymbol{A}_{1,ij}\boldsymbol{m}_1^t - \frac{\boldsymbol{\alpha}_{3,ij}^t + 1}{\mu} & \text{if } 2 - \boldsymbol{A}_{1,ij}\boldsymbol{m}_1^t - \frac{\boldsymbol{\alpha}_{3,ij}^t}{\mu} > \frac{1}{\mu} \\ 0 & \text{if } 0 \leq 2 - \boldsymbol{A}_1\boldsymbol{m}_1^t - \frac{\boldsymbol{\alpha}_3^t}{\mu} \leq \frac{1}{\mu} \\ 2 - \boldsymbol{A}_{1,ij}\boldsymbol{m}_1^t - \frac{\boldsymbol{\alpha}_{3,ij}^t}{\mu} & \text{if } 2 - \boldsymbol{A}_{1,ij}\boldsymbol{m}_1^t - \frac{\boldsymbol{\alpha}_{3,ij}^t}{\mu} < 0 \end{cases} \tag{2.18}$$

(2) Update q_{ij}

$$\text{argmin}_{q_{ij}} L_\mu = \text{argmin}_{q_{ij}} \left\{ c \max_{i,j}(q_{ij}) + \boldsymbol{\alpha}_4^T q_{ij} + \frac{\mu}{2}||q_{ij} - \boldsymbol{A}_{2,ij}\boldsymbol{m}_2||_2^2 \right\}.$$

According to [51], the optimisation function

$$\min_x \max_i x_i + \frac{1}{2\lambda}||x - v||_2^2$$

can be written as

$$\begin{aligned} &\min_x t + \tfrac{1}{2\lambda}||x - v||_2^2 \\ &s.t. \quad x_i \leq t \quad i = 1, \cdots, n. \end{aligned}$$

According to Section 6.4.1 of [51], the optimal value $t^\star$ needs to satisfy the condition

$$\sum_{i=1}^{n} \frac{1}{\lambda} \max(0, v_i - t^\star) = 1,$$

and this equation can be solved by bisection. Then, the optimal $x^\star$ can be obtained as

$$x_i^\star = \min(t^\star, v_i).$$

Therefore, we rewrite our objective function as follows:

$$\min_{q_{ij}} L_\mu \Leftrightarrow \min_{q_{ij}} \max(q_{ij}) + \frac{\mu}{2c} ||q_{ij} - (\boldsymbol{A}_{2,ij}\boldsymbol{m}_2 - \frac{\boldsymbol{\alpha}_{4,ij}}{\mu})||_2^2.$$

Hence

$$q_{ij}^{t+1} = \min(t^\star, \boldsymbol{A}_{2,ij}\boldsymbol{m}_2^t - \frac{\boldsymbol{\alpha}_{4,ij}^t}{\mu}), \tag{2.19}$$

and $t^\star$ satisfies

$$\sum_{i,j=1}^{N} \frac{\mu}{c} (\boldsymbol{A}_{2,ij}\boldsymbol{m}_2^t - \frac{\boldsymbol{\alpha}_{4,ij}^t}{\mu} - t^\star) = 1.$$

(3) Update $\boldsymbol{m}_1$

$$\begin{aligned}\min_{\boldsymbol{m}1} L_\mu \Leftrightarrow \min_{\boldsymbol{m}1} \quad & \boldsymbol{\alpha}_1^T \boldsymbol{m}_1 + \boldsymbol{\alpha}_3^T \boldsymbol{A}_1 \boldsymbol{m}_1 + \\ & \frac{\mu}{2} ||\boldsymbol{m}_1 - \boldsymbol{m}||_2^2 + \frac{\mu}{2} ||\boldsymbol{p} + \boldsymbol{A}_1 \boldsymbol{m}_1 - 2||_2^2.\end{aligned}$$

Take the derivative with respect to $\boldsymbol{m}_1$, we get

$$\mu(\boldsymbol{A}_1^T \boldsymbol{A}_1 + \mathbf{I})\boldsymbol{m}_1^\star + \boldsymbol{\alpha}_1 + \boldsymbol{A}_1^T \boldsymbol{\alpha}_3 - \mu \boldsymbol{m} + \mu \boldsymbol{A}_1^T \boldsymbol{p} - 2\mu \boldsymbol{A}_1^T \mathbf{1} = \mathbf{0},$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{1}$ is the vector with all components being 1. Hence, we update $\boldsymbol{m}_1$ as follows:

$$\boldsymbol{m}_1^{t+1} = (\boldsymbol{A}_1^T \boldsymbol{A}_1 + \mathbf{I})^{-1} (\boldsymbol{m}^t - \frac{\boldsymbol{\alpha}_1^t + \boldsymbol{A}_1^T \boldsymbol{\alpha}_3^t + \mu \boldsymbol{A}_1^T \boldsymbol{p}^{t+1} - 2\mu \boldsymbol{A}_1^T \mathbf{1}}{\mu}). \tag{2.20}$$

We can save $(\boldsymbol{A}_1^T \boldsymbol{A}_1 + \mathbf{I})^{-1}$ in the memory so as to improve the computational efficiency.

(4) Update $\boldsymbol{m}_2$

$$\begin{aligned}\min_{\boldsymbol{m}2} L_\mu \Leftrightarrow \min_{\boldsymbol{m}2} \quad & \boldsymbol{\alpha}_2^T \boldsymbol{m}_2 - \boldsymbol{\alpha}_4^T \boldsymbol{A}_2 \boldsymbol{m}_2 + \\ & \frac{\mu}{2} ||\boldsymbol{m}_2 - \boldsymbol{m}||_2^2 + \frac{\mu}{2} ||\boldsymbol{q} - \boldsymbol{A}_2 \boldsymbol{m}_2||_2^2.\end{aligned}$$

Take the derivative with respect to $\boldsymbol{m}_2$, we get

$$\mu(\boldsymbol{A}_2^T \boldsymbol{A}_2 + \mathbf{I})\boldsymbol{m}_2^\star + \boldsymbol{\alpha}_2 - \boldsymbol{A}_2^T \boldsymbol{\alpha}_4 - \mu \boldsymbol{m} - \mu \boldsymbol{A}_2^T \boldsymbol{q} = 0.$$

Update $\boldsymbol{m}_2$ as follows:

$$\boldsymbol{m}_2^{t+1} = (\boldsymbol{A}_2^T \boldsymbol{A}_2 + \mathbf{I})^{-1} (\boldsymbol{m}^t + \frac{\boldsymbol{A}_2^T \boldsymbol{\alpha}_4^t + \mu \boldsymbol{A}_2^T \boldsymbol{q}^{t+1} - \boldsymbol{\alpha}_2^t}{\mu}). \tag{2.21}$$

(5) Update $\boldsymbol{M'}$ *(and hence* $\boldsymbol{m}$)

$$\begin{aligned}\min_{\boldsymbol{M'}/\boldsymbol{m}} \tilde{I}_{\boldsymbol{M}_+}(\boldsymbol{M'}) + \boldsymbol{\alpha}_1^T(\boldsymbol{m}_1 - \boldsymbol{m}) + \boldsymbol{\alpha}_2^T(\boldsymbol{m}_2 - \boldsymbol{m}) \\ + \tfrac{\mu}{2}||\boldsymbol{m}_1 - \boldsymbol{m}||_2^2 + \tfrac{\mu}{2}||\boldsymbol{m}_2 - \boldsymbol{m}||_2^2.\end{aligned}$$

Hence, update $\boldsymbol{m}$ as

$$\begin{aligned}\boldsymbol{m}^{t+1} = & \prod\nolimits_{\boldsymbol{M}_+} \big(\text{matrix}(\frac{\boldsymbol{m}_1^{t+1} + \boldsymbol{m}_2^{t+1}}{2} + \frac{\boldsymbol{\alpha}_1^t + \boldsymbol{\alpha}_2^t}{2\mu}) + \\ & \text{matrix}(\frac{\boldsymbol{m}_1^{t+1} + \boldsymbol{m}_2^{t+1}}{2} + \frac{\boldsymbol{\alpha}_1^t + \boldsymbol{\alpha}_2^t}{2\mu})'\big)/2,\end{aligned} \tag{2.22}$$

where $\boldsymbol{V} = \text{matrix}(\boldsymbol{v})$ is the reverse operation of $\boldsymbol{v} = \text{vector}(\boldsymbol{V})$ and it reshapes a vector $v \in \mathbb{R}^{(p\times p)\times 1}$ into a matrix $\boldsymbol{V} \in \mathbb{R}^{p\times p}$. $\prod_{\boldsymbol{M}_+}$ denotes the projection of a symmetric matrix onto the positive semi-definite cone $\boldsymbol{M}_+$.

(6) Update $\boldsymbol{\alpha}$

$$\begin{aligned}\boldsymbol{\alpha}_1^{t+1} &= \boldsymbol{\alpha}_1^t + \mu(\boldsymbol{m}_1^{t+1} - \boldsymbol{m}^{t+1}) \\ \boldsymbol{\alpha}_2^{t+1} &= \boldsymbol{\alpha}_2^t + \mu(\boldsymbol{m}_2^{t+1} - \boldsymbol{m}^{t+1}) \\ \boldsymbol{\alpha}_3^{t+1} &= \boldsymbol{\alpha}_3^t + \mu(\boldsymbol{p}^{t+1} + \boldsymbol{A}_1 \boldsymbol{m}_1^{t+1} - 2) \\ \boldsymbol{\alpha}_4^{t+1} &= \boldsymbol{\alpha}_4^t + \mu(\boldsymbol{q}^{t+1} - \boldsymbol{A}_2 \boldsymbol{m}_2^{t+1}).\end{aligned} \tag{2.23}$$

Chapter 3

Metric Learning with Local Metrics

3.1 Introduction

Classification is a long-standing area of machine learning. While deep learning classifiers have obtained superior performance on numerous applications, they generally require a large amount of labelled data. For small data sets, traditional classification algorithms remain valuable.

The nearest neighbour (NN) classifier is one of the most commonly used methods for classification, which determines the class label based on the distances between a new instance and the training instances. However, with different metrics, the performance of NN could be quite different. Hence it is very beneficial to 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 available data. Metric learning with a convex objective function was first proposed in the seminal work of Xing [78]. After that, many other metric learning methods have been developed and widely adopted, such as the large margin nearest neighbour (LMNN) [76] and the information theoretic metric learning [9]. Some theoretical work has also been proposed for metric learning, especially on deriving different generalisation bounds [33, 8, 20, 67], and deep networks have been used to represent nonlinear metrics [24, 40]. In addition, metric learning methods have been developed for specific purposes, including multi-output tasks [39], multi-view learning [25], medical image retrieval [80], kinship verification tasks [79], face

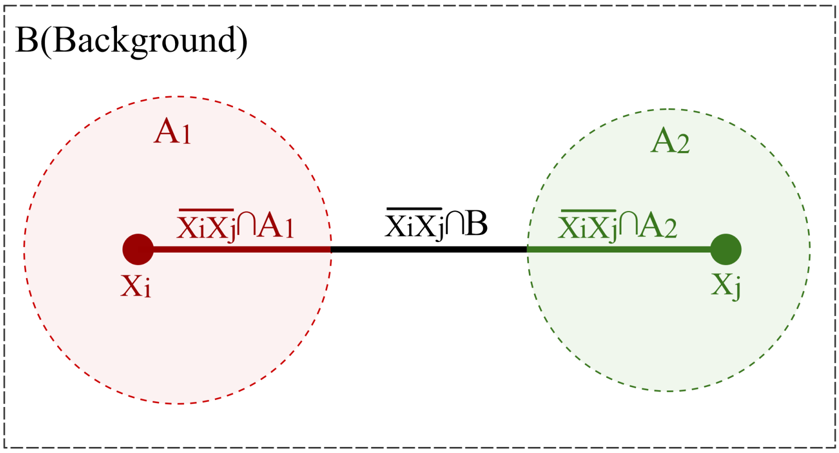

Figure 3.1: An example of calculating the distance between points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ with local metrics. A_1 and A_2 are different influential regions with metrics $M(A_1)$ and $M(A_2)$ and B is the background region with metric $\boldsymbol{M}(B)$. The distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ equals to the sum of three line segments' local distances, namely $l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j} \cap A_1; M(A_1))$, $l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j} \cap A_2; M(A_2))$ and $l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j} \cap B; \boldsymbol{M}(B))$.

recognition tasks [27], tracking problems [69] and so on.

Most aforementioned methods use a single metric for the whole metric space and thus may not suit well for data sets with multimodality. To solve this problem, local metric learning algorithms have been proposed [14, 76, 70, 26, 5, 59, 55, 62, 49].

Most of these localised algorithms could be categorised 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 [70], i.e. $M(\boldsymbol{x}_i) \neq M(\boldsymbol{x}_j)$ may lead to an unequal distance $\rho(\boldsymbol{x}_i, \boldsymbol{x}_j; M(\boldsymbol{x}_i)) \neq \rho(\boldsymbol{x}_j, \boldsymbol{x}_i; M(\boldsymbol{x}_j))$. 2) Each line segment or cluster of line segments has a local metric $M(\boldsymbol{x}_i, \boldsymbol{x}_j)$. The definitions of $M(\boldsymbol{x}_i, \boldsymbol{x}_j)$, such as $\sum_k w_k(\boldsymbol{x}_i, \boldsymbol{x}_j)M_k$ in [5] where w_k is defined as $P(k|\boldsymbol{x}_i) + P(k|\boldsymbol{x}_j)$ to guarantee the symmetry and $P(k|\boldsymbol{x}_i)$ or $P(k|\boldsymbol{x}_j)$ is the posterior probability that $\boldsymbol{x}$ belongs to the kth component in the Gaussian mixture model(GMM), are nonetheless not very intuitive.

In this chapter, an intuitive, symmetric distance and a novel local metric learning method are proposed. By splitting the metric space into influential regions and a background region, the distance between any two points is defined as the sum of

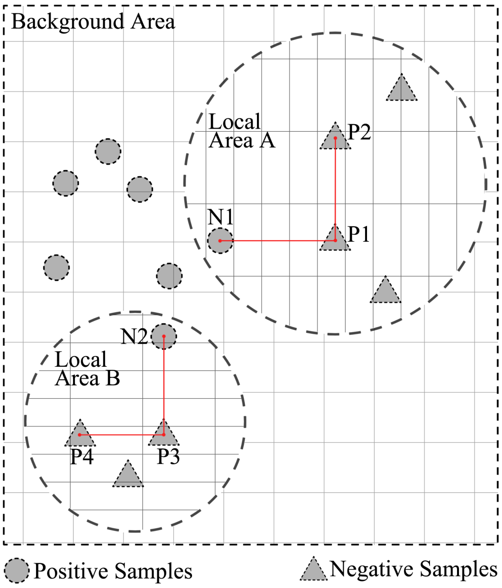

Figure 3.2: An illustration of the benefits of learning local influential regions. The distance between the adjacent vertical/horizontal grids is one unit. The multimodality issue is solved by dividing positive samples into two local regions. A suitable local metric helps increase the class separability, such as increasing $l(\overline{N_1 P_1})$ and $l(\overline{N_2 P_3})$ while decreasing $l(\overline{P_1 P_2})$ and $l(\overline{P_3 P_4})$.

lengths of line segments in each region, as illustrated in Figure 3.1. Building multiple influential regions solves the multimodality issue and learning a suitable local metric in each influential region improves class separability, as shown in Figure 3.2.

To establish the proposed new distance and local metric learning method, the rest of this chapter is organised as follows. First, in Section 3.2, some key concepts are introduced, namely influential regions, local metrics and line segments, which lead to a new definition of the distance. Next, in Section 3.3, we calculate the distance by discussing the geometric relationship between line segment and influential regions. Then, in Section 3.4 we build a novel classifier based on the proposed local metric and its learnablity is studied in Section 3.4.2. After that in Section 3.5, we formulate a non-convex optimisation problem using the empirical hinge loss and regularisation terms from the derived learning bound and solve it via the gradient descent algorithm. In Section 3.6, the proposed local metric learning algorithm is tested on 14 publicly available data sets. It achieves the best performance on eight of these data sets, much better than state-of-the-art competitors. Section 3.7 presents some concluding remarks and future work.

3.2 Definitions of Influential Regions, Local Metrics and Distance

In this section, influential regions $A_s, s = 1, \ldots, S$, and the background region B will be defined first. Next, 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 with corresponding local metrics $\boldsymbol{M}(A_s)$ and $\boldsymbol{M}(B)$, as illustrated in Figure 3.1. Since the metric is defined with respect to line segments, the distance is symmetric, that is $\rho(\boldsymbol{x}_i, \boldsymbol{x}_j) = \rho_{M(\overline{\boldsymbol{x}_i\boldsymbol{x}_j})}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \rho_{M(\overline{\boldsymbol{x}_j\boldsymbol{x}_i})}(\boldsymbol{x}_j, \boldsymbol{x}_i) = \rho(\boldsymbol{x}_j, \boldsymbol{x}_i)$.

To simplify later calculations, the shape of each influential region is restricted to be a ball.

Definition 15. *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; $A_s = Ball(\boldsymbol{o}_s, r_s)$, in which $Ball(\boldsymbol{o}_s, r_s)$ denotes a ball with the centre at $\boldsymbol{o}_s$ and radius of r_s; the location of each influential region is determined by the Euclidean distance; and points $\boldsymbol{x} \in A_s$ form a set with the following form

$$\{\boldsymbol{x} | (\boldsymbol{o}_s - \boldsymbol{x})^T (\boldsymbol{o}_s - \boldsymbol{x}) \leq r_s^2\}. \tag{3.1}$$

Definition 16. *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 chapter, 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. $\rho(\boldsymbol{x}_i, \boldsymbol{x}_j) = l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j})$. Length $l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j})$

in influential regions and the background region will be defined separately with respective metrics.

Definition 17. Each influential region A_s has its own *local metric* $\boldsymbol{M}(A_s)$. The length of a line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ inside an influential region A_s is defined as[1]

$$\begin{aligned} l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j}; \boldsymbol{M}(A_s)) =&\rho_{\boldsymbol{M}(A_s)}(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ =&\sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{M}(A_s)(\boldsymbol{x}_i - \boldsymbol{x}_j)}. \end{aligned} \tag{3.2}$$

To make illustrations more intuitive, the distance adopted in this chapter would be based on the Mahalanobis distance[2].

Definition 18. The background region B has a *background metric* $\boldsymbol{M}(B)$. 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

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

Two remarks are made here:

1. While the metrics $\boldsymbol{M}(A_s)$ and $\boldsymbol{M}(B)$ will be learned inside influential regions and the background region, the Euclidean distance is used to determine the location of influential regions.

2. 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 generally different from $\rho_{\boldsymbol{M}(B)}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. This is because some parts of the line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ may lie in influential regions so their lengths should be calculated via the related local metrics.

To calculate the distance between any $\boldsymbol{x}_i \in U$ and $\boldsymbol{x}_j \in U$, the relationship between the line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ and influential regions needs to be considered, which

[1] Since influential regions are restricted to be ball-shaped and a ball is a convex set, the line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ lies in the ball for any two point $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ inside the ball.

[2] This is different from the widely adopted squared Mahalanobis distance and enjoys convenience when solving the optimisation problem.

can be simplified to one of the following three cases: no-intersection, tangent and with-intersection.

Definition 19. The *intersection* of a line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ and an influential region A_s is denoted as $A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}$. In the case of no-intersection, $A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j} = \emptyset$; in the case of tangent, $A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j} = \boldsymbol{t}_{ij}^s$, where $\boldsymbol{t}_{ij}^s$ is the tangent point; in the case of with-intersection, $A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j} = \overline{\boldsymbol{p}_{ij}^s\boldsymbol{q}_{ij}^s}$, where $\overline{\boldsymbol{p}_{ij}^s\boldsymbol{q}_{ij}^s}$ is the maximum sub-line segment of $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ inside A_s, $\boldsymbol{p}_{ij}^s$ is the point which lies closer to $\boldsymbol{x}_i$ and $\boldsymbol{q}_{ij}^s$ is the point which lies closer to $\boldsymbol{x}_j$. The *intersection* of a line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ and the background region B is defined as

$$B \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j} = \overline{\boldsymbol{x}_i\boldsymbol{x}_j} - \bigcup_{s=1\ldots S} (A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}), \tag{3.3}$$

where $\bigcup_{s=1\ldots S}(A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j})$ is the union of intersections between the line segment and all influential regions. It can also be understood as a set of non-overlapping line segments[3].

Accordingly, the length of line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ can be calculated through the length of intersection.

Definition 20. The *length of intersection* of a line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ and an influential region A_s is defined as $l(A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}; \boldsymbol{M}(A_s))$. In the case of tangent or no-intersection, $l(A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}; \boldsymbol{M}(A_s)) \triangleq 0$; in the case of with-intersection, it is defined to be the length of $\overline{\boldsymbol{p}_{ij}^s\boldsymbol{q}_{ij}^s}$, i.e. $l(A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}; \boldsymbol{M}(A_s)) = l(\overline{\boldsymbol{p}_{ij}^s\boldsymbol{q}_{ij}^s}; \boldsymbol{M}(A_s))$. The *length of the intersection* of a line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ and the background region B is defined as

$$\begin{aligned} l(B \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}; \boldsymbol{M}(B)) = {} & l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j}; \boldsymbol{M}(B)) \\ & - l(\bigcup_{s=1\ldots S} (A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}); \boldsymbol{M}(B)). \end{aligned} \tag{3.4}$$

[3] This can be easily proved by recursively combining any overlapping line segments until no overlapping one is found.

Table 3.1: A summary of the notations in Chapter 2.

Notation	Detail
a	$(\boldsymbol{x}_j - \boldsymbol{x}_i)^T(\boldsymbol{x}_j - \boldsymbol{x}_i)$
b	$2(\boldsymbol{x}_j - \boldsymbol{x}_i)^T(\boldsymbol{x}_i - \boldsymbol{o}_s)$
c	$(\boldsymbol{x}_i - \boldsymbol{o}_s)^T(\boldsymbol{x}_i - \boldsymbol{o}_s) - r_s^2$
Δ	$b^2 - 4ac$
λ_u	$\frac{-b-\sqrt{\Delta}}{2a}$
λ_v	$\frac{-b+\sqrt{\Delta}}{2a}$
$\boldsymbol{u}$	$\boldsymbol{x}_i + \lambda_u(\boldsymbol{x}_j - \boldsymbol{x}_i)$
$\boldsymbol{v}$	$\boldsymbol{x}_i + \lambda_v(\boldsymbol{x}_j - \boldsymbol{x}_i)$
$\boldsymbol{p}$	$\boldsymbol{x}_i + \lambda_p(\boldsymbol{x}_j - \boldsymbol{x}_i)$
$\boldsymbol{q}$	$\boldsymbol{x}_i + \lambda_q(\boldsymbol{x}_j - \boldsymbol{x}_i)$
γ	$\lambda_q - \lambda_p$

Definition 21. The *length of line segment* is defined as

$$\begin{aligned} l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j}; \boldsymbol{M}(\overline{\boldsymbol{x}_i\boldsymbol{x}_j})) &= \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{M}(\overline{\boldsymbol{x}_i\boldsymbol{x}_j})(\boldsymbol{x}_i - \boldsymbol{x}_j)} \\ &= l(B \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}; \boldsymbol{M}(B)) \\ &+ \sum_s l(A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}; \boldsymbol{M}(A_s)), \end{aligned} \tag{3.5}$$

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

3.3 Calculation of Distances

3.3.1 Length of Intersection with Influential Regions

We first give an intuitive explanation on calculating the length of intersection with influential regions, as illustrated in Figure 3.3. If the line $\boldsymbol{x}_i\boldsymbol{x}_j$ does not intersect with or is the tangent to the influential ball, the length is zero. This is equivalent to identifying the start and end points of line $\boldsymbol{x}_i\boldsymbol{x}_j$ and the ball, $\boldsymbol{u}, \boldsymbol{v}$, via one variable quadratic equation. If the line intersects with the ball, the length would be calculated by considering the relationship between the intersection of the line $\boldsymbol{x}_i\boldsymbol{x}_j$ and the influential ball, i.e. $\boldsymbol{uv}$, and the intersection of the line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ and the influential ball, i.e. $\boldsymbol{pq}$. $\boldsymbol{p}, \boldsymbol{q}$ can be obtained based on points $\boldsymbol{u}, \boldsymbol{v}$ and the constraint

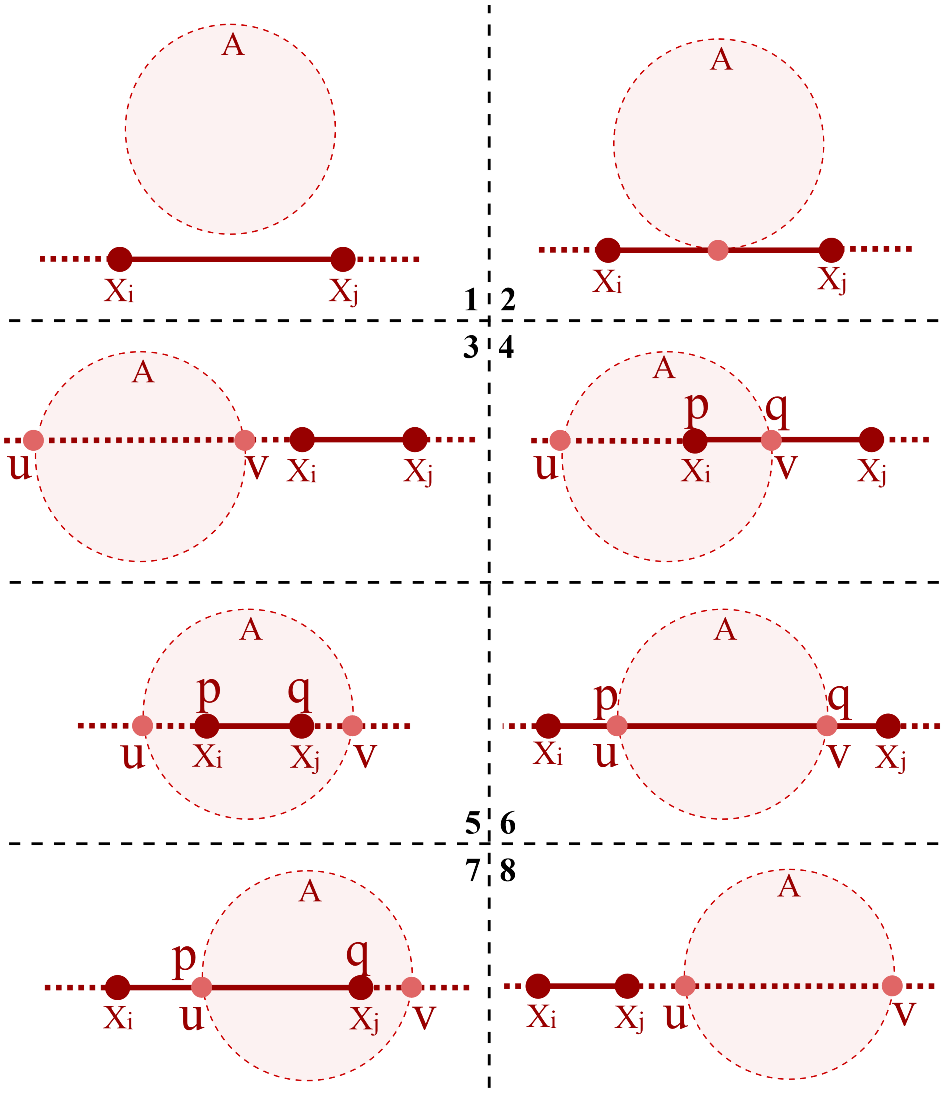

Figure 3.3: An illustration of the relationship between $\boldsymbol{u}, \boldsymbol{v}$ and $\boldsymbol{p}, \boldsymbol{q}$. The positions of $\boldsymbol{u}, \boldsymbol{v}$ (intersection points between line $\boldsymbol{x}_i\boldsymbol{x}_j$ and the influential region A) and $\boldsymbol{p}, \boldsymbol{q}$ (intersection points between line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ and A) under different situations.

Table 3.2: The intersection of line segment and influential region under different cases. The column of 'Line $\boldsymbol{x}_i\boldsymbol{x}_j$' indicates the relationship between the line $\boldsymbol{x}_i\boldsymbol{x}_j$ and the influential region, which is determined by the value of Δ; 'no-inter' stands for no-intersection and 'inter' stands for with-intersection. The column of '$\overline{\boldsymbol{pq}}$' indicates the relationship between the line segment $\overline{\boldsymbol{pq}}$ and the influential regions, which is determined by the values of λ_u and λ_v. The column 'case' refers to the corresponding case in Figure 3.3.

Δ	Line $\boldsymbol{x}_i\boldsymbol{x}_j$	Values of λ_u λ_v	$\overline{\boldsymbol{pq}}$	Values of λ_p λ_q	$l = l(A_s \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}; \boldsymbol{M}(A_s))$	Case
$\Delta < 0$	no-inter				0	1
$\Delta = 0$	tangent				0	2
$\Delta > 0$	inter	$\lambda_u < 0, \lambda_v < 0$	$\emptyset$	$\lambda_p, \lambda_q \triangleq 0$	$\gamma\sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{M}(A_s)(\boldsymbol{x}_i - \boldsymbol{x}_j)}$, where $\gamma = \lambda_q - \lambda_p$	3
		$\lambda_u < 0, 0 \le \lambda_v \le 1$	$\overline{\boldsymbol{x}_i\boldsymbol{v}}$	$\lambda_p = 0, \lambda_q = \lambda_v$		4
		$\lambda_u < 0, \lambda_v > 1$	$\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$	$\lambda_p = 0, \lambda_q = 1$		5
		$0 \le \lambda_u, \lambda_v \le 1$	$\overline{\boldsymbol{uv}}$	$\lambda_p = \lambda_u, \lambda_q = \lambda_v$		6
		$0 \le \lambda_u \le 1, \lambda_v > 1$	$\overline{\boldsymbol{ux}_j}$	$\lambda_p = \lambda_u, \lambda_q = 1$		7
		$\lambda_u > 1, \lambda_v > 1$	$\emptyset$	$\lambda_p, \lambda_q \triangleq 1$		8

that the start and end points should be on the line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$.

Definition 22. 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 $\lambda_u, \lambda_v \in \mathbb{R}$, $\lambda_u \le \lambda_v$ 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 $0 \le \lambda_p \le \lambda_q \le 1$ 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 8. 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

$$\begin{aligned} l(A \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}; \boldsymbol{M}(A_s)) &= \sqrt{(\boldsymbol{q} - \boldsymbol{p})^T \boldsymbol{M}(A_s)(\boldsymbol{q} - \boldsymbol{p})} \\ &= \gamma\sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{M}(A_s)(\boldsymbol{x}_i - \boldsymbol{x}_j)}. \end{aligned} \tag{3.6}$$

As shown in the above proposition, the length of intersection can be calculated given the local metric $\boldsymbol{M}(A_s)$ and γ, where the latter term can be obtained from λ_q and λ_p.

The computation of γ consists of two steps.

1) Calculate the intersection points of the line $\boldsymbol{x}_i\boldsymbol{x}_j$ and the ball: $\boldsymbol{u}$ and $\boldsymbol{v}$, i.e. $\boldsymbol{x}_i + \lambda_u(\boldsymbol{x}_j - \boldsymbol{x}_i)$ and $\boldsymbol{x}_i + \lambda_v(\boldsymbol{x}_j - \boldsymbol{x}_i)$.

The coefficients λ_u and λ_v could be easily solved through the following quadratic equation with one variable:

$$\|\boldsymbol{x}_i + \lambda(\boldsymbol{x}_j - \boldsymbol{x}_i) - \boldsymbol{o}_s\|_2^2 = r_s^2, \tag{3.7}$$

with $\Delta = b^2 - 4ac = [2(\boldsymbol{x}_j - \boldsymbol{x}_i)^T(\boldsymbol{x}_i - \boldsymbol{o}_s)]^2 - 4[(\boldsymbol{x}_j - \boldsymbol{x}_i)^T(\boldsymbol{x}_j - \boldsymbol{x}_i)][(\boldsymbol{x}_i - \boldsymbol{o}_s)^T(\boldsymbol{x}_i - \boldsymbol{o}_s) - r_s^2]$; and when $\Delta > 0$, the solutions $\lambda^s_{u,ij} \leq \lambda^s_{v,ij}$ to the above equation are

$$\begin{aligned} \lambda^s_{u,ij} &= \frac{-b - \sqrt{\Delta}}{2a} = \frac{-2(\boldsymbol{x}_j - \boldsymbol{x}_i)^T(\boldsymbol{x}_i - \boldsymbol{o}_s) - \sqrt{\Delta}}{2(\boldsymbol{x}_j - \boldsymbol{x}_i)^T(\boldsymbol{x}_j - \boldsymbol{x}_i)}, \\ \lambda^s_{v,ij} &= \frac{-b + \sqrt{\Delta}}{2a} = \frac{-2(\boldsymbol{x}_j - \boldsymbol{x}_i)^T(\boldsymbol{x}_i - \boldsymbol{o}_s) + \sqrt{\Delta}}{2(\boldsymbol{x}_j - \boldsymbol{x}_i)^T(\boldsymbol{x}_j - \boldsymbol{x}_i)}. \end{aligned}$$

Hence the two intersection points between the ball and the line become

$$\begin{aligned} \boldsymbol{u}^s_{ij} &= \boldsymbol{x}_i + \lambda^s_{u,ij}(\boldsymbol{x}_j - \boldsymbol{x}_i), \\ \boldsymbol{v}^s_{ij} &= \boldsymbol{x}_i + \lambda^s_{v,ij}(\boldsymbol{x}_j - \boldsymbol{x}_i). \end{aligned}$$

For simplicity, the superscript s and subscript ij for λ, u, v, p and q would be discarded if no confusion is caused.

2) Calculate the intersection points of the line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ and the ball: $\boldsymbol{p}$ and $\boldsymbol{q}$, i.e. $\boldsymbol{x}_i + \lambda_p(\boldsymbol{x}_j - \boldsymbol{x}_i)$ and $\boldsymbol{x}_i + \lambda_q(\boldsymbol{x}_j - \boldsymbol{x}_i)$.

The number of solutions to (3.7) would be checked. If (3.7) has 0 or 1 solution, the line has no intersection or is tangent to the region, and thus $l(A \cap \overline{\boldsymbol{x}_i\boldsymbol{x}_j}; \boldsymbol{M}(A_s)) = 0$. If it has two solutions, the intersection between the line and the ball A_s is a line segment $\overline{\boldsymbol{u}\boldsymbol{v}}$. Based on the value of λ_u, λ_v[4], The relationship between $\overline{\boldsymbol{u}\boldsymbol{v}}$ and $\overline{\boldsymbol{p}\boldsymbol{q}}$ could be obtained and the values of λ_p and λ_q could be calculated

[4] If and only if the value of λ_u or λ_v lies in the range of $[0, 1]$, the corresponding point lies inside the line segment $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$.

from

$$\lambda_p = \min(\max(\lambda_u, 0), 1),$$
$$\lambda_q = \min(\max(\lambda_v, 0), 1).$$

A summary of the notation used in this section is listed in Table 3.1; the details of the distance calculation are illustrated in Figure 3.3 and Table 3.2.

3.3.2 Length of Intersection with Local Metrics

Proposition 9. In the case of non-overlapping influential regions, i.e. $A_i \cap A_j = \emptyset, \forall i \neq j$,

$$\begin{aligned}\rho_{\boldsymbol{M}}(\boldsymbol{x}_i\boldsymbol{x}_j) &\triangleq l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j}; M(\overline{\boldsymbol{x}_i\boldsymbol{x}_j})) \\ &= \gamma_b\sqrt{(\boldsymbol{x}_i-\boldsymbol{x}_j)^T\boldsymbol{M}(B)(\boldsymbol{x}_i-\boldsymbol{x}_j)} \\ &+\sum_s \gamma_s\sqrt{(\boldsymbol{x}_i-\boldsymbol{x}_j)^T\boldsymbol{M}(A_s)(\boldsymbol{x}_i-\boldsymbol{x}_j)} \\ &= (1-\sum_s\gamma_s)\sqrt{(\boldsymbol{x}_i-\boldsymbol{x}_j)^T\boldsymbol{M}(B)(\boldsymbol{x}_i-\boldsymbol{x}_j)} \\ &+\sum_s \gamma_s\sqrt{(\boldsymbol{x}_i-\boldsymbol{x}_j)^T\boldsymbol{M}(A_s)(\boldsymbol{x}_i-\boldsymbol{x}_j)},\end{aligned} \tag{3.8}$$

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

Proposition 9 suggests that the distance can be obtained once metrics ($\boldsymbol{M}(A_s)$, $\boldsymbol{M}(B)$) and the intersection ratio γ_s are known. As all calculations are in closed form, the computation is efficient.

In the case of overlapping influential regions, the following formula is the same as (3.8)

$$\begin{aligned}\rho_{\boldsymbol{M}}(\boldsymbol{x}_i\boldsymbol{x}_j) &\triangleq l(\overline{\boldsymbol{x}_i\boldsymbol{x}_j}; M(\overline{\boldsymbol{x}_i\boldsymbol{x}_j})) \\ &= \gamma_b\sqrt{(\boldsymbol{x}_i-\boldsymbol{x}_j)^T\boldsymbol{M}(B)(\boldsymbol{x}_i-\boldsymbol{x}_j)} \\ &+\sum_s \gamma_s\sqrt{(\boldsymbol{x}_i-\boldsymbol{x}_j)^T\boldsymbol{M}(A_s)(\boldsymbol{x}_i-\boldsymbol{x}_j)}.\end{aligned} \tag{3.9}$$

The calculation of γ_b in (3.9) is slightly different from that in (3.8). In the following sections, an approximation of γ_b is used for simplicity: $\gamma_b = \max(1 - \sum_s \gamma_s, 0)$.

3.4 Learnability of the Classifier

In this section, we select Lipschitz continuous functions as the classifiers. Based on the resultant learning bounds, we obtain the regularisation terms in order to improve the generalisation ability.

3.4.1 Classifier

In the Euclidean space, it is intuitive to see the following classifier gives the same classification results as 1-NN:

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

where $h(\boldsymbol{x}) < 0$ indicates that $\boldsymbol{x}$ belongs to negative class and $h(\boldsymbol{x}) > 0$ indicates that $\boldsymbol{x}$ belongs to positive class; $\rho_{set}(\boldsymbol{x}, \boldsymbol{X}^{-/+}) = \{\rho(\boldsymbol{x}, \boldsymbol{x}_t) | \forall \boldsymbol{x}_t \in$ negative class / positive class$\}$ is the set that contains the Euclidean distance values between $\boldsymbol{x}$ and any instance of the negative or positive class, and $\rho(\boldsymbol{x}_i, \boldsymbol{x}_j)$ indicates the Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$.

Similarly, we extend the above equation to consider more nearby instances as follows:

$$h(\boldsymbol{x}) = \frac{1}{K} \operatorname{sumKmin} \rho_{set}(\boldsymbol{x}, \boldsymbol{X}^-) - \frac{1}{K} \operatorname{sumKmin} \rho_{set}(\boldsymbol{x}, \boldsymbol{X}^+), \tag{3.10}$$

where $\operatorname{sumKmin}$ denotes the sum of the K minimal elements of the set. This function is used as the classifier in our algorithm.

3.4.2 Learnability of the Classifier with Local Metrics

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

Definition 23. ([74]) The *Lipschitz constant* of a function f with respect to input $\boldsymbol{x}$

is

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

Proposition 10. ([74]) Let $\mathrm{lip}(f \leftarrow \boldsymbol{x}) \leq L_f$ and $\mathrm{lip}(g \leftarrow \boldsymbol{x}) \leq L_g$, then

(a) $\mathrm{lip}(f + g \leftarrow \boldsymbol{x}) \leq L_f + L_g$;

(b) $\mathrm{lip}(f - g \leftarrow \boldsymbol{x}) \leq L_f + L_g$;

(c) $\mathrm{lip}(af \leftarrow \boldsymbol{x}) \leq |a|L_f$, where a is a constant.

Proposition 11. Let $\mathrm{lip}(f_k \leftarrow \boldsymbol{x}) \leq L_k, k = 1, \ldots, K$, then $\mathrm{lip}(\mathrm{sumKmin}\, f_k \leftarrow \boldsymbol{x})$ is bounded by $K \max_k L_k$, where $\mathrm{sumKmin}\, f_k$ denotes the function of $\mathrm{sumKmin}\{f_k(\boldsymbol{x}), k = 1, \ldots, K\}$.

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

$$\begin{aligned}&\mathrm{sumKmin}\{f_k(\boldsymbol{x}_i)\} \\ =&\mathrm{sumKmin}\{f_k(\boldsymbol{x}_j + (\boldsymbol{x}_i - \boldsymbol{x}_j))\} \\ \leq&\mathrm{sumKmin}\{f_k(\boldsymbol{x}_j) + L_k\|\boldsymbol{x}_i - \boldsymbol{x}_j\|\} \\ \leq&\mathrm{sumKmin}\{f_k(\boldsymbol{x}_j) + (\max_k L_k)\|\boldsymbol{x}_i - \boldsymbol{x}_j\|\} \\ =&\mathrm{sumKmin}\{f_k(\boldsymbol{x}_j)\} + K(\max_k L_k)\|\boldsymbol{x}_i - \boldsymbol{x}_j\|.\end{aligned}$$

Therefore,

$$\mathrm{sumKmin}\{f_k(\boldsymbol{x}_i)\} - \mathrm{sumKmin}\{f_k(\boldsymbol{x}_j)\} \leq K(\max_k L_k)\|\boldsymbol{x}_i - \boldsymbol{x}_j\|.$$

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

Lemma 7. With the distance defined in (3.9), the Lipschitz constant of the classifier specified in (3.10) is bound by $2(\sum_s \sqrt{\|\boldsymbol{M}(A_s)\|_F} + \sqrt{\|\boldsymbol{M}(B)\|_F})$, where $\|\cdot\|_F$ denotes the matrix Frobenius norm.

Proof. Let $d_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}_k)$ denote the Mahalanobis distance with metric $\boldsymbol{M}$, that is

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

and $d_{\boldsymbol{I}}(\boldsymbol{x}, \boldsymbol{x}_k)$ denote the Euclidean distance.

$\text{lip}(f_1 \leftarrow \boldsymbol{x})$, where $f_1 = d_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}_k)$, is bounded by $\|\boldsymbol{M}\|_F$ as follows:

$$\begin{aligned}
\text{lip}(f_1 \leftarrow \boldsymbol{x}) &= \max_{\boldsymbol{x}_a, \boldsymbol{x}_b \in \mathcal{X}, \boldsymbol{x}_a \neq \boldsymbol{x}_b} \frac{f_1(\boldsymbol{x}_a) - f_1(\boldsymbol{x}_b)}{d_{\boldsymbol{I}}(\boldsymbol{x}_a, \boldsymbol{x}_b)} \\
&\leq \max_{\boldsymbol{x}_a, \boldsymbol{x}_b \in \mathcal{X}, \boldsymbol{x}_a \neq \boldsymbol{x}_b} \frac{d_{\boldsymbol{M}}(\boldsymbol{x}_a, \boldsymbol{x}_b)}{d_{\boldsymbol{I}}(\boldsymbol{x}_a, \boldsymbol{x}_b)} \\
&\leq \max_{\boldsymbol{x}_a, \boldsymbol{x}_b \in \mathcal{X}, \boldsymbol{x}_a \neq \boldsymbol{x}_b} \frac{d_{\boldsymbol{I}}(\boldsymbol{x}_a, \boldsymbol{x}_b)\sqrt{\|\boldsymbol{M}\|_F}}{d_{\boldsymbol{I}}(\boldsymbol{x}_a, \boldsymbol{x}_b)} \\
&= \sqrt{\|\boldsymbol{M}\|_F},
\end{aligned}$$

where the first inequality follows the triangle inequality of distance, and the second inequality is based on the fact that matrix Frobenius norm is consistent with the vector L_2-norm[5], i.e.

$$\begin{aligned}
d_{\boldsymbol{M}}(\boldsymbol{x}_a, \boldsymbol{x}_b) &= \sqrt{\|(\boldsymbol{x}_a - \boldsymbol{x}_b)^T \boldsymbol{M} (\boldsymbol{x}_a - \boldsymbol{x}_b)\|_2} \\
&\leq \sqrt{\|\boldsymbol{x}_a - \boldsymbol{x}_b\|_2^2 \|\boldsymbol{M}\|_F} \\
&= \|\boldsymbol{x}_a - \boldsymbol{x}_b\|_2 \sqrt{\|\boldsymbol{M}\|_F} \\
&= d_{\boldsymbol{I}}(\boldsymbol{x}_a, \boldsymbol{x}_b)\sqrt{\|\boldsymbol{M}\|_F}.
\end{aligned}$$

According to the definition of distance in (3.9), we have

$$\rho_M(\boldsymbol{x}, \boldsymbol{x}_k) = \sum_s d_{\boldsymbol{M}(A_s)}(\boldsymbol{x}, \boldsymbol{x}_k) + d_{\boldsymbol{M}(B)}(\boldsymbol{x}, \boldsymbol{x}_k);$$

and it follows Proposition 10 that,

$$\begin{aligned}
\text{lip}(\rho_{\boldsymbol{M},k} \leftarrow \boldsymbol{x}) &= \gamma_s \sqrt{\|\boldsymbol{M}(A_s)\|_F} + \sqrt{\|\boldsymbol{M}(B)\|_F} \\
&\leq \sum_s \sqrt{\|\boldsymbol{M}(A_s)\|_F} + \sqrt{\|\boldsymbol{M}(B)\|_F}
\end{aligned}$$

[5] The consistence between a matrix norm $\|\cdot\|_M$ and a vector norm $\|\cdot\|_v$ indicates $\|\boldsymbol{A}\boldsymbol{b}\|_v \leq \|\boldsymbol{A}\|_M \|\boldsymbol{b}\|_v$, where $\boldsymbol{A}$ is a matrix and $\boldsymbol{b}$ is a vector.

where $\rho_{\boldsymbol{M},k}$ denotes $\rho_M(\boldsymbol{x}, \boldsymbol{x}_k)$.

Based on the composition property illustrated in Proposition 11,

$$\begin{aligned}&\operatorname{lip}(\operatorname{sumKmin} \rho_{M,k} \leftarrow \boldsymbol{x}\}) \\ \leq& K\left\{\sum_s \sqrt{\|\boldsymbol{M}(A_s)\|_F} + \sqrt{\|\boldsymbol{M}(B)\|}_F\right\},\end{aligned}$$

where $\rho_{M,k}$ denotes the function of $\operatorname{sumKmin} \rho_{set}(\boldsymbol{x}, \boldsymbol{X}^-)$ or $\operatorname{sumKmin} \rho_{set}(\boldsymbol{x}, \boldsymbol{X}^+)$. Finally, based on Proposition 10, $f(\boldsymbol{x})$ in (3.10) is bounded by $2(\sum_s \sqrt{\|\boldsymbol{M}(A_s)\|_F} + \sqrt{\|\boldsymbol{M}(B)\|_F})$. □

Definition 24. [74] The *diameter* of a metric space $(\mathcal{X}, \rho)$ is defined as

$$\operatorname{diam}(\mathcal{X}, \rho) = \sup_{\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X}} \rho(\boldsymbol{x}_i, \boldsymbol{x}_j).$$

Definition 25. [18] For a metric space $(\mathcal{X}, \rho)$, let λ be the smallest number such that every ball in $\mathcal{X}$ can be covered by λ balls of half the radius. Then λ is called the *doubling constant* of $\mathcal{X}$ and the *doubling dimension* of $\mathcal{X}$ is $\operatorname{ddim}(\mathcal{X}) = \log_2 \lambda$.

As presented in [18], a low Euclidean dimension implies a low doubling dimension (Euclidean metrics of dimension d have doubling dimension $O(d)$); a low doubling dimension is more general than a low Euclidean dimension and can be utilized to measure the 'dimension' of a general metric space.

By combining the results of Proposition 7 and the Corollary 6 of [18], we can obtain the following Corollary.

Corollary 3. Let the metric space $(\mathcal{X}, \rho)$ have doubling dimension $\operatorname{ddim}(\mathcal{X})$ and let $\mathcal{F}$ be the collection of real-valued functions over $\mathcal{X}$ with the Lipschitz constant at most L. Then for any $f \in \mathcal{F}$, if f is correct on all but k training instances, we have with probability at least $1 - \delta$

$$\begin{aligned}&P\{\operatorname{sign}[f(\boldsymbol{x})] \neq t\} \\ \leq& \frac{k}{n} + \sqrt{\frac{2}{n}(c \log_2(34en/c) \log_2(578n) + \log_2(4/\delta))},\end{aligned} \tag{3.11}$$

Table 3.3: Calculation of partial derivatives $\frac{\partial \gamma}{\partial \boldsymbol{o}}$ and $\frac{\partial \gamma}{\partial r}$ in different cases.

Δ	λ_u, λ_v	γ	partial derivatives
$\Delta \leq 0$			$\frac{\partial \gamma}{\partial \boldsymbol{o}} = \boldsymbol{0}, \frac{\partial \gamma}{\partial r} = 0$
	$\lambda_u < 0, \lambda_v < 0$	0	
$\Delta > 0$	$\lambda_u < 0, \lambda_v > 1$	1	$\frac{\partial \gamma}{\partial \boldsymbol{o}} = \boldsymbol{0}, \frac{\partial \gamma}{\partial r} = 0$
	$\lambda_u > 1, \lambda_v > 1$	0	
$\Delta > 0$	$0 \leq \lambda_u \leq 1$	$\lambda_v - \lambda_u$	$\frac{\partial \gamma}{\partial \boldsymbol{o}} = 4\Delta^{-\frac{1}{2}}[\boldsymbol{x}_i + \frac{-b}{2a}(\boldsymbol{x}_j - \boldsymbol{x}_i) - \boldsymbol{o}]$
	$0 \leq \lambda_v \leq 1$		$\frac{\partial \gamma}{\partial r} = 4\Delta^{-\frac{1}{2}}r$
$\Delta > 0$	$\lambda_u < 0$	λ_v	$\frac{\partial \gamma}{\partial \boldsymbol{o}} = \frac{1}{a}\big[(\boldsymbol{x}_j - \boldsymbol{x}_i) - \Delta^{-\frac{1}{2}}\big(b(\boldsymbol{x}_j - \boldsymbol{x}_i) + 2a(\boldsymbol{o} - \boldsymbol{x}_i)\big)\big]$
	$0 \leq \lambda_v \leq 1$		$\frac{\partial \gamma}{\partial r} = 2\Delta^{-\frac{1}{2}}r$
$\Delta > 0$	$0 \leq \lambda_u \leq 1$	$1 - \lambda_u$	$\frac{\partial \gamma}{\partial \boldsymbol{o}} = \frac{1}{a}\big[(\boldsymbol{x}_i - \boldsymbol{x}_j) - \Delta^{-\frac{1}{2}}\big(b(\boldsymbol{x}_j - \boldsymbol{x}_i) + 2a(\boldsymbol{o} - \boldsymbol{x}_i)\big)\big]$
	$\lambda_v > 1$		$\frac{\partial \gamma}{\partial r} = 2\Delta^{-\frac{1}{2}}r$

where

$$c = \Big(16(\sum_s \sqrt{\|\boldsymbol{M}(A_s)\|_F} + \sqrt{\|\boldsymbol{M}(B)\|_F})\,\mathrm{diam}(\mathcal{X}, \rho)\Big)^{\mathrm{ddim}(\mathcal{X})+1}.$$

The above learning bound illustrates 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. Based on the bound, reducing the value of $\sum_s \sqrt{\|M(A_s)\|_F} + \sqrt{\|M(B)\|_F}$ would help reduce the gap between the empirical error and the expected error. For the reason that for each term $\sqrt{\|M(A_s)\|_F}$, reducing the value of $\|M(A_s)\|_F$ would reduce $\sqrt{\|M(A_s)\|_F}$. $\sum_s \sqrt{\|M(A_s)\|_F} + \sqrt{\|M(B)\|_F}$ would be used as the regularization term to improve the generalization ability of the classifier.

3.5 Optimisation

3.5.1 Objective Function

Based on the discussion in previous sections, in order to obtain low training error and good generalisation ability, the objective function of the optimisation problem would be the sum of hinge loss and the regularisation term $\sum_s \|\boldsymbol{M}(A_s)\|_F +$

$\|\boldsymbol{M}(B)\|_F$:

$$\begin{aligned} \min_{\Theta,\boldsymbol{\xi}} \quad & \tfrac{1}{N_1}\textstyle\sum_{(i,j)}\xi_{ij} + \tfrac{1}{N_2}\textstyle\sum_{(m,n)}\xi_{mn} + \alpha\|\boldsymbol{M}(B)\|_F + \alpha\textstyle\sum_s \|\boldsymbol{M}(A_s)\|_F \\ s.t. \quad & \rho_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j) \le 1 - C + \xi_{ij} \\ & \rho_{\boldsymbol{M}}(\boldsymbol{x}_m,\boldsymbol{x}_n) \ge 1 + C - \xi_{mn} \\ & \xi_{ij},\xi_{mn} \ge 0, \boldsymbol{M} \in \boldsymbol{M}_+ \\ & i,n = 1,\ldots,N, j \rightarrow i, m \nrightarrow n, \end{aligned} \tag{3.12}$$

where $\Theta = \{\boldsymbol{M}(A_s), \boldsymbol{M}(B), \boldsymbol{o}, \boldsymbol{r}\}$ denotes the set of parameters to be optimised; $j \rightarrow i$ indicates that $\boldsymbol{x}_j$ is $\boldsymbol{x}_i$'s K nearest neighbour comparing against all instances in the same class; $m \nrightarrow n$ indicates that $\boldsymbol{x}_m$ is $\boldsymbol{x}_n$'s K nearest neighbour comparing against all instances in the different class; and ξ_{ij} and ξ_{mn} indicates the errors. α is a trade-off parameters; and C is a constant which has the intuition of margin; $\boldsymbol{M}_+$ denotes the set of positive semi-definite matrices.

The parameters to be optimised include local metrics $\boldsymbol{M}(A_s)$, background metric $\boldsymbol{M}(B)$, centers of influential regions $\boldsymbol{o}_s$ and radius of influential regions r_s. Thus in the proposed algorithm, the locations of influential regions ($\boldsymbol{o}_s, r_s$) and the metrics of influential/background regions ($\boldsymbol{M}(B), \boldsymbol{M}(A_s)$) would be learned under the same framework.

3.5.2 Gradient Descent

With $\rho_{\boldsymbol{M}(A_s)}$ and $\rho_{\boldsymbol{M}(B)}$ being the Mahalanobis distances, the optimisation problem is not a convex problem even when $\boldsymbol{o}, \boldsymbol{r}$ is fixed and $\boldsymbol{M}(A_s)$ and $\boldsymbol{M}(B)$ are updated. Thus the gradient descent algorithm is used:

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

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

The objective function g is

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

where the distance is

$$\rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = [1 - \sum_s \gamma_s(\boldsymbol{o}_s, r_s)]_+ \rho_{\boldsymbol{M}(B)}(\boldsymbol{x}_i, \boldsymbol{x}_j) + \sum_s \gamma_s(\boldsymbol{o}_s, r_s) \rho_{\boldsymbol{M}(A_s)}(\boldsymbol{x}_i, \boldsymbol{x}_j).$$

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; $[x]_+ = \max(x, 0)$.

The gradient with respect to each set of parameters is

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

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

Now $\frac{\partial \rho_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)}{\partial \Theta}|_{\Theta^t}$ will be calculated separately for the parameters $\boldsymbol{M}(A)$, $\boldsymbol{M}(B)$, $\boldsymbol{o}^s$, $\boldsymbol{r}^s$:

$$\begin{aligned} \frac{\partial \rho(\boldsymbol{x}_i, \boldsymbol{x}_j)}{\partial \boldsymbol{M}(B)}|_{\Theta^t} =& \mathbb{1}[\gamma_b(\boldsymbol{o}_s^t, r_s^t) > 0] \gamma_b(\boldsymbol{o}_s^t, r_s^t)[(\boldsymbol{x}_i - \boldsymbol{x}_j)^T M^t(B)(\boldsymbol{x}_i - \boldsymbol{x}_j)]^{-1/2} \\ & (\boldsymbol{x}_i - \boldsymbol{x}_j)(\boldsymbol{x}_i - \boldsymbol{x}_j)^T, \end{aligned}$$

where $\gamma_b(\boldsymbol{o}_s^t, r_s^t) = 1 - \sum_s \gamma_s(\boldsymbol{o}_s^t, r_s^t)$;

$$\frac{\partial \rho(\boldsymbol{x}_i, \boldsymbol{x}_j)}{\partial \boldsymbol{M}(A_s)}|_{\Theta^t} = \gamma_s(\boldsymbol{o}_s^t, r_s^t)[(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{M}^t(A_s)(\boldsymbol{x}_i - \boldsymbol{x}_j)]^{-1/2}(\boldsymbol{x}_i - \boldsymbol{x}_j)(\boldsymbol{x}_i - \boldsymbol{x}_j)^T;$$

$$\frac{\partial \rho(\boldsymbol{x}_i, \boldsymbol{x}_j)}{\partial \boldsymbol{o}_s}|_{\Theta^t} = \rho_{\boldsymbol{M}^t(A_s)}(\boldsymbol{x}_i, \boldsymbol{x}_j) \frac{\partial \gamma_s}{\partial \boldsymbol{o}_s} - \mathbb{1}[1 - \sum_s \gamma_s(\boldsymbol{o}_s^t, r_s^t) > 0] \rho_{\boldsymbol{M}^t(B)} \frac{\partial \gamma_s}{\partial \boldsymbol{o}_s},$$

where $\frac{\partial \gamma}{\partial \boldsymbol{o}}$ could be obtained as illustrated in Table 3.3;

$$\frac{\partial \rho(\boldsymbol{x}_i, \boldsymbol{x}_j)}{\partial r_s}|_{\Theta^t} = \rho_{\boldsymbol{M}^t(A_s)}(\boldsymbol{x}_i, \boldsymbol{x}_j)\frac{\partial \gamma_s}{\partial r_s} - \mathbb{1}[1 - \sum_s \gamma_s(\boldsymbol{o}_s^t, r_s^t) > 0]\rho_{\boldsymbol{M}^t(B)}\frac{\partial \gamma_s}{\partial r_s},$$

where $\frac{\partial \gamma}{\partial r}$ could be obtained as illustrated in Table 3.3.

In this way, all of the gradients with respect to each set of parameters could be obtained and gradient descent could be used to solve the optimisation problem.

Initial values are very important for non-convex optimisation problems. A heuristic method is adopted to initialise the parameters as follows. 1) Extract local discriminative direction $e(\boldsymbol{x}) \in \mathbb{R}^D$ for each training instance $\boldsymbol{x}$, where D indicates the number of features of $\boldsymbol{x}$:

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

where $\boldsymbol{x}[d]$ indicates the dth dimension of vector $\boldsymbol{x}$; $j \rightarrow i$ indicates $\boldsymbol{x}_j$ is $\boldsymbol{x}_i$'s K nearest neighbour comparing against all instances in the same class; $k \nrightarrow i$ indicates $\boldsymbol{x}_k$ is $\boldsymbol{x}_i$'s K nearest neighbour comparing against all instances in the different class. 2) Cluster with augmented features: $[\boldsymbol{x}, e(\boldsymbol{x})]$ are used as features and the K-means clustering algorithm with random initial points is adopted to divide all instances into K clusters. 3) Initialise the parameters: Cluster centres are initialised as $\boldsymbol{o}_s$; the distance between 80 percentiles and the cluster centre is set as initial value of r_s; the local metric is set as $\boldsymbol{M}(A_s) = \boldsymbol{I} + 0.1 \times \mathrm{diag}(\mathrm{mean}(e(\boldsymbol{x}), \boldsymbol{x} \in \text{cluster } s))$, where diag is an operation which returns a square diagonal matrix with elements of the input vector on the main diagonal.

3.6 Experiments

The proposed algorithm is compared with nine established metric learning algorithms from two categories: 1) The most cited algorithms, including large Margin nearest neighbor (LMNN) [76], information theoretic metric learning (ITML) [9], neighborhood Component Analysis (NCA) [17] and metric learning by collapsing classes (MCML) [16]; (2) the most state-of-the-art algorithms, including ge-

Table 3.4: Characteristics of the data sets. The total number of instances (and the numbers of instances in each class in brackets) and the number of features.

	Instances	Features
Australian	690 (383, 307)	14
Breastcancer	683 (444, 239)	10
Diabetes	768 (268, 500)	8
Fourclass	862(555, 307)	2
German	1000 (700, 300)	24
Haberman	206(81, 125)	3
Heart	270 (150, 120)	13
ILPD	583(167, 416)	10
Liverdisorders	345(145, 200)	6
Pima	768(268, 500)	8
Vote	435 (168, 267)	16
WDBC	569 (357, 212)	30

ometric mean metric learning (GMML) [83], regressive virtual metric learning (RVML) [52], stochastic neighbor compression (SNC) [35], sparse compositional metric learning (SCML) [59] and reduced-rank local distance metric learning (R2LML) [26]. LMNN and ITML are implemented with metric-learn toolbox[6]; NCA and MCML are implemented with the drToolbox[7]; and GMML, RVML, SCML, R2LML and SNC are implemented by using the authors' code.

The experiment is focused on binary classification of 12 publicly available data sets from the websites of UCI[8] and LibSVM[9], namely Australian, Breastcancer, Diabetes, Fourclass, Germannumber, Haberman, Heart, ILPD, Liverdisorders, Pima, Voting and WDBC. All data sets are pre-processed by firstly subtracting the mean and dividing by the standard deviation, and then normalising 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. 10-fold cross-validation is

[6] https://all-umass.github.io/metric-learn/

[7] https://lvdmaaten.github.io/drtoolbox/

[8] https://archive.ics.uci.edu/ml/datasets.html

[9] https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/binary.html

Table 3.5: Experiment results of local metric learning. Mean accuracy (percentage) and standard deviation are reported with the best ones in bold; '# of best' indicates the number of data sets that an algorithm performs the best.

Dataset	LMNN	ITML	MCML	NCA	RVML
Australian	78.8 ±2.57	77.17 ±1.94	78.77 ±1.70	79.96 ±1.63	83.01 ±1.58
Breastcancer	95.91 ±0.69	96.39 ±1.04	96.35 ±0.77	95 ±1.52	95.77 ±1.09
Diabetes	69.16 ±1.44	69.09 ±1.24	69.19 ±1.18	68.47 ±2.46	71.04 ±2.60
Fourclass	72.06 ±2.31	72.09 ±2.22	72.06 ±2.43	72.06 ±2.46	70.46 ±1.40
German	67.85 ±1.54	66.95 ±2.05	67.67 ±1.48	69.95 ±2.88	71.65 ±1.78
Haberman	67.89 ±3.34	67.97 ±4.05	67.56 ±2.75	67.4 ±3.33	66.67 ±2.30
Heart	76.2 ±3.82	76.94 ±3.30	77.22 ±3.66	75.56 ±2.01	77.69 ±4.05
ILPD	66.97 ±2.13	68.67 ±2.83	67.48 ±2.58	66.8 ±1.19	67.95 ±2.90
Liverdisorders	61.01 ±4.80	57.17 ±4.01	60.65 ±5.12	59.78 ±3.44	64.64 ±3.93
Pima	68.54 ±1.64	67.95 ±2.01	68.31 ±2.33	65.91 ±3.04	69.45 ±1.68
Voting	94.83 ±0.77	90.75 ±1.44	92.64 ±1.58	94.77 ±0.92	95.75 ±1.26
WDBC	96.58 ±1.12	94.91 ±0.92	95.7 ±0.90	96.58 ±0.85	96.58 ±1.34
# of best	0	0	0	0	0

Dataset	GMML	SCML	R2LML	SNC	local
Australian	84.35 ±1.04	82.25 ±1.40	84.67 ±1.32	81.78 ±8.8	**84.78 ±1.93**
Breastcancer	**97.26 ±0.81**	97.01 ±0.91	97.01 ±0.66	96.65 ±0.69	97.15 ±1.32
Diabetes	74.16 ±2.58	71.49 ±2.21	73.8 ±1.37	**75.32 ±2.74**	75.19 ±3.59
Fourclass	76.12 ±1.87	75.54 ±1.42	76.12 ±1.91	73.39 ±8.7	**79.71 ±1.73**
German	71.55 ±1.12	70.9 ±2.65	**72.9 ±1.83**	70.13 ±3.33	72.45 ±2.2
Haberman	71.22 ±3.35	69.19 ±2.47	71.06 ±3.39	71.98 ±5.2	**74.06 ±3.25**
Heart	81.2 ±2.69	78.98 ±3.24	**82.04 ±3.81**	77.04 ±5.32	81.66 ±3.09
ILPD	67.14 ±2.17	68.03 ±2.90	65.85 ±2.22	68.91 ±2.67	**69.27 ±2.58**
Liverdisorders	63.84 ±5.43	61.74 ±4.57	**66.81 ±3.68**	63.31 ±5.18	65.28 ±3.99
Pima	72.95 ±1.84	71.14 ±2.64	72.34 ±1.54	73.99 ±2.59	**74.31 ±2.68**
Voting	95.17 ±1.88	95 ±1.30	**96.32 ±1.19**	94.45 ±1.2	95.74 ±1.48
WDBC	96.71 ±0.78	96.97 ±0.89	96.93 ±1.67	96.93 ±0.85	**97.28 ±1.37**
# of best	1	0	4	1	6

used to select the trade-off parameters in the compared algorithms, namely the regularisation 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 $\{0.1, 0.3, 0.5, 0.7, 0.9\}$), λ in RVML (from $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$), Ratio in SNC (from $\{0.01, 0.02, 0.04, 0.08, 0.16\}$). All other parameters are set as default. For the proposed algorithm, the parameters are set as follows: α and C in the optimisation formula are 0.1 and 0.5 respectively; K in the classifier is 10; and the number of clusters when initialising the parameters is 4.

As shown in Table 5.1, the proposed algorithm achieves the best accuracy on 6 data sets out of the 12 data sets. None of the other algorithms performs the best in more than 4 data sets. In cases where our algorithm is not leading, it performs quite nice and stays close to the best one. Such encouraging results demonstrate the effectiveness of our proposed method.

3.7 Conclusion

In this chapter, a very intuitive distance is defined through the introduction of influential regions and the background region. The distance can be computed efficiently and encouraging results are obtained on publicly available data sets. It is straightforward to extend the proposed algorithm to the multi-class case and use more advanced optimisation techniques. Other metrics or types of influential regions can also be adopted for specific tasks. Domain knowledge can facilitate the partition of regions.

Chapter 4

Metric Learning with Instance Extraction

4.1 Introduction

As we know, the nearest neighbour (NN) classifier is one of the oldest and simplest methods for classification, which compares the distances between a new instance and training instances and assigns the new instance to the class of its nearest training instance. Although it has been used as a benchmark tool, NN suffers from the following two problems. First, the performance of NN is highly affected by the distance metric used in the algorithm. Due to the difficulty in handcrafting a well-suited and adaptive distance metric, metric learning has been proposed to enable the algorithms to automatically learn a metric from available data. Metric learning with a convex objective function was first proposed in the pioneering work of Xing et al. [78]. After that, many other metric learning methods have been developed and widely adopted, such as the large margin nearest neighbour (LMNN) [75] and the information theoretic metric learning [9]. Some theoretical work has also been proposed for metric learning, in particular on deriving different kinds of learning bounds [33, 4, 20, 8, 67].

Second, NN suffers from storage and computation problems. In order to classify a test instance, NN has to store all training instances and calculate its distances to all training instances. The high time and space complexity makes computing the decision rule impracticable for resource-constraint or real-time applications. Some work has been conducted on NN compression, such as [21, 35, 19, 73].

In this chapter, we propose a metric learning with instance extraction (MLIE) classifier to solve the above two problems in NN. First, to solve the storage and computation problem of NN, MLIE extracts several training instances from each class and then calculates the distances between the test instance and the extracted training instances. When the number of extracted training instances is much less than the total number of training instances, the storage and computation costs can be largely reduced by MLIE. Second, MLIE learns a tailored distance metric from the training data automatically and would be suitable for specific data sets.

Moreover, we also illustrate the intuitive and theoretical properties of MLIE in this chapter. First, to make deep insight into the classification mechanism of MLIE, we discuss the relationship between the proposed MLIE and the local linear classifier (LLC). LLC divides the feature space to several local regions and uses a linear classifier within each region. Here we show that MLIE is a special case of the local linear classifier, which can simultaneously learn the local regions and their associated linear classifiers. Second, the PAC learning bound of MLIE has been discussed. The regularisation term used in the learning algorithm of MLIE is proposed based on the learning bound of MLIE, which guarantees the test performance of MLIE.

The proposed algorithm has been tested on 12 benchmark real data sets and the experiment results demonstrate the superior performance of MLIE to state-of-the-art metric learning algorithms.

4.2 Metric Learning with Instance Extraction

In this section, we first define the notations used in this chapter. We then propose the metric learning with instance extraction (MLIE) classifier to solve the two problems in NN. MLIE can learn the tailored distance metric from data automatically and can also save the storage and computational costs by extracting informative training instances and calculating fewer distances than NN. Lastly, we discuss the relationship between MLIE and the local linear classifier (LLC) for better understanding the classification mechanism of MLIE.

4.2.1 Notations

In this chapter, we focus on binary classification. Let $\boldsymbol{x}^n = \{\boldsymbol{x}_i, i = 1, \ldots, n\}$ denote the set of training instances, where $\boldsymbol{x}_i \in \mathcal{X} \subseteq \mathbb{R}^D$ and n denotes the number of training instances. Let $\boldsymbol{y}^n = \{y_i, i = 1, \ldots, n\}$ denote the corresponding labels, where $y_i \in \{-1, +1\}$. $y_i = +1$ and $y_i = -1$ indicate that $\boldsymbol{x}_i$ belongs to the positive and negative class, respectively. The number of positive and negative class training instances are defined as n_+ and n_-, respectively. Let $\boldsymbol{x}_+^{n+} = \{\boldsymbol{x}_i^+, i = 1, \ldots, n_+\}$ and $\boldsymbol{x}_-^{n-} = \{\boldsymbol{x}_i^-, i = 1, \ldots, n_-\}$ denote the set of positive and negative training instances, respectively. Let $\overline{\boldsymbol{x}_i\boldsymbol{x}_j}$ denote the line segment connecting $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ and $\boldsymbol{x}_i\boldsymbol{x}_j$ denote the line passing through $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ when $\boldsymbol{x}_i \neq \boldsymbol{x}_j$.

Let $\boldsymbol{r}_-^{m-} = \{\boldsymbol{r}_i^-, i = 1, \ldots, m_-\}$ and $\boldsymbol{r}_+^{m+} = \{\boldsymbol{r}_j^+, j = 1, \ldots, m_+\}$ denote the set of extracted positive and negative instances, respectively, where m_+ and m_- denote the number of extracted positive and negative instances, respectively. Let $\boldsymbol{r}^m = \{\boldsymbol{r}_i^-, \boldsymbol{r}_j^+; i = 1, \ldots, m_+, j = 1, \ldots, m_-\}$ denote the set of all extracted instances.

Let $d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T(\boldsymbol{x}_i - \boldsymbol{x}_j)}$ denote the Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. Let $d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T\boldsymbol{M}(\boldsymbol{x}_i - \boldsymbol{x}_j)}$ denote the Mahalanobis distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, where $\boldsymbol{M} \in \boldsymbol{M}_+$ is the parameter matrix of $d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $\boldsymbol{M}_+$ denote the set of positive semi-definite matrices. In this chapter, we aim to learn a Mahalanobis distance metric, which is the mostly adopted distance metric in metric learning.

4.2.2 Metric Learning with Instance Extraction (MLIE)

NN classifies a test instance to the class of its nearest training instance. Therefore, NN has to calculate the distances between the test instance and all the training instances. When the number of training instances is large, NN suffers from the storage and computational problems. Therefore, NN is not suitable for situations when the computational resource is limited and also for real-time applications when the computation time is restricted.

To solve this problem, several informative training instances may be extracted and we only need calculate the distances between the test instance and the extracted training instances. In this way, the computational cost can be reduced. The extracted training instances are automatically learned from the available data. The test instance is then classified to the class of its nearest extracted training instance. The distance metric used in this method is also automatically learned from data. This kind of classifiers is named metric learning with instance extraction (MLIE) and one example is defined as follows[1]:

$$\begin{aligned} h(\boldsymbol{x};\boldsymbol{r}^m,\boldsymbol{M}) &= \min_i d^2_{\boldsymbol{M}}(\boldsymbol{x},\boldsymbol{r}_i^-) - \min_j d^2_{\boldsymbol{M}}(\boldsymbol{x},\boldsymbol{r}_j^+) \\ &= \min_i (\boldsymbol{x}-\boldsymbol{r}_i^-)^T\boldsymbol{M}(\boldsymbol{x}-\boldsymbol{r}_i^-) - \min_j (\boldsymbol{x}-\boldsymbol{r}_j^+)^T\boldsymbol{M}(\boldsymbol{x}-\boldsymbol{r}_j^+), \end{aligned} \tag{4.1}$$

where $\boldsymbol{r}_i^- \in \boldsymbol{r}_-^{m-}$ denotes the ith extracted negative instance in the same space of $\mathcal{X}$, $\boldsymbol{r}_i^+ \in \boldsymbol{r}_+^{m+}$ denotes the jth extracted positive instance in the same space of $\mathcal{X}$ and $\boldsymbol{M}$ is the parameter matrix of the distance metric. $\min_i d^2_{\boldsymbol{M}}(\boldsymbol{x},\boldsymbol{r}_i^-)$ and $\min_j d^2_{\boldsymbol{M}}(\boldsymbol{x},\boldsymbol{r}_j^+)$ indicate the distances between $\boldsymbol{x}$ and the negative and positive classes, respectively, which are represented by the minimum distances between $\boldsymbol{x}$ and the associated extracted training instances. Thus the test instance $\boldsymbol{x}$ is classified to the positive class when $h(\boldsymbol{x};\boldsymbol{r}^m,\boldsymbol{M}) \geq 0$ and to the negative class when $h(\boldsymbol{x};\boldsymbol{r}^m,\boldsymbol{M}) < 0$.

Note that learning the parameter matrix $\boldsymbol{M}$ in (4.1) is equivalent to learning a linear mapping $\boldsymbol{L}$ for the instances. This is because (4.1) can be written as

$$\begin{aligned} h(\boldsymbol{x};\boldsymbol{r}^m,\boldsymbol{L}) &= \min_i d^2(\boldsymbol{L}\boldsymbol{x},\boldsymbol{L}\boldsymbol{r}_i^-) - \min_j d^2(\boldsymbol{L}\boldsymbol{x},\boldsymbol{L}\boldsymbol{r}_j^+) \\ &= \min_i (\boldsymbol{x}-\boldsymbol{r}_i^-)^T\boldsymbol{L}^T\boldsymbol{L}(\boldsymbol{x}-\boldsymbol{r}_i^-) - \min_j (\boldsymbol{x}-\boldsymbol{r}_j^+)^T\boldsymbol{L}^T\boldsymbol{L}(\boldsymbol{x}-\boldsymbol{r}_j^+), \end{aligned} \tag{4.2}$$

where $\boldsymbol{L} \in \mathbb{R}^{D\times D}$ denotes a linear mapping and $\boldsymbol{L}^T\boldsymbol{L} = \boldsymbol{M}$.

From (4.2), we can also learn the extracted instance in the mapped space di-

[1] In this chapter, all discussions are limited to learn a squared Mahalanobis distance.

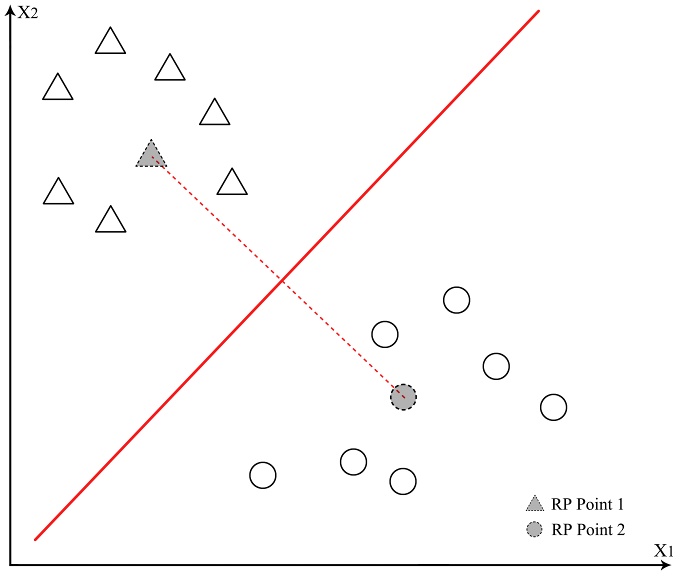

Figure 4.1: An illustrative example of nearest neighbour with instance extraction (extract one instance per class). It is equivalent to learn one representative points (RP) per class and find a linear classifier based on the two representative points.

rectly as follows:

$$\begin{aligned} h(\boldsymbol{x}; \boldsymbol{r}^m, \boldsymbol{L}) &= \min_i d^2(\boldsymbol{L}\boldsymbol{x}, \boldsymbol{r}_i^-) - \min_j d^2(\boldsymbol{L}\boldsymbol{x}, \boldsymbol{r}_j^+) \\ &= \min_i (\boldsymbol{L}\boldsymbol{x} - \boldsymbol{r}_i^-)^T (\boldsymbol{L}\boldsymbol{x} - \boldsymbol{r}_i^-) - \min_j (\boldsymbol{L}\boldsymbol{x} - \boldsymbol{r}_j^+)^T (\boldsymbol{L}\boldsymbol{x} - \boldsymbol{r}_j^+), \end{aligned} \tag{4.3}$$

where $\boldsymbol{r}_i^- \in \boldsymbol{r}_-^{m+}$ and $\boldsymbol{r}_j^+ \in \boldsymbol{r}_+^{m+}$ denote the ith extracted negative instance and the jth extracted positive instance in the mapped space after liner mapping $\boldsymbol{L}$, respectively.

In MLIE, the extracted instances $\boldsymbol{r}^m$ and the parameter matrix $\boldsymbol{M}$ of the distance metric are automatically learned from data. We will introduce the learning algorithm in Section 4.4, after showing the learning bound of MLIE in Section 4.3. In the rest of this section, we will focus on discussing the classification mechanism of MLIE, by showing its relationship with the local linear classifier (LLC).

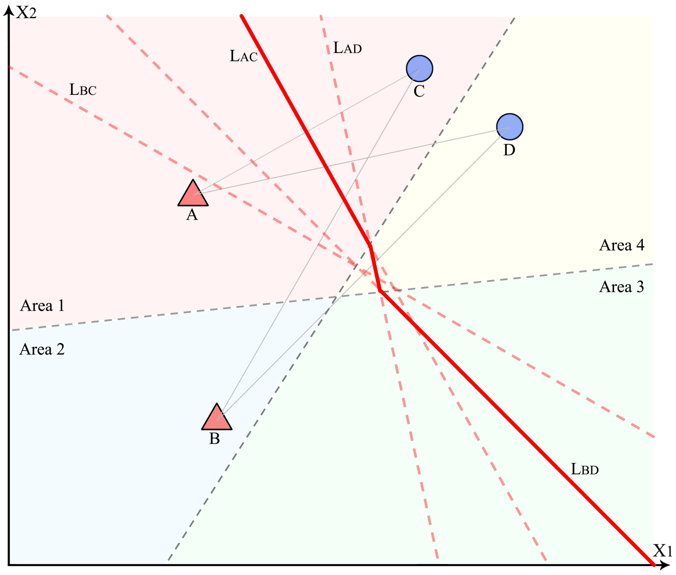

Figure 4.2: An illustrative example of nearest neighbour with instance extraction (extract two instances per class). A and B are the extracted instances for the triangle class and C and D are the extracted instances for the circle class. We can then find the equivalent Local Linear Classifier (LLC) based on A, B, C and D. The red dashed lines indicates the inter-class perpendicular line. For example, L_{AC} indicates the perpendicular line for the line segment AC, where A comes from triangle class and C comes from circle class. The black dashed lines indicates the perpendicular line for extracted instances from the same class. The intra-class perpendicular lines (for points A,B and points C,D) segment the coordinate system into four local regions. Inside each local region, the inter-class perpendicular line acts as the linear classifier. For example, for points falling into region 1, since they are closest to points A and C, the perpendicular line for points AC, i.e. L_{AC}, acts as the classification boundary.

4.2.3 Classification Mechanism of MLIE

In this section, we discuss the classification mechanism of MLIE by showing its relationship with local linear classifier (LLC). We start with a simple and special case of MLIE for illustrative purpose: MLIE with the Euclidean distance metric and extracting two instances per class. Then we discuss the classification mechanism of the general case of MLIE and show its relationship with LLC.

4.2.3.1 Local Linear Classifier

Local linear strategies has been adopted in machine learning society to solve the classification problems [38, 36, 10]. In this thesis, local linear classifier is defined as follows.

Definition 26. A classifier is called a *Local Linear Classifier* (LLC) if (1) it could be represented by a continuous function and (2) there exists a set of local regions C

$$C = \{C_s, s = 1, \ldots, S\} \quad \text{and} \quad \bigcup_s C_s = \mathcal{X},$$

such that the classification boundary inside each local region is specified by a linear function.

Besides linear inside each local regions, in the above definition, LLC should be continuous at the boundary. Different partitions of the local regions and different local linear classifiers make LLC a powerful tool to fit data. At the same time, the local linear property, as well as the continuity property at the boundary, constrain the complexity of LLC and hence make it learnable in certain cases.

4.2.3.2 Nearest Neighbour with Instance Extraction

We first provide a special case of MLIE to illustrate the mechanism of MLIE and its relationship with LLC: MLIE with the Euclidean distance metric and extracting one instance per class. We call this classifier as the nearest neighbour with instance extraction (NNIE) and define it as follows:

$$h(\boldsymbol{x}; \boldsymbol{r}^m) = \min_i d^2(\boldsymbol{x}, \boldsymbol{r}_i^-) - \min_j d^2(\boldsymbol{x}, \boldsymbol{r}_j^+).$$

In the case of extracting one instance per class, NNIE is a linear classifier, as illustrated in Figure 4.1. The grey triangle and the grey circle are the two extracted training instances from the two classes, respectively. The decision boundary is the red solid line.

In the case of extracting multiple instances per class, NNIE is a LLC, as illustrated in Figure 4.1. With two extracted instances for each class, the two bisectors between intra-class line segment can divide the space into four regions. Then based on the bisectors of the related inter-class line segment, the instances inside each class can be classified. Because the classification function is continuous, the classification boundary is continuous and the overall classification boundary is a piece-wise linear function, which would be illustrated intuitively in two dimension case. The relationship between NNIE and LLC is illustrated in Proposition 12.

Proposition 12. NNIE with the following classifier is an LLC.

$$h(\boldsymbol{x}; \boldsymbol{r}^m) = \min_a d^2(\boldsymbol{x}, \boldsymbol{r}_a^-) - \min_b d^2(\boldsymbol{x}, \boldsymbol{r}_b^+).$$

Proof. Based on the definition of the classifier, $\mathcal{X}$ can be divided into (at most) $n_- n_+$ local regions $C_{(i,j)} = \{\boldsymbol{x} | i = \operatorname{argmin}_a d(\boldsymbol{x}, \boldsymbol{r}_a^-), a = 1, \ldots, n_-;\ j = \operatorname{argmin}_b d(\boldsymbol{x}, \boldsymbol{r}_b^+), b = 1, \ldots, n_+\}$. For $\boldsymbol{x} \in C_{(i,j)}$, the nearest neighbour inside the negative class is i and the nearest neighbour inside the positive class is j. The bisector hyperplane of line segment $\overline{\boldsymbol{r}_i \boldsymbol{r}_j}$ is the classification boundary inside the local region $C_{(i,j)}$ because the local classification function is $h(\boldsymbol{x}) = d^2(\boldsymbol{x}, \boldsymbol{r}_i^-) - d^2(\boldsymbol{x}, \boldsymbol{r}_j^+) = 2(\boldsymbol{r}_j^+ - \boldsymbol{r}_i^-)^T \boldsymbol{x} + (\boldsymbol{r}_i^{-T} \boldsymbol{r}_i^- - \boldsymbol{r}_j^{+T} \boldsymbol{r}_j^+)$, which is a linear classifier. Meanwhile, the classifier is a continuous function. Therefore, it is an LLC. □

4.2.3.3 Metric Learning with Instance Extraction (MLIE)

Here we show the general case of MLIE: MLIE with extracting multiple instances per class and the Mahalanobis distance metric is an LLC. Thus MLIE can be understood as a way to simultaneously learn a partition of the local regions and the local classifier.

Proposition 13. MLIE with the following classifier is an LLC,

$$h(\boldsymbol{x}; \boldsymbol{r}^m, \boldsymbol{M}) = \min_a d^2_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{r}^-_a) - \min_b d^2_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{r}^+_b).$$

Proof. Based on the definition of the classifier, $\mathcal{X}$ can be divided into (at most) $n_- n_+$ local regions $C_{(i,j)} = \{\boldsymbol{x} | i = \operatorname{argmin}_a d_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{r}^-_a), a = 1, \ldots, n_-;\ j = \operatorname{argmin}_b d_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{r}^+_b), b = 1, \ldots, n_+\}$. For $\boldsymbol{x} \in C_{(i,j)}$, the nearest neighbour inside the negative class is i and the nearest neighbour inside the positive class is j. A linear classifier would classifier the instances inside $C_{(i,j)}$ because the local classification function is $h(\boldsymbol{x}) = d^2_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{r}^-_i) - d^2_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{r}^+_j) = 2(\boldsymbol{r}^+_j - \boldsymbol{r}^-_i)^T \boldsymbol{M}\boldsymbol{x} + (\boldsymbol{r}^{-T}_i \boldsymbol{M} \boldsymbol{r}^-_i - \boldsymbol{r}^{+T}_j \boldsymbol{M} \boldsymbol{r}^+_j)$ and it is a linear classifier. Meanwhile, the classifier is a continuous function. Therefore, it is an LLC. □

Proposition 14. MLIE with the following classifier is an LLC,

$$h(\boldsymbol{x}; \boldsymbol{r}^m, \boldsymbol{L}) = \min_a d^2(\boldsymbol{L}\boldsymbol{x}, \boldsymbol{r}^-_a) - \min_b d^2(\boldsymbol{L}\boldsymbol{x}, \boldsymbol{r}^+_b).$$

Proof. Based on the definition of the classifier, $\mathcal{X}$ can be divided into (at most) $n_- n_+$ local regions $C_{(i,j)} = \{\boldsymbol{x} | i = \operatorname{argmin}_a d(\boldsymbol{L}\boldsymbol{x}, \boldsymbol{r}^-_a), a = 1, \ldots, n_-;\ j = \operatorname{argmin}_b d(\boldsymbol{L}\boldsymbol{x}, \boldsymbol{r}^+_b), b = 1, \ldots, n_+\}$. For $\boldsymbol{x} \in C_{(i,j)}$, the nearest neighbour inside the negative class is i and the nearest neighbour inside the positive class is j. A linear classifier would classifier the instances inside $C_{(i,j)}$ because the local classification function is $h(\boldsymbol{x}) = d^2(\boldsymbol{L}\boldsymbol{x}, \boldsymbol{r}^-_i) - d^2(\boldsymbol{L}\boldsymbol{x}, \boldsymbol{r}^+_j) = 2(\boldsymbol{r}^+_j - \boldsymbol{r}^-_i)^T \boldsymbol{L}\boldsymbol{x} + (\boldsymbol{r}^{-T}_i \boldsymbol{r}^-_i - \boldsymbol{r}^{+T}_j \boldsymbol{r}^+_j)$ and it is a linear classifier. Meanwhile, the classifier is a continuous function. Therefore, it is an LLC. □

As we have discussed before, learning the parameter $\boldsymbol{M}$ of Mahalanobis/squared Mahalanobis distance is the same as learning a linear mapping $\boldsymbol{L}$, where $\boldsymbol{L}^T\boldsymbol{L} = \boldsymbol{M}$. Then $\boldsymbol{x}$ is mapped into another space $\boldsymbol{x}' = \boldsymbol{L}\boldsymbol{x}$ and Euclidean distance is used in the new space. The learned $\boldsymbol{M}$ (or equivalent $\boldsymbol{L}$) will map the data into another space which would be easier for the data to be classified. At the same time, the learned extracted instances would determine a partition of

Table 4.1: Relationship between metric learning with instance extraction (MLIE) and the related classifiers. LC denotes linear classifier, LM denotes linear mapping, LLC denotes Local linear classifier.

Instance Extraction	Metric Learning	Equivalent
Extract one instance per class	none	LC
$m_+ = m_- = 1$	M of Mahalanobis	LM + LC
Extract more than one instance	none	LLC
m_+ or $m_- \geq 2$	M of Mahalanobis	LM+LLC

local regions and the local classifiers inside each region. Table 4.1 summarises the relationships between the discussed algorithms.

4.3 Learnability of MLIE

4.3.1 Notations and Assumptions

Let $h(\boldsymbol{x}, \boldsymbol{\theta})$ denote a function with input $\boldsymbol{x}$ and parameter $\boldsymbol{\theta}$. The output of $h(\boldsymbol{x}, \boldsymbol{\theta})$ is a real value for binary classification. Let $\boldsymbol{x} \in \mathcal{X} \subseteq R^D$, where $\mathcal{X}$ denotes a set which contains all possible values of $\boldsymbol{x}$. Let $\boldsymbol{\theta} = (\theta_{[1]}, \ldots, \theta_{[Q]}) \in \Theta \subseteq R^Q$ be the parameter vector of the classifier, where Θ denotes a set which contains all possible values of $\boldsymbol{\theta}$. $y = \text{sign}\left[h(\boldsymbol{x}, \boldsymbol{\theta})\right]$ returns the classification result of input $\boldsymbol{x}$ given parameter $\boldsymbol{w}$, where $y \in \mathcal{Y} = \{-1, 1\}$ and $\text{sign}[\cdot]$ is the sign function: $\text{sign}[a] = 1$ if $a \geq 0$ and $\text{sign}[a] = -1$ if $a < 0$, where $a \in \mathbb{R}$.

Suppose the input $\boldsymbol{x}$ is a random variable distributed according to an *unknown* distribution with probability density function (PDF) $f(\boldsymbol{x})$. Let $\boldsymbol{x}^n = \{\boldsymbol{x}_i, i = 1, \ldots, n\}$ denote a set of n independent and identically (i.i.d.) distributed instances sampled from $f(\boldsymbol{x})$. Let $\boldsymbol{y}^n = \{y_i, i = 1, \ldots, n\}$ denote the label set, where y_i denotes the corresponding label of $\boldsymbol{x}_i$. $P(y|\boldsymbol{x})$ follows an *unknown* underlying distribution. Let $\boldsymbol{z}^n = \{\ (\boldsymbol{x}_i, y_i), i = 1, \ldots, n\}$ denote the set of training instance and label pairs, based on the assumptions of $f(\boldsymbol{x})$ and $P(y|\boldsymbol{x})$, $\boldsymbol{z}^n$ are n i.i.d. training pairs sampled from $p(\boldsymbol{z}) = p(\boldsymbol{x}, y)$ and $p(\boldsymbol{z})$ is *unknown*.

$R_n(\boldsymbol{z}^n, h_{\boldsymbol{\theta}}) := \frac{1}{n}\sum_i r(\boldsymbol{z}_i, h_{\boldsymbol{\theta}}) := \frac{1}{n}\sum_i l(h(\boldsymbol{x}_i, \boldsymbol{\theta}); y_i)$ is called the *training error* or *empirical risk* and it indicates the training loss given parameter $\boldsymbol{\theta}$ for training instances $\boldsymbol{x}^n$, where r and l denote the risk function and the loss function respectively. $R(h_{\boldsymbol{\theta}}) := \mathbb{E}_{\boldsymbol{z}'} r(\boldsymbol{z}', h_{\boldsymbol{\theta}}) := \mathbb{E}_{\boldsymbol{z}'} l(h(\boldsymbol{x}', \boldsymbol{\theta}); y')$ is called the *test error* or *expected risk* and it indicates the expected value of test loss given a test input pair $\boldsymbol{z}' = (\boldsymbol{x}', y')$ and the parameter $\boldsymbol{\theta}$. The gap between the training error and test error, i.e. $R(h_{\boldsymbol{\theta}}) - R_n(\boldsymbol{z}^n, h_{\boldsymbol{\theta}})$, is called the *generalisation gap*.

Let $\|\boldsymbol{v}\|$ denote the L_2-norm of a vector $\boldsymbol{v}$. Let $\|\boldsymbol{M}\|$ or $\|\boldsymbol{M}\|_F$ denote the matrix Frobenius norm of a matrix $\boldsymbol{M}$. Unless they are clear from context, the random variables over which we take expectation and probabilities are specified in subscript, i.e. $\mathbb{E}_{\boldsymbol{z}}$ and $\mathbb{P}_{\boldsymbol{z}}$ denote the expectation and probability with respect to the random variable of $\boldsymbol{z}$, respectively.

4.3.2 The Learning Bound of MLIE

In this section, we will discuss the learning bound of MLIE based on the diameter of the parameter space and the Lipschitz constant. To start with, we define the pseudometric space, the diameter of a parameter space and the Lipschitz constant as follows.

Definition 27. A *pseudometric space*[2] $(\mathcal{V}, \rho)$ is a set $\mathcal{V}$ and a function ρ: $\mathcal{V} \times \mathcal{V} \to [0, \infty)$ satisfying: (1)$\rho(\boldsymbol{x}, \boldsymbol{y}) \geq 0$; (2) $\rho(\boldsymbol{x}, \boldsymbol{y}) = \rho(\boldsymbol{y}, \boldsymbol{x})$; (3) $\rho(\boldsymbol{x}, \boldsymbol{z}) \leq \rho(\boldsymbol{x}, \boldsymbol{y}) + \rho(\boldsymbol{y}, \boldsymbol{z})$.

Definition 28. Let $\mathcal{U} \in \mathcal{V}$ and $(\mathcal{V}, \rho)$ be a metric space. The *diameter of a set* $\mathcal{U}$ is defined as

$$\operatorname{diam}(\mathcal{U}, \rho) = \max_{\boldsymbol{u}_i, \boldsymbol{u}_j \in \mathcal{U}} \rho(\boldsymbol{u}_i, \boldsymbol{u}_j).$$

Definition 29. [74] Let $(\mathcal{U}, \rho_{\mathcal{U}})$, $(\mathcal{V}, \rho_{\mathcal{V}})$ be two metric spaces. A function $h : \mathcal{U} \to \mathcal{V}$ is called *Lipschitz continuous* if $\exists L < \infty, \forall \boldsymbol{u}_1, \boldsymbol{u}_2 \in \mathcal{U}$,

$$\rho_{\mathcal{V}}(h(\boldsymbol{u}_1), h(\boldsymbol{u}_2)) \leq L \rho_{\mathcal{U}}(\boldsymbol{u}_1, \boldsymbol{u}_2).$$

[2] To simplify the discussion, we refer to pseudometrics as metrics, pointing out the distinction only when necessary.

The *Lipschitz constant of a Lipschitz function* h is

$$\begin{aligned}\mathrm{lip}(h;\mathcal{U}\to\mathcal{V}) &= \min\{L\in\mathbb{R}|\forall \boldsymbol{u}_1,\boldsymbol{u}_2\in\mathcal{U},\boldsymbol{u}_1\neq\boldsymbol{u}_2,\\ &\qquad \rho_{\mathcal{V}}(h(\boldsymbol{u}_1),h(\boldsymbol{u}_2))\leq L\rho_{\mathcal{U}}(\boldsymbol{u}_1,\boldsymbol{u}_2)\}\\ &= \max_{\boldsymbol{u}_1,\boldsymbol{u}_2\in\mathcal{U}:\boldsymbol{u}_1\neq\boldsymbol{u}_2}\frac{\rho_{\mathcal{V}}(h(\boldsymbol{u}_1),h(\boldsymbol{u}_2))}{\rho_{\mathcal{U}}(\boldsymbol{u}_1,\boldsymbol{u}_2)},\end{aligned}$$

$\mathrm{lip}(h;\mathcal{U}\to\mathcal{V})$ will be written as $\mathrm{lip}(h\leftarrow\boldsymbol{u})$ if $\mathcal{U}$ and $\mathcal{V}$ are clear from the context.

Theorem 5. Let $h(\boldsymbol{z};\boldsymbol{\theta})$ be a parameterised function and $\boldsymbol{\theta}\in\mathbb{R}^Q$. Suppose $\mathrm{lip}(h\leftarrow\boldsymbol{\theta})\leq L_1$, $\mathrm{lip}(r\leftarrow h)\leq L_2$, and $\mathrm{diam}(\Theta,\|\cdot\|)\leq B$, then $\forall\boldsymbol{\theta}\in\Theta$, with probability at least $1-\delta$, the following bound holds

$$R(h_{\boldsymbol{\theta}})\leq R_n(\boldsymbol{z}^n,h_{\boldsymbol{\theta}})+CL_1L_2B\sqrt{\frac{Q}{n}}+\sqrt{\frac{\ln 1/\delta}{2n}},$$

where C is a universal constant.

The proof procedure of Theorem 5 mainly follows Peter Bartlett's notes on covering numbers, chaining and Dudleys integral [2] and the details are shown in Appendix 4.7.1. Based on Theorem 5, controlling $\mathrm{lip}(h\leftarrow\boldsymbol{\theta})$ and $\mathrm{diam}(\Theta)$ is an efficient way to control the generalisation gap. In this way, we can improve the generalisation ability of a classifier.

Suppose a parameterised function set has k sets of parameters $\boldsymbol{\theta}=(\boldsymbol{\theta}_{[1]},\ldots,\boldsymbol{\theta}_{[K]})$ and $\mathrm{lip}(h\leftarrow\boldsymbol{\theta})$ may not be easily calculated by using the k sets together. In this case, we propose to calculate $\mathrm{lip}(h\leftarrow\boldsymbol{\theta})$ with k sets separately. We then show the extension of Theorem 5 with k parameter sets as follows.

Definition 30. Suppose a function with multiple parameter vectors $h(\cdot;\boldsymbol{\theta}_{[1]},\ldots,\boldsymbol{\theta}_{[K]})$ maps $\boldsymbol{z}$ into a real value, where $\boldsymbol{\theta}_{[k]}$ denotes the kth parameter vector, $\boldsymbol{\theta}_{[k]}\in\Theta_{[k]}\subseteq\mathbb{R}^{Q_k}$, and K denotes the total number. Given $h(\cdot;\cdot,\boldsymbol{\theta}_{[k]})\in\mathcal{H}_{\boldsymbol{\theta}_{[k]}}$, the metric in space

$\mathcal{H}_{\boldsymbol{\theta}_{[k]}}$ is defined as follows: $\forall \boldsymbol{\theta}_{[k],1}, \boldsymbol{\theta}_{[k],2} \in \Theta_{[k]}$

$$\begin{aligned}
&\rho_{\mathcal{H}_{\boldsymbol{\theta}_{[k]}}}\Big(h(\cdot;\cdot,\boldsymbol{\theta}_{[k],1}), h(\cdot;\cdot,\boldsymbol{\theta}_{[k],2})\Big)\\
=&\max_{\boldsymbol{z},\boldsymbol{\theta}_{[i]},i\in[K]\backslash k}\Big|h(\boldsymbol{z};\boldsymbol{\theta}_{[1]},\ldots,\boldsymbol{\theta}_{[k-1]},\boldsymbol{\theta}_{[k],1},\boldsymbol{\theta}_{[k+1]},\ldots,\boldsymbol{\theta}_{[K]})\\
&-h(\boldsymbol{z};\boldsymbol{\theta}_{[1]},\ldots,\boldsymbol{\theta}_{[k-1]},\boldsymbol{\theta}_{[k],2},\boldsymbol{\theta}_{[k+1]},\ldots,\boldsymbol{\theta}_{[K]})\Big|
\end{aligned} \tag{4.4}$$

where $\boldsymbol{\theta}_{[i]} \in \Theta_{[i]}, i \in [K]\backslash k$ and $[K] = \{1, 2, \ldots, K\}$, $[K]\backslash k$ denotes the integers from 1 to K without k.

Based on the definition, $\rho_{\mathcal{H}_{\boldsymbol{\theta}_{[k]}}}$ satisfies all conditions of pseudometric. The non-negativity and symmetry properties could be easily verified. The triangle inequality is proved as follows:

$$\begin{aligned}
&\rho_{\mathcal{H}_{\boldsymbol{\theta}_{[k]}}}\Big(h(\cdot;\cdot,\boldsymbol{\theta}_{[k],1}), h(\cdot;\cdot,\boldsymbol{\theta}_{[k],2})\Big)\\
=&\max_{\boldsymbol{z},\boldsymbol{\theta}_{[K]\backslash k}}|h(\boldsymbol{z},\boldsymbol{\theta}_{[K]\backslash k},\boldsymbol{\theta}_{[k],1}) - h(\boldsymbol{z},\boldsymbol{\theta}_{[K]\backslash k},\boldsymbol{\theta}_{[k],3}) + h(\boldsymbol{z},\boldsymbol{\theta}_{[K]\backslash k},\boldsymbol{\theta}_{[k],3}) - h(\boldsymbol{z},\boldsymbol{\theta}_{[K]\backslash k},\boldsymbol{\theta}_{[k],2})|\\
\leq&\max_{\boldsymbol{z},\boldsymbol{\theta}_{[K]\backslash k}}\Big(|h(\boldsymbol{z},\boldsymbol{\theta}_{[K]\backslash k},\boldsymbol{\theta}_{[k],1}) - h(\boldsymbol{z},\boldsymbol{\theta}_{[K]\backslash k},\boldsymbol{\theta}_{[k],3})| + |h(\boldsymbol{z},\boldsymbol{\theta}_{[K]\backslash k},\boldsymbol{\theta}_{[k],3}) - h(\boldsymbol{z},\boldsymbol{\theta}_{[K]\backslash k},\boldsymbol{\theta}_{[k],2})|\Big)\\
\leq&\max_{\boldsymbol{z},\boldsymbol{\theta}_{[K]\backslash k}}|h(\boldsymbol{z},\boldsymbol{\theta}_{[K]\backslash k},\boldsymbol{\theta}_{[k],1}) - h(\boldsymbol{z},\boldsymbol{\theta}_{[K]\backslash k},\boldsymbol{\theta}_{[k],3})| + \max_{\boldsymbol{z},\boldsymbol{\theta}_{[K]\backslash k}}|h(\boldsymbol{z},\boldsymbol{\theta}_{[K]\backslash k},\boldsymbol{\theta}_{[k],3}) - h(\boldsymbol{z},\boldsymbol{\theta}_{[K]\backslash k},\boldsymbol{\theta}_{[k],2})|\\
=&\rho_{\mathcal{H}_{\boldsymbol{\theta}_{[k]}}}\Big(h(\cdot;\cdot,\boldsymbol{\theta}_{[k],1}), h(\cdot;\cdot,\boldsymbol{\theta}_{[k],3})\Big) + \rho_{\mathcal{H}_{\boldsymbol{\theta}_{[k]}}}\Big(h(\cdot;\cdot,\boldsymbol{\theta}_{[k],3}), h(\cdot;\cdot,\boldsymbol{\theta}_{[k],2})\Big)
\end{aligned}$$

where $\boldsymbol{\theta}_{[K]\backslash k}$ denotes $\boldsymbol{\theta}_{[i]} \in \Theta_{[i]}, i \in [K]\backslash k$ and the triangle inequality has been proved.

$\mathrm{lip}\Big(h; \Theta_{[k]} \to \mathcal{H}_{\boldsymbol{\theta}_{[k]}}\Big)$ will be written as $\mathrm{lip}(h \leftarrow \boldsymbol{\theta}_{[k]})$ if $\Theta_{[k]}$ and $\mathcal{H}_{\boldsymbol{\theta}_{[k]}}$ are clear from the context. Using the Euclidean distance in the space $\Theta_{[k]}$, $\mathrm{lip}(h \leftarrow \Theta_{[k]})$ has the following formula

$$\mathrm{lip}(h \leftarrow \boldsymbol{\theta}_{[k]}) = \max_{\boldsymbol{\theta}_{[k],1},\boldsymbol{\theta}_{[k],2}\in\Theta_{[k]},\boldsymbol{\theta}_{[k],1}\neq\boldsymbol{\theta}_{[k],2}} \frac{\rho_{\mathcal{H}(\boldsymbol{\theta}_{[k]})}\Big(h(\cdot;\cdot\boldsymbol{\theta}_{[k],1}), h(\cdot;\cdot\boldsymbol{\theta}_{[k],2})\Big)}{\|\boldsymbol{\theta}_{[k],1} - \boldsymbol{\theta}_{[k],2}\|}. \tag{4.5}$$

Theorem 6. The relationship between $L_{[k]} = \mathrm{lip}(h \leftarrow \boldsymbol{\theta}_{[k]})$ and $L = \mathrm{lip}(h \leftarrow \boldsymbol{\theta})$

can be written as

$$L \leq \sqrt{\sum_{k=1}^{K} L_{[k]}^2}.$$

Thus let $\boldsymbol{L} = (L_{[1]}, \ldots, L_{[K]}) \in \mathbb{R}^K$,

$$L \leq \|\boldsymbol{L}\|_2.$$

The detailed proof of Theorem 6 is shown in Appendix 4.7.2.

Corollary 4. Let $h(\boldsymbol{z}; \boldsymbol{\theta})$ be a parameterised function and $\boldsymbol{\theta} = (\boldsymbol{\theta}_{[1]}, \ldots, \boldsymbol{\theta}_{[K]})$. Suppose $\mathrm{lip}(r \leftarrow h) \leq L_l$, $\mathrm{lip}(h \leftarrow \boldsymbol{\theta}_k) \leq L_{[k]}$ and $\mathrm{diam}(\Theta_{[k]}, \|\cdot\|_2) \leq B_{[k]}$. Let $\boldsymbol{L} = (L_{[1]}, \ldots, L_{[K]})$ and $\boldsymbol{B} = (B_{[1]}, \ldots, B_{[K]})$. Then $\forall \boldsymbol{\theta} \in \Theta$, with probability $1 - \delta$, the following bound holds

$$R(h_{\boldsymbol{\theta}}) \leq R_n(\boldsymbol{z}^n, h_{\boldsymbol{\theta}}) + C L_l \|\boldsymbol{L}\| \|\boldsymbol{B}\| \sqrt{\frac{Q}{n}} + \sqrt{\frac{\ln 1/\delta}{2n}},$$

where C is a universal constant.

Proof. Substitute the result of Theorem 6 into Theorem 5 and check the definition of B, the result is obtained. □

Based on Corollary 4, the three components, L_l, $\|\boldsymbol{L}\|$ and $\|\boldsymbol{B}\|$ can affect the bound of the expected risk. In MLIE, the set of parameters is $\boldsymbol{\theta} = \{\boldsymbol{r}^m, \boldsymbol{M}\}$. To guarantee the generalisation ability of MLIE, the following factors are considered:

1. L_l: The loss function with smaller $\mathrm{lip}(r \leftarrow h)$ is preferred. Therefore, Hinge loss with $L_l = 1$ is selected.

2. $\|\boldsymbol{B}\|$: Based on the definition of $\boldsymbol{B}$, the parameters are considered separately. The space of $\boldsymbol{r}_i^- / \boldsymbol{r}_j^+$ in (4.1), (4.2) and (4.3) is bounded when L_2-norm regularisation is used. The space of $\boldsymbol{L}$ or $\boldsymbol{M}$ can be bounded when matrix Frobenius norm regularisation is used.

3. $||\boldsymbol{L}||$: With a convex regularisation term, as illustrated in Section 1.6.2, the parameter is restricted to be inside a convex set. Based on Corollary 1 in Chapter 1, the Lipschitz constant with respect to each parameter can be bounded by $\max_{\boldsymbol{z},\boldsymbol{\theta}} ||\frac{\partial h(\boldsymbol{z};\boldsymbol{\theta})}{\partial \theta_{[i]}}||$. The partial gradients of the classifier (4.3) can be bounded so long as $\mathcal{X}$ is bounded, which would be illustrated in Section 4.4.2. Here we use G_{max} to denote the bound of $||\boldsymbol{L}||$.

Then the learning bound of the algorithm can be written as follows: $\forall \boldsymbol{\theta} \in \Theta$, with probability at least $1-\delta$, the following bound holds

$$R(h_{\boldsymbol{\theta}}) \leq R_n(\boldsymbol{z}^n, h_{\boldsymbol{\theta}}) + CG_{max}B\sqrt{\frac{Q}{n}} + \sqrt{\frac{\ln 1/\delta}{2n}},$$

where $B \leq \sqrt{\sum_i \mathrm{diam}(\boldsymbol{r}_i^-)^2 + \mathrm{diam}(\boldsymbol{r}_j^+)^2 + \mathrm{diam}(\boldsymbol{M})^2}$ or $B \leq \sqrt{\sum_i \mathrm{diam}(\boldsymbol{r}_i^-)^2 + \mathrm{diam}(\boldsymbol{r}_j^+)^2 + \mathrm{diam}(\boldsymbol{L})^2}$, where $\mathrm{diam}(\boldsymbol{r}_i^-)$, $\mathrm{diam}(\boldsymbol{r}_j^+)$, $\mathrm{diam}(\boldsymbol{M})$ and $\mathrm{diam}(\boldsymbol{L})$ denote the diameters of the space restricted by vector L_2-norm or matrix Frobenius norm regularisation.

4.4 Algorithm of MLIE

4.4.1 Objective Function

Based on the discussion in previous sections, with the classifier of (4.3), hinge loss and the regularisation term $\sum_i ||\boldsymbol{r}_i^-||_2^2 + ||\boldsymbol{r}_j^+||_2^2 + ||\boldsymbol{L}||_F^2$, the following optimisation problem is proposed:

$$\begin{aligned} \min_{\Theta,\boldsymbol{\xi}} \quad & \tfrac{1}{n}\textstyle\sum_i \xi_i + \lambda\Big(\textstyle\sum_i ||\boldsymbol{r}_i^-||_2^2 + ||\boldsymbol{r}_j^+||_2^2 + ||\boldsymbol{L}||_F^2\Big) \\ s.t. \quad & y_i h(\boldsymbol{x}_i; \boldsymbol{r}^m, \boldsymbol{M}) \geq 1 - \xi_i \\ & \xi_i \geq 0 \\ & i = 1, \ldots, n, \end{aligned} \tag{4.6}$$

where $h(\boldsymbol{x}; \boldsymbol{r}^m, \boldsymbol{M}) = \min_i(\boldsymbol{L}\boldsymbol{x} - \boldsymbol{r}_i^-)^T(\boldsymbol{L}\boldsymbol{x} - \boldsymbol{r}_i^-) - \min_j(\boldsymbol{L}\boldsymbol{x} - \boldsymbol{r}_j^-)^T(\boldsymbol{L}\boldsymbol{x} - \boldsymbol{r}_j^-)$ denotes the classifier; $\Theta = \{\boldsymbol{r}^m, \boldsymbol{L}\}$ denotes the set of parameters to be optimised; $\boldsymbol{L}$ denotes the linear mapping to be learned and the learning of $\boldsymbol{L}$ is equivalent to the learning of $\boldsymbol{M}$ of Mahalanobis distance; $\boldsymbol{r}^m = \{\boldsymbol{r}_i^-, \boldsymbol{r}_j^+; i = 1, \dots, m_+, j = 1, \dots, m_-\}$ denotes the set of all extracted instances; α is a trade-off parameter which balances the loss term and the regularisation term.

4.4.2 Gradient Descent

For the reason of the non-convexity of $h(\boldsymbol{x}; \boldsymbol{r}^m, \boldsymbol{M})$, the optimisation problem is not a convex one. Thus the gradient descent algorithm is used:

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

where β is the learning rate; the superscript t denotes the time step during optimisation; g denotes the objective function.

The partial derivative of the objective function g with respect to each set of parameters is

$$\frac{\partial g}{\partial \Theta}|_{\Theta^t} = \Big(\frac{1}{n} \sum_i y_i \frac{\partial l}{\partial h(\boldsymbol{x}_i; \Theta)} \frac{\partial h}{\partial \Theta} - 2\lambda\Theta\Big)|_{\Theta^t},$$

where l indicates the Hinge loss function;

$$\frac{\partial h(\boldsymbol{x}_k; \Theta)}{\partial \boldsymbol{r}_a^-}|_{\Theta^t} = \mathbb{1}\Big[a = \mathrm{argmin}_i\, d^2(\boldsymbol{L}\boldsymbol{x}_k, \boldsymbol{r}_i^-)\Big](2\boldsymbol{r}_a^- - 2\boldsymbol{L}\boldsymbol{x}_k)|_{\Theta^t};$$

$$\frac{\partial h(\boldsymbol{x}_k; \Theta)}{\partial \boldsymbol{r}_b^+}|_{\Theta^t} = -\mathbb{1}\Big[b = \mathrm{argmin}_j\, d^2(\boldsymbol{L}\boldsymbol{x}_k, \boldsymbol{r}_j^-)\Big](2\boldsymbol{r}_b^+ - 2\boldsymbol{L}\boldsymbol{x}_k)|_{\Theta^t};$$

$$\begin{aligned}\frac{\partial h(\boldsymbol{x}_k; \Theta)}{\partial \boldsymbol{L}}|_{\Theta^t} =& \sum_a \mathbb{1}\Big[a = \mathrm{argmin}_i\, d^2(\boldsymbol{L}\boldsymbol{x}_k, \boldsymbol{r}_i^-)\Big] 2(\boldsymbol{L}\boldsymbol{x}_k - \boldsymbol{r}_a^-)\boldsymbol{x}_k^T|_{\Theta^t} \\ &- \sum_b \mathbb{1}\Big[b = \mathrm{argmin}_j\, d^2(\boldsymbol{L}\boldsymbol{x}_k, \boldsymbol{r}_j^-)\Big] 2(\boldsymbol{L}\boldsymbol{x}_k - \boldsymbol{r}_b^+)\boldsymbol{x}_k^T|_{\Theta^t};\end{aligned}$$

$\mathbb{1}(\cdot)$ denotes the indicator function, its value is 1 when the condition is satisfied and its value is 0 otherwise.

Based on the equation, $\frac{\partial h(\boldsymbol{x}_k;\Theta)}{\partial \boldsymbol{r}_a^-}|_\Theta$ and $\frac{\partial h(\boldsymbol{x}_k;\Theta)}{\partial \boldsymbol{r}_b^+}|_\Theta$ are bounded by $2\,\mathrm{diam}(\boldsymbol{r}_a^-) + 2\,\mathrm{diam}(\boldsymbol{L})\,\mathrm{diam}(\boldsymbol{x})$ and $2\,\mathrm{diam}(\boldsymbol{r}_b^+) + 2\,\mathrm{diam}(\boldsymbol{L})\,\mathrm{diam}(\boldsymbol{x})$ respectively; $\frac{\partial h(\boldsymbol{x}_k;\Theta)}{\partial \boldsymbol{L}}|_\Theta$ is bounded by $4(\mathrm{diam}(\boldsymbol{r}_a^-) + \mathrm{diam}(\boldsymbol{L})\,\mathrm{diam}(\boldsymbol{x}))\,\mathrm{diam}(\boldsymbol{x})$. Therefore, as long as the regularisation terms are convex, the related $\mathrm{lip}(h \leftarrow \boldsymbol{\theta})$ is bounded in Corollary 4. The gradient of hinge loss $l(a)$ is -1 when $a \leq 1$ and 0 when $a > 1$ and the final updating equation is as follows:

$$\begin{aligned}\boldsymbol{r}_a^{-,t+1} &= \boldsymbol{r}_a^{-,t} - 2\lambda\alpha\boldsymbol{r}_a^{-,t} \\ &+ \frac{\alpha}{n}\sum_{k=1}^{n} y_k \mathbb{1}\Big[y_k h(\boldsymbol{x}_k;\Theta) \leq 1\Big]\mathbb{1}\Big[a = \mathrm{argmin}_i\, d^2(\boldsymbol{L}\boldsymbol{x}_k, \boldsymbol{r}_i^-)\Big](2\boldsymbol{r}_a^- - 2\boldsymbol{L}\boldsymbol{x}_k)|_{\Theta^t};\end{aligned}$$

$$\begin{aligned}\boldsymbol{r}_b^{+,t+1} &= \boldsymbol{r}_b^{+,t} - 2\lambda\alpha\boldsymbol{r}_b^{-,t} \\ &- \frac{\alpha}{n}\sum_{k=1}^{n} y_k \mathbb{1}\Big[y_k h(\boldsymbol{x}_k;\Theta) \leq 1\Big]\mathbb{1}\Big[b = \mathrm{argmin}_j\, d^2(\boldsymbol{L}\boldsymbol{x}_k, \boldsymbol{r}_j^-)\Big](2\boldsymbol{r}_b^+ - 2\boldsymbol{L}\boldsymbol{x}_k)|_{\Theta^t};\end{aligned}$$

$$\begin{aligned}\boldsymbol{L}^{t+1} &= \boldsymbol{L}^t - 2\lambda\alpha\boldsymbol{L}^t \\ &+ \frac{\alpha}{n}\sum_{k=1}^{n} y_k \mathbb{1}\Big[y_k h(\boldsymbol{x}_k;\Theta) \leq 1\Big]\sum_a \mathbb{1}\Big[a = \mathrm{argmin}_i\, d^2(\boldsymbol{L}\boldsymbol{x}_k, \boldsymbol{r}_i^-)\Big]2(\boldsymbol{L}\boldsymbol{x}_k - \boldsymbol{r}_a^-)\boldsymbol{x}_k^T|_{\Theta^t} \\ &- \frac{\alpha}{n}\sum_{k=1}^{n} y_k \mathbb{1}\Big[y_k h(\boldsymbol{x}_k;\Theta) \leq 1\Big]\sum_b \mathbb{1}\Big[b = \mathrm{argmin}_j\, d^2(\boldsymbol{L}\boldsymbol{x}_k, \boldsymbol{r}_j^-)\Big]2(\boldsymbol{L}\boldsymbol{x}_k - \boldsymbol{r}_b^+)\boldsymbol{x}_k^T|_{\Theta^t}.\end{aligned}$$

4.5 Experiments

The proposed algorithm is compared with nine established metric learning algorithms from two categories: 1) the most cited algorithms, including large margin nearest neighbor (LMNN) [76], information theoretic metric learning (ITML) [9], neighbourhood component analysis (NCA) [17] and metric learning by collapsing classes (MCML) [16]; (2) the most state-of-the-art algorithms, including geometric mean metric learning (GMML) [83], regressive virtual metric learning (RVML) [52], stochastic neighbour compression (SNC) [35], sparse compositional metric learning (SCML) [59] and reduced-rank local distance metric learning (R2LML) [26]. LMNN and ITML are implemented by using the metric-learn toolbox[3]; NCA and MCML are implemented by using the drToolbox[4]; and GMML, RVML, SCML, R2LML and SNC are implemented by using the authors' code.

The experiment is focused on binary classification of 12 publicly available data

[3] https://all-umass.github.io/metric-learn/

[4] https://lvdmaaten.github.io/drtoolbox/

Table 4.2: Experiment results of metric learning with instance extraction. Mean accuracy (percentage) and standard deviations are reported with the best ones in bold; '# of best' indicates the number of data sets that an algorithm performs the best.

Data set	LMNN	ITML	MCML	NCA	RVML
Australian	78.8 ±2.57	77.17 ±1.94	78.77 ±1.70	79.96 ±1.63	83.01 ±1.58
Breastcancer	95.91 ±0.69	96.39 ±1.04	96.35 ±0.77	95 ±1.52	95.77 ±1.09
Diabetes	69.16 ±1.44	69.09 ±1.24	69.19 ±1.18	68.47 ±2.46	71.04 ±2.60
Fourclass	72.06 ±2.31	72.09 ±2.22	72.06 ±2.43	72.06 ±2.46	70.46 ±1.40
German	67.85 ±1.54	66.95 ±2.05	67.67 ±1.48	69.95 ±2.88	71.65 ±1.78
Haberman	67.89 ±3.34	67.97 ±4.05	67.56 ±2.75	67.4 ±3.33	66.67 ±2.30
Heart	76.2 ±3.82	76.94 ±3.30	77.22 ±3.66	75.56 ±2.01	77.69 ±4.05
ILPD	66.97 ±2.13	68.67 ±2.83	67.48 ±2.58	66.8 ±1.19	67.95 ±2.90
Liverdisorders	61.01 ±4.80	57.17 ±4.01	60.65 ±5.12	59.78 ±3.44	64.64 ±3.93
Pima	68.54 ±1.64	67.95 ±2.01	68.31 ±2.33	65.91 ±3.04	69.45 ±1.68
Voting	94.83 ±0.77	90.75 ±1.44	92.64 ±1.58	94.77 ±0.92	95.75 ±1.26
WDBC	96.58 ±1.12	94.91 ±0.92	95.7 ±0.90	96.58 ±0.85	96.58 ±1.34
# of best	0	0	0	0	0

Data set	GMML	SCML	R2LML	SNC	MLIE
Australian	84.35 ±1.04	82.25 ±1.40	84.67 ±1.32	81.78 ±8.8	**84.71 ±1.93**
Breastcancer	**97.26 ±0.81**	97.01 ±0.91	97.01 ±0.66	96.65 ±0.69	96.18 ±1.32
Diabetes	74.16 ±2.58	71.49 ±2.21	73.8 ±1.37	**75.32 ±2.74**	73.42 ±3.59
Fourclass	76.12 ±1.87	75.54 ±1.42	76.12 ±1.91	73.39 ±8.7	**77.02 ±1.73**
German	71.55 ±1.12	70.9 ±2.65	72.9 ±1.83	70.13 ±3.33	**73.63 ±2.2**
Haberman	71.22 ±3.35	69.19 ±2.47	71.06 ±3.39	71.98 ±5.2	**72.56 ±3.25**
Heart	81.2 ±2.69	78.98 ±3.24	**82.04 ±3.81**	77.04 ±5.32	80.83 ±3.09
ILPD	67.14 ±2.17	68.03 ±2.90	65.85 ±2.22	68.91 ±2.67	**69.63 ±2.58**
Liverdisorders	63.84 ±5.43	61.74 ±4.57	**66.81 ±3.68**	63.31 ±5.18	61.44 ±3.99
Pima	72.95 ±1.84	71.14 ±2.64	72.34 ±1.54	**73.99 ±2.59**	73.78 ±2.68
Voting	95.17 ±1.88	95 ±1.30	**96.32 ±1.19**	94.45 ±1.2	94.91 ±1.48
WDBC	96.71 ±0.78	**96.97 ±0.89**	96.93 ±1.67	96.93 ±0.85	95.17 ±1.37
# of best	1	1	3	2	5

sets from the websites of UCI[5] and LibSVM[6], namely Australian, Breastcancer, Diabetes, Fourclass, Germannumber, Haberman, Heart, ILPD, Liverdisorders, Pima, Voting and WDBC. All data sets are pre-processed by firstly subtracting the mean and dividing by the standard deviation, and then normalizing the l_2-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

[5] https://archive.ics.uci.edu/ml/datasets.html

[6] https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/binary.html

accuracy and the standard deviation are reported. 10-fold cross-validation is used to select the trade-off parameters in the compared algorithms, namely the regularisation 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 $\{0.1, 0.3, 0.5, 0.7, 0.9\}$), λ in RVML (from $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$), and ratio in SNC (from $\{0.01, 0.02, 0.04, 0.08, 0.16\}$). All other parameters are set as default. For the proposed algorithm, the parameters are set as follows: the initial weight of $\boldsymbol{L}$ is set as the identity matrix $\boldsymbol{I}$; the initial values of $\boldsymbol{r}^m$ are set as the k-means clustering (Matlab kmeans function with random initial values) centres of the positive and negative classes; the number of extracted instances for each class is set as 2; the trade-off parameter λ is set as 1 and the learning rate α is set as 0.001. The maximum number of iterations is set as 5000 and the final result is based on the parameters at time t, which is the earliest time when the smallest training error is obtained.

As shown in Table 5.1, MLIE achieves the best accuracy on 5 data sets out of the 12 data sets. None of the other algorithms performs the best in more than 3 data sets. These experiment results show that MLIE enjoys competitive performance against state-of-the-art metric learning algorithms. Furthermore, based on the intuitive of LLC, in the bench mark experiment, only two instances from each class are extracted, which would result into four local regions. MLIE has provided an effective way to conduct instance compression for NN classifier.

4.6 Conclusion

In this chapter, a MLIE classifier is proposed and the classification mechanism of MLIE is illustrated by showing its relationship with LLC. Meanwhile, the learning bound of MLIE has been obtained, which guarantees the generalisation ability of MLIE. The experiments on benchmark data sets show the competitive classification and compression performance of MLIE.

4.7 Appendix

4.7.1 Proof of Theorem 5

First, the definitions of Rademacher complexity, uniform convergence and covering number are introduced. Dudley's Integral Theorem that uses covering number to bound Rademacher complexity is also introduced. Then, the Lipschitz constant is shown to bound the covering number of functional space by the covering number of parameter space. Finally, based on Dudley's Integral Theorem, Theorem 5 is shown.

4.7.1.1 Preliminary

Definition 31. [46] Let $\boldsymbol{\epsilon}^n = \{\epsilon_1, \dots \epsilon_n\}$ be i.i.d. ±1-valued random variables with $P(\epsilon_i = +1) = P(\epsilon_i = -1) = \frac{1}{2}$. $\boldsymbol{z}^n = \{\boldsymbol{z}_1, \dots, \boldsymbol{z}_n\}$ are i.i.d. samples. The *empirical Rademacher complexity* is defined as

$$\hat{\mathrm{Rad}}_n(\mathcal{H}) = \mathbb{E}_{\boldsymbol{\epsilon}^n}\Big[\max_{h\in\mathcal{H}} \frac{1}{n}\sum_i \epsilon_i h(\boldsymbol{z}_i)\Big|\boldsymbol{z}^n\Big];$$

and the *Rademacher complexity* is defined as

$$\mathrm{Rad}(\mathcal{H}) = \mathbb{E}_{\boldsymbol{z}^n}\Big[\hat{\mathrm{Rad}}_n(\mathcal{H})\Big].$$

Theorem 7. [46] With probability at least $1-\delta$ the following bounds hold

$$R(h) - R_n(\boldsymbol{z}^n, h) \leq 2\hat{\mathrm{Rad}}_n(\phi\circ\mathcal{H}) + 3\sqrt{\frac{\ln\frac{2}{\delta}}{2n}},$$

where $\phi : \mathbb{R} \to \mathbb{R}$ denotes the loss function $l(h(\boldsymbol{x}); y)$; $\circ$ denotes the composition of functions.

Lemma 8. [46] Let $\phi : \mathbb{R} \to \mathbb{R}$ be an L-Lipschitz. Then, for any hypothesis set $\mathcal{H}$ of real-valued functions, *Talagrands Lemma* indicates the following inequality holds:

$$\hat{\mathrm{Rad}}_n(\phi\circ\mathcal{H}) \leq L\hat{\mathrm{Rad}}_n(\mathcal{H}).$$

Corollary 5. Suppose $\mathrm{lip}(r \leftarrow h) \leq L$, then with probability at least $1 - \delta$ the following bound holds

$$R(h) - R_n(\boldsymbol{z}^n, h) \leq 2L\hat{\mathrm{Rad}}_n(\mathcal{H}) + 3\sqrt{\frac{\ln \frac{2}{\delta}}{2n}}$$

Proof. Substitute the result of Lemma 8 into Theorem 7, the result can be obtained. □

Definition 32. [68] An *ϵ-cover* of a subset $\mathcal{U}$ of a metric space $(\mathcal{V}, \rho)$ is a set $\hat{\mathcal{U}} \subseteq \mathcal{U}$ such that for each $\boldsymbol{u} \in \mathcal{U}$ there is a $\hat{\boldsymbol{u}} \in \hat{\mathcal{U}}$ such that $\rho(\boldsymbol{u}, \hat{\boldsymbol{u}}) \leq \epsilon$. The *$\epsilon$-cover number* of $\mathcal{U}$ is

$$N(\epsilon, \mathcal{U}, \rho) = \min\{|\hat{\mathcal{U}}| : \hat{\mathcal{U}} \text{ is an } \epsilon\text{-cover of } \mathcal{U}\}.$$

The following theorem illustrates how to bound the covering number.

Theorem 8. [68] Let $\mathcal{U} \subseteq \mathcal{V} = \mathbb{R}^D$. Then

$$\Big(\frac{1}{\epsilon}\Big)^D \frac{\mathrm{vol}(\mathcal{U})}{\mathrm{vol}(\mathcal{B})} \leq N(\epsilon, \mathcal{U}, \|\cdot\|) \leq \Big(\frac{\mathrm{vol}(\mathcal{U} + \frac{\epsilon}{2}\mathcal{B})}{\mathrm{vol}(\frac{\epsilon}{2}\mathcal{B})}\Big)$$

where $+$ is the Minkovski sum, $\mathcal{B}$ is the unit norm ball and vol indicates the volume of the set.

Remark: Consider $\mathcal{U} \in \mathbb{R}^D$ with diameter $\mathrm{diam}(\mathcal{U})$, then based on the last inequality, we have

$$N(\epsilon, \mathcal{U}, \|\cdot\|) \leq \Big(\frac{\mathrm{vol}(\mathcal{U} + \frac{\epsilon}{2}\mathcal{B})}{\mathrm{vol}(\frac{\epsilon}{2}\mathcal{B})}\Big) \leq \Big(\frac{\mathrm{diam}(\mathcal{U}) + \epsilon}{\epsilon}\Big)^D = \Big(1 + \frac{\mathrm{diam}(\mathcal{U})}{\epsilon}\Big)^D.$$

Definition 33. Let $\forall h_1, h_2 \in \mathcal{H}$ be two functions mapping $\boldsymbol{z} \in \mathcal{Z}$ into real value, $\rho_{\mathcal{H}|\boldsymbol{z}^n}$ is defined as follows:

$$\rho_{\mathcal{H}|\boldsymbol{z}^n}(h_1, h_2) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(h_1(\boldsymbol{z}_i) - h_2(\boldsymbol{z}_i))^2}.$$

Theorem 9. [61] With metric $\rho_{\mathcal{H}|\boldsymbol{z}^n}$ on $\mathcal{H}$, *Dudley's integral* indicates

$$\hat{\operatorname{Rad}}_n(\mathcal{H}) \leq 12 \int_0^{\infty} \sqrt{\frac{\log N(\epsilon, \mathcal{H}, \rho_{\mathcal{H}|\boldsymbol{z}^n})}{n}} d\epsilon.$$

Dudley's integral bounds the empirical Rademacher complexity by the covering number of the function space (with a metric based on the difference of the function value on n inputs).

4.7.1.2 Learning Bounds with B and Lipschitz constant

To start with, another definition of metric in function space is given as follows.

Definition 34. A metric $\rho_{\mathcal{H}_{\boldsymbol{\theta}}}$ in parametric function space is defined as follows:

$$\rho_{\mathcal{H}_\theta}(h(\cdot;\boldsymbol{\theta}_1), h(\cdot;\boldsymbol{\theta}_2)) = \max_{\boldsymbol{x}\in\mathcal{X}} |h(\boldsymbol{x};\boldsymbol{\theta}_1) - h(\boldsymbol{x};\boldsymbol{\theta}_2)|. \tag{4.7}$$

$\operatorname{lip}(h; \Theta \to \mathcal{H}_{\boldsymbol{\theta}})$ can be written as $\operatorname{lip}(h \leftarrow \boldsymbol{\theta})$ because Θ and $\mathcal{H}_{\boldsymbol{\theta}}$ is clear from the context:

$$\begin{aligned}\operatorname{lip}(h \leftarrow \boldsymbol{\theta}) &= \max_{\boldsymbol{\theta}_1,\boldsymbol{\theta}_2\in\Theta,\boldsymbol{\theta}_1\neq\boldsymbol{\theta}_2} \frac{\rho_{\mathcal{H}_{\boldsymbol{\theta}}}\Big(h(\cdot;\cdot,\boldsymbol{\theta}_1), h(\cdot;\cdot,\boldsymbol{\theta}_2)\Big)}{\rho_\Theta(\boldsymbol{\theta}_1,\boldsymbol{\theta}_2)} \\ &= \max_{\boldsymbol{\theta}_1,\boldsymbol{\theta}_2\in\Theta,\boldsymbol{\theta}_1\neq\boldsymbol{\theta}_2,\boldsymbol{x}} \frac{|h(\boldsymbol{x};\boldsymbol{\theta}_1) - h(\boldsymbol{x};\boldsymbol{\theta}_2)|}{||\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2||}.\end{aligned}$$

Proposition 15. For all spaces of parametric functions $\mathcal{H}_{\boldsymbol{\theta}}$, $\forall\epsilon, \forall\mathcal{H}$,

$$N(\epsilon, \mathcal{H}, \rho_{\mathcal{H}|\boldsymbol{z}^n}) \leq N(\epsilon, \mathcal{H}, \rho_{\mathcal{H}_\theta}), \tag{4.8}$$

where $\boldsymbol{\theta}$ denotes all parameters of the function, $\rho_{\mathcal{H}|\boldsymbol{z}^n}$ is defined in Definition 33 and $\rho_{\mathcal{H}_{\boldsymbol{\theta}}}$ is defined in Definition 34.

Proof. Let $\{\hat{h}_1, \ldots, \hat{h}_N\}$ be an ϵ-covering set in $\mathcal{H}_{\boldsymbol{\theta}}$ with metric $\rho_{\mathcal{H}_\theta}$, then based on the definition of covering set,

$$\forall h \in \mathcal{H}, \min_i \rho_{\mathcal{H}_{\boldsymbol{\theta}}}(h, \hat{h}_i) \leq \epsilon.$$

Based on the definitions of $\rho_{\mathcal{H}|\boldsymbol{z}^n}$ and $\rho_{\mathcal{H}_{\boldsymbol{\theta}}}$, we have

$$\begin{aligned}
\rho_{\mathcal{H}|\boldsymbol{z}^n}(h, \hat{h}_i) &= \sqrt{\frac{1}{n}\sum_{i=1}^{n}(h(\boldsymbol{z}_i) - \hat{h}_i(\boldsymbol{z}_i))^2} \\
&\leq \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\max_{\boldsymbol{z}} |h(\boldsymbol{z}) - \hat{h}_i(\boldsymbol{z}_i)|)^2} \\
&= \sqrt{\frac{1}{n} \times n \times \rho_{\mathcal{H}_{\boldsymbol{\theta}}}(h, \hat{h}_i)} \\
&= \rho_{\mathcal{H}_{\boldsymbol{\theta}}}(h, \hat{h}_i) \\
&\leq \epsilon.
\end{aligned}$$

Therefore, $\{\hat{h}_1, \ldots, \hat{h}_N\}$ is also an ϵ-covering set of $\mathcal{H}_{\boldsymbol{\theta}}$ with metric $\rho_{\mathcal{H}|\boldsymbol{z}^n}$ and

$$N(\epsilon, \mathcal{H}, \rho_{\mathcal{H}|\boldsymbol{z}^n}) \leq |\{\hat{h}_1, \ldots, \hat{h}_N\}| = N(\epsilon, \mathcal{H}, \rho_{\mathcal{H}_{\boldsymbol{\theta}}}).$$

□

Corollary 6. The empirical Rademacher complexity can be bounded by the covering number with metric $\rho_{\mathcal{H}_{\boldsymbol{\theta}}}$ as follows:

$$\hat{\mathrm{Rad}}_n(\mathcal{H}) \leq 12 \int_0^\infty \sqrt{\frac{\log N(\epsilon, \mathcal{H}, \rho_{\mathcal{H}_{\boldsymbol{\theta}}})}{n}} d\epsilon.$$

Proof. Substitute the result of Proposition 15 into Theorem 9. □

Proposition 16. Let $h(\boldsymbol{z}; \boldsymbol{\theta})$ be a parameterised function and $\boldsymbol{\theta} \in \Theta \in \mathbb{R}^Q$. Suppose $\mathrm{lip}(h \leftarrow \boldsymbol{\theta}) \leq L$, i.e.

$$\forall \boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \Theta, \boldsymbol{\theta}_1 \neq \boldsymbol{\theta}_2 \quad \rho_{\mathcal{H}_{\boldsymbol{\theta}}}\Big(h(\cdot; \boldsymbol{\theta}_1), h(\cdot; \boldsymbol{\theta}_2)\Big) \leq L\rho_{\Theta}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2),$$

then

$$
\begin{aligned}
& N(\epsilon, \mathcal{H}_{\boldsymbol{\theta}}, \rho_{\mathcal{H}_{\boldsymbol{\theta}}}) \\
(a) \quad & \leq N(\epsilon/L, \Theta, \rho_\Theta) \\
(b) \quad & \leq \Big(1 + \frac{\mathrm{diam}(\Theta)L}{\epsilon}\Big)^Q.
\end{aligned}
$$

Proof. Let $\{\hat{\boldsymbol{\theta}}_1, \ldots, \hat{\boldsymbol{\theta}}_N\}$ be an (ϵ/L)-covering set in Θ, then based on the definition of covering set,

$$\forall \boldsymbol{\theta} \in \Theta, \min_i \rho_\Theta(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}_i) \leq \epsilon/L$$

Based on the Lipschitz continuous property

$$\forall h(\cdot;\boldsymbol{\theta}) \in \mathcal{H}_{\boldsymbol{\theta}}, \min_i \rho_{\mathcal{H}_\theta}\Big(h(\cdot;\boldsymbol{\theta}), h(\cdot;\hat{\boldsymbol{\theta}}_i)\Big) \leq L \min_i \rho_\Theta(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}_i) \leq \epsilon,$$

Therefore, $\{h(\cdot;\hat{\boldsymbol{\theta}}_1), \ldots, h(\cdot;\hat{\boldsymbol{\theta}}_N)\}$ is a ϵ-covering set of $\mathcal{H}$ and

$$
\begin{aligned}
& N(\epsilon, \mathcal{H}(\boldsymbol{\theta}), \rho_{\mathcal{H}_{\boldsymbol{\theta}}}) \\
(c) \quad & \leq |\{h(\cdot;\hat{\boldsymbol{\theta}}_1), \ldots, h(\cdot;\hat{\boldsymbol{\theta}}_N)\}| \\
(d) \quad & \leq |\{\hat{\boldsymbol{\theta}}_1, \ldots, \hat{\boldsymbol{\theta}}_N\}| \\
(e) \quad & = N(\epsilon/L, \Theta, \rho_\Theta).
\end{aligned}
$$

where the inequality (c) is based on the definition of covering number and the inequality (d) is due to the fact that h is a function. Therefore, inequality (a) of Lemma 16 is proved. inequality (b) of Lemma 16 is based on Theorem 8. □

Finally, Theorem 5 can be proved as follows.

Proof. Based on the result of Corollary 6,

$$\begin{aligned}
\hat{\text{Rad}}_n(\mathcal{H}) &\leq 12\int_0^{\infty}\sqrt{\frac{\log N(\epsilon,\mathcal{H},\rho_{\mathcal{H}_{\boldsymbol{\theta}}})}{n}}d\epsilon \\
(a) \quad &= 12\int_0^{LB}\sqrt{\frac{\log N(\epsilon,\mathcal{H},\rho_{\mathcal{H}_{\boldsymbol{\theta}}})}{n}}d\epsilon \\
(b) \quad &= \frac{12}{\sqrt{n}}\int_0^{LB}\sqrt{\log\Big(1+\frac{LB}{\epsilon}\Big)^Q}d\epsilon \\
(c) \quad &= \frac{12LB}{\sqrt{n}}\int_0^{1}\sqrt{Q\log\Big(1+\frac{1}{\epsilon'}\Big)}d\epsilon' \\
(d) \quad &\leq 12LB\sqrt{\frac{Q}{n}}\int_0^{1}\sqrt{\log(2/\epsilon')}d\epsilon' \\
(e) \quad &= 24LB\sqrt{\frac{Q}{n}}\int_0^{1/2}\sqrt{\log(1/\epsilon)}d\epsilon.
\end{aligned}$$

Here equality (a) is because the value of h is bounded by LB. If $\epsilon > LB$, then $\log N(\epsilon,\mathcal{H},\rho_{\mathcal{H}_{\boldsymbol{\theta}}}) = 0$, equality (b) is based on Proposition 16, equality (c) can be shown by variable substitution $\epsilon' = \frac{\epsilon}{LB}$, inequality (d) is due to $\epsilon \in [0,1]$ and equality (e) is due to variable substitution $\epsilon = \frac{\epsilon'}{2}$.

Then we calculate the integral

$$\begin{aligned}
&\int_0^{1/2}\sqrt{\log(1/\epsilon)}d\epsilon \\
(a) \quad &= \int_{\infty}^{\sqrt{log2}} yd(e^{-y^2}) \\
(b) \quad &= e^{-y^2}y|_{\infty}^{\sqrt{\log 2}} - \int_{\infty}^{\sqrt{\log 2}} e^{-y^2}dy \\
&= e^{-y^2}y|_{\infty}^{\sqrt{\log 2}} + \int_{\sqrt{\log 2}}^{\infty} e^{-y^2}dy \\
&\leq e^{-y^2}y|_{\infty}^{\sqrt{\log 2}} + \int_0^{\infty} e^{-y^2}dy \\
&= \frac{\sqrt{\log 2}}{2} + \frac{\sqrt{\pi}}{2},
\end{aligned}$$

where equality (a) is based on variable substitution $y = \sqrt{\log(1/\epsilon)}$, i.e. $\epsilon = e^{-y^2}$ and equality (b) is based on integral by part.

Therefore,

$$\begin{aligned}\hat{\mathrm{Rad}}_n(\mathcal{H}) &\leq 24LB\sqrt{\frac{Q}{n}}\int_0^{1/2}\sqrt{\log(1/\epsilon)}d\epsilon\\ &\leq 24(\frac{\sqrt{\log 2}}{2}+\frac{\sqrt{\pi}}{2})LB\sqrt{\frac{Q}{n}}\\ &= CLB\sqrt{\frac{Q}{n}}\end{aligned}$$

where $C = 24(\frac{\sqrt{\log 2}}{2}+\frac{\sqrt{\pi}}{2})$.

Finally, substitute the above bound of empirical Rademacher complexity into Corollary 5, Theorem 5 is shown. □

4.7.2 Proof of Theorem 6

Proof. Let $\nabla h_{ij} = \rho_{\mathcal{H}_{\boldsymbol{\theta}}}(h(\cdot;\boldsymbol{\theta}_i), h(\cdot;\boldsymbol{\theta}_j))$, $\nabla\theta_{ij} = \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_j\|_2$ and $\nabla\theta_{ij,[k]} = \|\theta_{i,[k]} - \theta_{j,[k]}\|_2$, $\theta_i = (\theta_{i,[1]}, \ldots, \theta_{i,[K]})$. Then

$$\forall\boldsymbol{\theta}_i, \boldsymbol{\theta}_j \in \Theta, \boldsymbol{\theta}_i \neq \boldsymbol{\theta}_j \quad L = \frac{\nabla h_{ij}}{\nabla\theta_{ij}} \leq \frac{\sum_k L_{[k]}\nabla\theta_{ij,[k]}}{\nabla\theta_{ij}} = \frac{\sum_k L_{[k]}\nabla\theta_{ij,[k]}}{\sqrt{\sum_k(\nabla\theta_{ij,[k]})^2}},$$

where the first inequality is based on the definition of L_k. This is because

$$\begin{aligned}
\nabla h_{ij} = \quad & \max_{\boldsymbol{z}} \Big(h(\boldsymbol{z}; \theta_{1,[1]}, \dots, \theta_{1,[K]}) - h(\boldsymbol{z}; \theta_{2,[1]}, \dots, \theta_{2,[K]} \Big) \\
(a) \quad = \quad & \max_{\boldsymbol{z}} \Big| (h(\boldsymbol{z}; \theta_{1,[1]}, \theta_{1,[2]}, \dots, \theta_{1,[K]}) - h(\boldsymbol{z}; \theta_{2,[1]}, \theta_{1,[2]}, \dots, \theta_{1,[K]}) + \\
& (h(\boldsymbol{z}; \theta_{2,[1]}, \theta_{1,[2]}, \dots, \theta_{1,[K]}) - h(\boldsymbol{z}; \theta_{2,[1]}, \theta_{2,[2]}, \theta_{1,[3]} \dots, \theta_{1,[K]}) + \\
& \vdots \\
& (h(\boldsymbol{z}; \theta_{2,[1]}, \dots, \theta_{2,[K-1]}, \theta_{1,[K]}) - h(\boldsymbol{z}; \theta_{2,[1]}, \dots, \theta_{2,[K]}) \Big| \\
(b) \quad \leq \quad & \max_{\boldsymbol{z}} \Big| (h(\boldsymbol{z}; \theta_{1,[1]}, \theta_{1,[2]}, \dots, \theta_{1,[K]}) - h(\boldsymbol{z}; \theta_{2,[1]}, \theta_{1,[2]}, \dots, \theta_{1,[K]}) \Big| + \\
& \max_{\boldsymbol{z}} \Big| (h(\boldsymbol{z}; \theta_{2,[1]}, \theta_{1,[2]}, \dots, \theta_{1,[K]}) - h(\boldsymbol{z}; \theta_{2,[1]}, \theta_{2,[2]}, \theta_{1,[3]} \dots, \theta_{1,[K]}) \Big| + \\
& \vdots \\
& \max_{\boldsymbol{z}} \Big| (h(\boldsymbol{z}; \theta_{2,[1]}, \dots, \theta_{2,[K-1]}, \theta_{1,[K]}) - h(\boldsymbol{z}; \theta_{2,[1]}, \dots, \theta_{2,[K]}) \Big| \\
(c) \quad \leq \quad & \max_{\boldsymbol{z}, \theta_{[i]}, i \in [K] \setminus 1} \Big| (h(\boldsymbol{z}; \theta_{1,[1]}, \theta_{[2]}, \dots, \theta_{[K]}) - h(\boldsymbol{z}; \theta_{2,[1]}, \theta_{[2]}, \dots, \theta_{[K]}) \Big| + \\
& \max_{\boldsymbol{z}, \theta_{[i]}, i \in [K] \setminus 2} \Big| (h(\boldsymbol{z}; \theta_{[1]}, \theta_{1,[2]}, \theta_{[3]}, \dots, \theta_{[K]}) - h(\boldsymbol{z}; \theta_{[1]}, \theta_{2,[2]}, \theta_{[3]}, \dots, \theta_{[K]}) \Big| + \\
& \vdots \\
& \max_{\boldsymbol{z}, \theta_{[i]}, i \in [K] \setminus K} \Big| (h(\boldsymbol{z}; \theta_{[1]}, \dots \theta_{[K-1]}, \theta_{1,[K]}) - h(\boldsymbol{z}; \theta_{[1]}, \dots, \theta_{[K-1]} \theta_{2,[K]}) \Big| \\
(d) \quad = \quad & \sum_k L_{[k]} \nabla \theta_{ij,[k]},
\end{aligned}$$

where equality (a) is because we add and delete the same terms; inequality (b) is because $|a + b| \leq |a| + |b|$ and $\max_z(a(z) + b(z)) \leq \max_z a(z) + \max_z b(z)$; inequality (c) is due to the definition of $\max$; and equality (d) is due to the definition of $L_{[k]}$, that is, (4.5).

The maximum value of $\frac{\sum_k L_{[k]} \nabla \theta_{ij,[k]}}{\sqrt{\sum_k (\nabla \theta_{ij,[k]})^2}}$ can be solve by the following optimisation problem

$$\begin{aligned}
\max_{\nabla \theta_{[k]}, k=1,\dots,K} \quad & \textstyle\sum_k L_{[k]} \nabla \theta_{[k]} \\
s.t. \quad & \sqrt{\textstyle\sum_k (\nabla \theta_{[k]})^2} \leq 1
\end{aligned} \tag{4.9}$$

Let $s_{[k]} = \nabla\theta_{[k]}$, the above optimisation problem is the same as

$$\begin{aligned} \max_{s_{[k]}} \quad & \textstyle\sum_k L_{[k]} s_{[k]} \\ s.t. \quad & \textstyle\sum_k (s_{[k]})^2 \leq 1 \end{aligned} \tag{4.10}$$

By introducing the Lagrange multiplier α, we have

$$\mathcal{L} = \sum_k L_{[k]} s_{[k]} - \alpha(\sum_k (s_{[k]})^2 - 1).$$

Based on the stationary condition

$$\frac{\partial \mathcal{L}}{\partial s_i} = L_{[k]} - \alpha s_{[k]} = 0, \quad k \in [K],$$

we obtain

$$\alpha = \frac{L_{[k]}}{s_{[k]}}, \quad k \in [K].$$

Based on the above equation and the constraint of $\sum_k (s_{[k]})^2 \leq 1$, we obtain the solution

$$\nabla\theta_{[k]} = s_{[k]} = \frac{L_{[k]}}{\sqrt{\sum_k L_{[k]}^2}}.$$

Thus,

$$\begin{aligned} \frac{\sum_k L_{[k]} \nabla\theta_{ij,[k]}}{\sqrt{\sum_k (\nabla\theta_{ij,[k]})^2}} &= \max_{\nabla\theta_{[k]}} \sum_k L_{[k]} \nabla\theta_{[k]} \text{ s.t } \sqrt{\sum_k (\nabla\theta_{ij,[k]})^2} \leq 1 \\ &= \sqrt{\sum_k L_{[k]}^2} = \|\boldsymbol{L}\|_2. \end{aligned}$$

□

Chapter 5

Smooth Metric Leaning with Instance Extraction

5.1 Introduction

As we have presented in Section 4.1, the nearest neighbour (NN) classifier needs appropriate distance metrics and is better to be combined with instance compression for model learning. Hence in Chapter 4, we propose the metric learning with the instance extraction algorithm and discuss its learnability.As illustrated in the references, such as [22, 47, 48], it is important for the final learning bound to cover the factors related to the optimisation algorithms, because the generalisation ability is closely linked with the optimisation algorithm. The complexity measure could help explain some observations with respect to the optimisation problem, such as the effect of the training time [22], the effect of the smoothness of the operator [22] and the effect of the number of parameters [48].

The rest of this chapter is organised as follows. In section 5.2, we discuss the learnability of the classifier based on the generalisation PAC bound. The generalisation PAC bound is defined and proved to be a sufficient condition for agnostic PAC learnablility in Section 5.2.2. We explain how to obtain this bound in Section 5.2.3 and give a detailed example on the learnability of the gradient descent algorithm in Section 5.2.4. The generalisation PAC bound covers factors in the optimisation algorithm and suggests that the Lipschitz smooth property, that is Lipschitz continuous for the gradient, is important to the generalisation ability. Consequently, in Section 5.3, we propose a smooth metric learning with instance extraction (MLIE) algorithm using the Lipschitz smooth classifier and loss function. In Section 5.4, the proposed algorithm is compared with state-of-the-art competitors on 12 publicly available data sets and shows encouraging results. The work is concluded in Section 5.5 and theoretical proofs are deferred to Appendix 5.6.

5.2 Learning Bounds

5.2.1 Notations

Let $h(\boldsymbol{x}, \boldsymbol{w})$ be a function with input $\boldsymbol{x}$ and parameter $\boldsymbol{w}$ and the output is restricted to be a real value in the binary classification case. Let $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^D$, where $\mathcal{X}$ denotes a set which contains all possible values of $\boldsymbol{x}$. Let $\boldsymbol{w} = (w_{[1]}, \ldots, w_{[Q]}) \in \mathcal{W} \subseteq \mathbb{R}^Q$ be the parameter of the classifier, where $\mathcal{W}$ denotes a set which contains all possible values of $\boldsymbol{w}$. $y = \text{sign}\left[h(\boldsymbol{x}, \boldsymbol{w})\right]$ returns the classification result of input $\boldsymbol{x}$ given parameters $\boldsymbol{w}$, where $y \in \mathcal{Y} = \{-1, 1\}$ and $\text{sign}[\cdot]$ is the sign function[1].

Suppose the input $\boldsymbol{x}$ is a random variable distributed according to an *unknown* distribution with probability density function (PDF) $f(\boldsymbol{x})$. Let $\boldsymbol{x}^n = \{\boldsymbol{x}_i, i = 1, \ldots, n\}$ denote a set of n independent and identically (i.i.d.) distributed instances sampled from $f(\boldsymbol{x})$. Let $\boldsymbol{y}^n = \{y_i, i = 1, \ldots, n\}$ denote the label set, where y_i denote the corresponding label of $\boldsymbol{x}_i$. $P(y = 1|\boldsymbol{x})$ and $P(y = 0|\boldsymbol{x})$ follow an underlying distribution but is *unknown*. Let $\boldsymbol{z}^n = \{\ (\boldsymbol{x}_i, y_i), i = 1, \ldots, n\}$ denote the set of training instance and label pairs. Based on the assumptions on $\boldsymbol{x}^n$ and $\boldsymbol{y}^n$, $\boldsymbol{z}^n$ are n i.i.d. training pairs sampled from $p(\boldsymbol{z}) = p(\boldsymbol{x}, y)$ and $p(\boldsymbol{z})$ is *unknown*.

During the training process of the classifier, given $\boldsymbol{x}^n$ and $\boldsymbol{y}^n$, $\hat{h}$ can be obtained from optimisation algorithms, such as gradient descent (GD). $R_n(\boldsymbol{z}^n, \hat{h}) := \frac{1}{n}\sum_i r(\boldsymbol{z}_i, \hat{h}) := \frac{1}{n}\sum_i l(\hat{h}(\boldsymbol{x}_i); y_i)$ is called the *training error* or *empirical risk* and it indicates the training loss given the hypothesis returned by the algorithm, where $l(\cdot;\cdot)$ denotes the loss function and $r(\cdot,\cdot)$ denotes the risk function. Let $\boldsymbol{s}$ denote the fixed setting of the algorithm, such as the initial values, the number of

[1] $\text{sign}[a] = 1$ if $a \geq 0$ and $\text{sign}[a] = -1$ if $a < 0$, where $a \in \mathbb{R}$.

iterations and the step size. With a parametric classifier, $\hat{\boldsymbol{w}}$ will be used to represent $\hat{h}$. The relationship between $\boldsymbol{w}$ and $\boldsymbol{x}^n, \boldsymbol{y}^n$ is represented as $\hat{\boldsymbol{w}} = \boldsymbol{m}(\boldsymbol{z}^n; \boldsymbol{s})$, where $\boldsymbol{m} : \mathcal{X} \times \mathcal{S} \to \Theta$ denotes a function from $\boldsymbol{x}$ and $\boldsymbol{s}$ to the learned parameters $\boldsymbol{\theta}$. Since $\hat{\boldsymbol{w}}$ is a function of the random variables $\boldsymbol{x}^n, \boldsymbol{y}^n$, $\hat{\boldsymbol{w}}$ is also a random variable. In the parametric case, the training error would be represented as $R_n(\boldsymbol{z}^n, \hat{\boldsymbol{w}}) := \frac{1}{n}\sum_i r(\boldsymbol{z}_i, \hat{\boldsymbol{w}}) := \frac{1}{n}\sum_i l(h(\boldsymbol{x}_i, \hat{\boldsymbol{w}}), y_i)$.

During the test process of the classifier, test pair $\boldsymbol{z}' = (\boldsymbol{x}', y')$ is sampled from the same unknown distribution $p(\boldsymbol{z})$. The predicted value $h(\boldsymbol{x}', \hat{h})$ would be compared with the test label to evaluate the performance of the algorithm. $R(\hat{h}) := \mathbb{E}_{\boldsymbol{z}'} r(\boldsymbol{z}', \hat{h}) := \mathbb{E}_{\boldsymbol{z}'} l(h(\boldsymbol{x}', \hat{h}); y')$ is called the *expected error* or *test error* and it indicates the expected value of test loss given the classifier $\hat{h}$. With a parametric classifier, the following notations would be used $R(\hat{\boldsymbol{w}}) := \mathbb{E}_{\boldsymbol{z}'} r(\boldsymbol{z}', \hat{\boldsymbol{w}}) := \mathbb{E}_{\boldsymbol{z}'} l(h(\boldsymbol{x}', \hat{\boldsymbol{w}}); y')$.

The gap between the training error and test error, i.e. $R(\hat{\boldsymbol{w}}) - R_n(\boldsymbol{z}^n, \hat{\boldsymbol{w}})$, is called the *generalisation gap*. A good classifier should have small training error and small generalisation gap so as to perform well on test instances.

5.2.2 Learnablility with the Generalisation PAC bounds

Definition 35. A hypothesis class $\mathcal{H}$ has the *generalisation PAC bound* if there exists a function $n^G_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$ such that for every $\epsilon, \delta \in (0,1)$ and for every probability distribution $\mathcal{D}_{\mathcal{Z}}$ over $\mathcal{Z}$, if $\boldsymbol{z}^n$ is a sample of $n \geq n^G_{\mathcal{H}}(\epsilon, \delta)$ i.i.d. examples drawn from $D_{\mathcal{Z}}$, the algorithm returns a hypothesis $\hat{h}$ such that the following inequality is satisfied

$$\mathbb{P}_{\boldsymbol{z}^n}\Big(R(\hat{h}) - R_n(\boldsymbol{z}^n, \hat{h}) \leq \epsilon\Big) \geq 1 - \delta. \tag{5.1}$$

Instead of the uniform bound of generalisation gap used in uniform convergence, this definition considers the probability of a small generalisation gap ($R(\hat{h}) - R_n(\boldsymbol{z}^n, \hat{h}) \leq \epsilon$) directly. It has been used in the research of stability [6]. The condition is weaker than the uniform convergence, as illustrated in Lemma 9, but is still a sufficient condition for the agnostic PAC learnability, as illustrated in

Theorem 10.

Lemma 9. The relationship between the proposed bound and the uniform convergence bound is as follows:

$$\mathbb{P}_{\boldsymbol{z}^n}\Big(R(\hat{h}) - R_n(\boldsymbol{z}^n, \hat{h}) \leq \epsilon\Big) \geq \mathbb{P}_{\boldsymbol{z}^n}\Big(\max_{h \in \mathcal{H}}[R(h) - R_n(\boldsymbol{z}^n, h)] \leq \epsilon\Big). \tag{5.2}$$

Theorem 10. Suppose $\mathrm{ERM}_{\mathcal{H}}$ exists for a class $\mathcal{H}$, where $\mathrm{ERM}_{\mathcal{H}}$ denotes the empirical risk minimisation strategy inside the set $\mathcal{H}$. If $\mathcal{H}$ has the generalisation PAC bound with a function $n^G_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$, then $\mathcal{H}$ is agnostic PAC learnable with the sample complexity function $n^{AL}_{\mathcal{H}}(\epsilon, \delta) \leq \max[n^G_{\mathcal{H}}(\epsilon/2, \delta/2), \frac{2C_r^2}{\epsilon^2} \ln \frac{4}{\delta}]$, where the range of the risk function $r(\boldsymbol{z}, h)$ is bounded by $[0, C_r]$. Furthermore, in this case, $\mathrm{ERM}_{\mathcal{H}}$ is a successful agnostic PAC learner for $\mathcal{H}$.

5.2.3 Decomposition

In this section, we will discuss how to obtain the generalisation PAC bound for the parametric hypothesis space, that is, h is specified by a set of parameters $\boldsymbol{w}$. One simple way is to use the uniform convergence bound in a much smaller set.

Theorem 11. (Decomposition Theorem) Let $\mathcal{W}$ denote the set of all possible values of $\boldsymbol{w}$ and $\hat{\mathcal{W}} \subseteq \mathcal{W}$; let $\delta_1, \delta_2 \geq 0$. If

$$\mathbb{P}_{\boldsymbol{z}^n}[\boldsymbol{m}(\boldsymbol{z}^n) \in \hat{\mathcal{W}}] \geq 1 - \delta_1 \tag{5.3}$$

and

$$\mathbb{P}_{\boldsymbol{z}^n}[\max_{\boldsymbol{w} \in \hat{\mathcal{W}}} R(\boldsymbol{w}) - R_n(\boldsymbol{z}^n, \boldsymbol{w}) \leq \epsilon] \geq 1 - \delta_2, \tag{5.4}$$

then, the following inequality holds

$$\mathbb{P}_{\boldsymbol{z}^n}[R(\hat{\boldsymbol{w}}) - R_n(\boldsymbol{z}^n, \hat{\boldsymbol{w}}) \leq \epsilon] \geq 1 - \delta_1 - \delta_2. \tag{5.5}$$

Theorem 11 decomposes the generalisation PAC bound, i.e. (5.5), into two terms which are easier to consider, namely (1) A set $\hat{\mathcal{W}}$ which includes $1 - \delta_1$ cases of $\boldsymbol{m}(\boldsymbol{z}^n)$, i.e. (5.3); (2) Uniform convergence of the generalisation error inside $\hat{\mathcal{W}}$, i.e. (5.4). In the following section, we will (1) bound δ_1 by considering the concentration property of $\boldsymbol{m}(\boldsymbol{z}^n)$; (2) bound δ_2 with the uniform convergence results.

5.2.4 Analyse Learnability of the Gradient Descent Algorithm

5.2.4.1 Settings

In the following section, learnability of the gradient descent (GD) algorithm will be discussed. The updating equation of the most conventional GD algorithm is as follows:

$$\hat{\boldsymbol{w}}^{(1)} = \boldsymbol{w}^{(0)} - \frac{\alpha^{(1)}}{n}\sum_{i=1}^{n}\frac{\partial r(\boldsymbol{z}_i, \boldsymbol{w})}{\partial \boldsymbol{w}}|_{\hat{\boldsymbol{w}}^{(0)}};$$

$$\vdots$$

$$\hat{\boldsymbol{w}}^{(T)} = \hat{\boldsymbol{w}}^{(T-1)} - \frac{\alpha^{(T)}}{n}\sum_{i=1}^{n}\frac{\partial r(\boldsymbol{z}_i, \boldsymbol{w})}{\partial \boldsymbol{w}}|_{\hat{\boldsymbol{w}}^{(T-1)}};$$

where $\alpha^{(t)} \geq 0$ denotes the learning rate at time t; $\hat{\boldsymbol{w}}^{(t)}$ denotes the estimated parameters of the classifier obtained after t iterations; $\boldsymbol{w}^{(0)}$ denotes the initial parameter of the algorithm; $r(\boldsymbol{z}_i, \boldsymbol{w}) = l(h(\boldsymbol{x}_i, \boldsymbol{w}), y_i)$ indicates the training error of the ith training instance given parameter $\boldsymbol{w}$. Here the number of iteration T and the learning rate $\alpha^{(t)}$ are treated as the setting parameters of the gradient descent algorithm and determined in advance, i.e. $\boldsymbol{s} = \{T, \alpha^{(t)}, t = 1, \dots, T\}$. The initial weight $\boldsymbol{w}^{(0)}$ is assumed to be fixed.

Based on Theorem 11, we need δ_1 and δ_2 to obtain the final learning bound:

$$\mathbb{P}_{\boldsymbol{z}^n}[\boldsymbol{m}(\boldsymbol{z}^n) \in \hat{\mathcal{W}}] \geq 1 - \delta_1,$$

$$\mathbb{P}_{\boldsymbol{z}^n}[\max_{\boldsymbol{w}\in\hat{\mathcal{W}}} R(\boldsymbol{w}) - R_n(\boldsymbol{z}^n, \boldsymbol{w}) \leq \epsilon] \geq 1 - \delta_2.$$

Based on Theorem 13 introduced in the Appendix, δ_2 could be bounded so long as, (i) the Lipschitz constant $\mathrm{lip}(h \leftarrow \boldsymbol{w})$ could be uniformly bounded in the whole space of $\hat{\mathcal{W}}$ and (ii) $\mathrm{diam}(\hat{\mathcal{W}}, \|\cdot\|_2)$ could be bounded. (i) $\mathrm{lip}(h \leftarrow \boldsymbol{w})$ could be bounded as long as h has a bounded gradient with respect to $\boldsymbol{w} \in \hat{\mathcal{W}}$, which is not a strict condition when $\mathrm{diam}(\hat{\mathcal{W}}, \|\cdot\|_2)$ could be bounded. (ii) To bound

$\mathrm{diam}(\hat{\mathcal{W}}, \|\cdot\|_2)$, $\hat{\mathcal{W}}$ is set as an Euclidean ball around $\mathbb{E}_{\boldsymbol{z}^n}\boldsymbol{m}(\boldsymbol{z}^n)$ and concentration techniques will be applied to guarantee $\mathbb{P}_{\boldsymbol{z}^n}\Big[\boldsymbol{m}(\boldsymbol{z}^n) \in \text{ball}\left(\mathbb{E}_{\boldsymbol{z}^n}\boldsymbol{m}^{(T)}(\boldsymbol{z}^n), \epsilon\right)\Big] \leq 1 - \delta_1$, where δ_1 will decrease to 0 when $n \to \infty$.

5.2.4.2 Concentration of $\hat{\boldsymbol{w}}^{(T)}$

In this section, the discussion holds for any $\boldsymbol{w}^{(0)}$. The mappings of $\boldsymbol{m}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s})$ and $\boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s})$ are defined as follows:

$$\boldsymbol{m}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s}) := \hat{\boldsymbol{w}}^{(T)} = \boldsymbol{w}^{(0)} - \sum_{t=1}^{T} \alpha^{(t)} \Big(\sum_{j=1}^{n} \frac{1}{n} \frac{\partial r(\boldsymbol{z}_i, \boldsymbol{w})}{\partial \boldsymbol{w}}\Big)|_{\hat{\boldsymbol{w}}^{(t-1)}};$$

$$\boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s}) := \hat{\boldsymbol{w}}_{[q]}^{(T)} = \left[\boldsymbol{w}^{(0)} - \sum_{t=1}^{T} \alpha^{(t)} \Big(\sum_{j=1}^{n} \frac{1}{n} \frac{\partial r(\boldsymbol{z}_i, \boldsymbol{w})}{\partial \boldsymbol{w}}\Big)|_{\hat{\boldsymbol{w}}^{(t-1)}}\right]_{[q]}; \quad (5.6)$$

where $\boldsymbol{v}_{[q]}$ denotes the qth value of a vector $\boldsymbol{v}$. With a fixed setting $\boldsymbol{s}$ and fixed $\boldsymbol{w}^{(0)}$, given the value of $\boldsymbol{z}^n$, the value of $\boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s})$ is determined. In other words, $\boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s})$ is a function from $\mathcal{Z}^n$ to $\mathbb{R}$. The McDiarmid's inequality can be used to obtain the concentration properties after a function mapping. Based on the McDiarmid's inequality (Lemma 11 in the Appendix), we can obtain the concentration property of $\boldsymbol{m}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s})$ as shown in the lemma below.

Definition 36. A operator $G : \mathcal{W} \to \mathcal{W}$ is called *η-expansive* if

$$\max_{\boldsymbol{w}_1, \boldsymbol{w}_2 \in \mathcal{W}, \boldsymbol{w}_1 \neq \boldsymbol{w}_2} \frac{\|G(\boldsymbol{w}_1) - G(\boldsymbol{w}_2)\|}{\|\boldsymbol{w}_1 - \boldsymbol{w}_2\|} \leq \eta.$$

If $\eta \leq 1$, then the operator G is *non-expansive*.

Lemma 10. The following bound holds for $\boldsymbol{s}$ and all value of $\boldsymbol{a}$,

$$\mathbb{P}_{\boldsymbol{z}^n}\Big[\|\boldsymbol{m}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s}) - \mathbb{E}_{\boldsymbol{z}^n}\boldsymbol{m}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s})\| \leq \epsilon\Big] \geq 1 - Q\exp(\frac{-2\epsilon^2 n}{QC^2}).$$

where $C = 2\Big(\sum_{t=1}^{T} \eta^{T-t}\alpha^{(t)}\Big)\,\mathrm{lip}\,(r \leftarrow \boldsymbol{w})$; η is the Lipschitz constant of operator G with respect to $\boldsymbol{w}$ and $G(\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n)) = \boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n) + \sum_{j\in[n]/i} \frac{\alpha^{(t)}}{n} \frac{\partial r(\boldsymbol{z}_j, \boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n)}$; $[n]/i$ denotes the set which contains the integers from 0 to n without i.

5.2.4.3 Using the Decomposition Theorem

Corollary 7. Let $\mathrm{ball}(\boldsymbol{o}, \boldsymbol{r})$ denote a ball with the centre at $\boldsymbol{o}_s$ and radius of r_s. The following inequality is satisfied for all $\epsilon > 0$

$$\mathbb{P}_{\boldsymbol{z}^n}\Big[\boldsymbol{m}^{(T)}(\boldsymbol{z}^n) \in \mathrm{ball}\left(\mathbb{E}_{\boldsymbol{z}^n}\boldsymbol{m}^{(T)}(\boldsymbol{z}^n), B\right)\Big] \geq 1 - Q\exp(\frac{-2B^2n}{QC^2}), \tag{5.7}$$

where $C = 2\Big(\sum_{t=1}^{T}\eta^{T-t}\alpha^{(t)}\Big)\ \mathrm{lip}\,(r \leftarrow \boldsymbol{w})$, $B > 0$ is a positive constant.

Proof. Based on Lemma 10,

$$\begin{aligned}
&\mathbb{P}_{\boldsymbol{z}^n}\Big[\|\boldsymbol{m}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s}) - \mathbb{E}_{\boldsymbol{z}^n}\boldsymbol{m}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s})\| \leq B\Big] \geq 1 - Q\exp(\frac{-2B^2n}{QC^2}) \\
\Leftrightarrow\quad &\mathbb{P}_{\boldsymbol{z}^n}\Big[\boldsymbol{m}^{(T)}(\boldsymbol{z}^n) \in \mathrm{ball}\left(\mathbb{E}_{\boldsymbol{z}^n}\boldsymbol{m}^{(T)}(\boldsymbol{z}^n), B\right)\Big] \geq 1 - Q\exp(\frac{-2B^2n}{QC^2})
\end{aligned}$$

□

Theorem 12. Suppose $\mathrm{lip}(h \leftarrow \boldsymbol{w}) \leq L_1$ and $\mathrm{lip}(r \leftarrow h) \leq L_l$, the following inequality holds

$$\mathbb{P}_{\boldsymbol{z}^n}[R(\boldsymbol{m}(\boldsymbol{z}^n)) - R_n(\boldsymbol{z}^n, \boldsymbol{m}(\boldsymbol{z}^n)) \leq \epsilon] \geq 1 - \delta_1 - \delta_2,$$

where

$$\epsilon = \frac{C_1C_2L_1^2L_l^2Q\sqrt{\frac{1}{2}\ln(Q/\delta_1)}}{n} + \frac{\sqrt{\frac{1}{2}\ln(1/\delta_2)}}{\sqrt{n}}$$

and $\boldsymbol{w} \in \mathbb{R}^Q$; C_1 is a universal constant; $C_2 = 2\Big(\sum_{t=1}^{T}\eta^{T-t}\alpha^{(t)}\Big)$; η is the Lipschitz constant of operator G with respect to $\boldsymbol{w}$ and $G(\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n)) = \boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n) - \sum_{j\in[n]/i}\frac{\alpha^{(t)}}{n}\frac{\partial r(\boldsymbol{z}_j,\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n)}$, $[n]/i$ denotes the set which contains the integers from 0 to n without i.

The above statement is equivalent to the follows: with probability at least $1-\delta_1-\delta_2$, the following bound holds

$$R(\boldsymbol{m}(\boldsymbol{z}^n, \boldsymbol{a})) - R_n(\boldsymbol{z}^n, \boldsymbol{m}(\boldsymbol{z}^n, \boldsymbol{a})) \leq \frac{C_1C_2L_1^2L_l^2Q\sqrt{\frac{1}{2}\ln(Q/\delta_1)}}{n} + \frac{\sqrt{\frac{1}{2}\ln(1/\delta_2)}}{\sqrt{n}}. \tag{5.8}$$

Based on Theorem 12, the following factors are considered:

(1) A small value of Q will give a tighter bound. In other words, the generalisation gap will be smaller with less number of parameters;

(2) The Lipschitz constants $\text{lip}(h \leftarrow \boldsymbol{w})$ and $\text{lip}(r \leftarrow h)$ will affect the learning bound. It is reasonable to select a loss function with smaller L_l and a classifier with smaller L_1.

(3) η, i.e. $\text{lip}(G \leftarrow \boldsymbol{w})$, also appears in the bound. Based on the definition of G and the addition property of Lipschitz functions, if the lipschitz constant of $\frac{\partial r(\boldsymbol{z}_j,\boldsymbol{w})}{\partial \boldsymbol{w}}$ with respect to $\boldsymbol{w}$ is bounded by L_s, then $\text{lip}(G \leftarrow \boldsymbol{w})$ is bounded by $1+\alpha L_s$. $\text{lip}(\frac{\partial r(\boldsymbol{z}_j,\boldsymbol{w})}{\partial \boldsymbol{w}} \leftarrow \boldsymbol{w})$ is called (Lipschitz) smooth as illustrated by the following definition.

Definition 37. A risk function $r(\boldsymbol{z}, \boldsymbol{w})$ is called *η-smooth*[2] (with respect to $\boldsymbol{w}$), if $\forall \boldsymbol{w}_1, \boldsymbol{w}_2 \in \mathcal{W}$, $\boldsymbol{z} \in \mathcal{Z}$,

$$\|\frac{\partial r(\boldsymbol{z},\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{w}_1} - \frac{\partial r(\boldsymbol{z},\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{w}_2}\| \leq \eta \|\boldsymbol{w}_1 - \boldsymbol{w}_2\|.$$

Compared with Corollary 4 in Chapter 4, Theorem 12 has an additional requirement that $\text{lip}(G \leftarrow \boldsymbol{w})$ should be bounded. A sufficient condition for $\eta = \text{lip}(G \leftarrow \boldsymbol{w})$ being bounded is $\text{lip}(\frac{\partial r}{\partial w} \leftarrow \boldsymbol{w})$ being bounded.

$$\text{lip}(\frac{\partial r}{\partial \boldsymbol{w}} \leftarrow \boldsymbol{w}) = \text{lip}(\frac{\partial r}{\partial h}\frac{\partial h}{\partial \boldsymbol{w}} \leftarrow \boldsymbol{w}) = \text{lip}(g_l \circ g_h \leftarrow \boldsymbol{w}),$$

where $\circ$ denotes the composition of functions, $g_l = \frac{\partial r}{\partial h}$ and $g_h = \frac{\partial h}{\partial \boldsymbol{w}}$. Based on the composition property of Lipschitz functions illustrated in Lemma 2(d), if the Lipschitz constant of g_l and g_h are bounded, $\text{lip}(g_l \circ g_h \leftarrow \boldsymbol{w})$ would be bounded. Therefore, as long as we have smooth loss, i.e. bounded $\text{lip}(\frac{\partial r}{\partial h} \leftarrow h)$, and smooth classifier, i.e. bounded $\text{lip}(\frac{\partial h}{\partial \boldsymbol{w}} \leftarrow \boldsymbol{w})$, $\text{lip}(G \leftarrow \boldsymbol{w})$ is bounded.

[2] Smooth is also called Lipschitz continuous gradient and Lipschitz smooth in some references.

5.3 Classifier and Optimisation

5.3.1 Classifier

Based on the bounds in the previous section, smoothness is also an important concept for generalisation and a smooth classifier with smooth loss function may enjoy good generalisation performance. Therefore, a smooth classifier is proposed and following Chapter 4, metric learning with instance extraction (MLIE) strategy has been adopted in order to extract representative instances and learn a suitable distance metric from data at the same time:

$$h(\boldsymbol{x}) = \sum_i \exp(-\gamma d^2_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{r}_i^+)) - \sum_j \exp(-\gamma d^2_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{r}_j^-)), \tag{5.9}$$

where $d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{M} (\boldsymbol{x}_i - \boldsymbol{x}_j)}$ denotes the Mahalanobis distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, $\boldsymbol{M} \in \boldsymbol{M}_+$ is the parameter matrix and $\boldsymbol{M}_+$ denotes the set of positive semi-definite matrices; γ controls the rate of the decay of the influence of the relatively far away distances; $\boldsymbol{r}_i^+$ and $\boldsymbol{r}_i^-$ denote the ith extracted positive and negative class instance respectively.

Based on the definition of the classifier, the extracted instances which lie closer to $\boldsymbol{x}$ have a larger impact on the classification result. With an increase in the distance, the influence on the result decays at an exponential rate. In the above formula, the parameter γ needs to be set in advance. To avoid this step, the following equivalent formula will be used

$$\begin{aligned} h(\boldsymbol{x}) &= \sum_i \exp(-d^2(\boldsymbol{L}\boldsymbol{x}, \boldsymbol{L}\boldsymbol{r}_i^+)) - \sum_j \exp(-d^2(\boldsymbol{L}\boldsymbol{x}, \boldsymbol{L}\boldsymbol{r}_i^-)) \\ &= \sum_i \exp(-\|\boldsymbol{L}\boldsymbol{x} - \boldsymbol{L}\boldsymbol{r}_i^+\|^2) - \sum_j \exp(-\|\boldsymbol{L}\boldsymbol{x} - \boldsymbol{L}\boldsymbol{r}_j^-\|^2), \end{aligned}$$

where $d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T (\boldsymbol{x}_i - \boldsymbol{x}_j)}$ denotes the Euclidean distance and $\|\cdot\|$ denotes the vector L_2-norm. $\boldsymbol{L}$ denotes a linear mapping and $\boldsymbol{L}^T\boldsymbol{L} = \boldsymbol{M}$. If we would like to learn the extracted instance after linear mapping, the following classifier may be used

$$\begin{aligned} h(\boldsymbol{x}) &= \sum_i \exp(-d^2(\boldsymbol{L}\boldsymbol{x}, \boldsymbol{r}_i^+)) - \sum_j \exp(-d^2(\boldsymbol{L}\boldsymbol{x}, \boldsymbol{r}_j^-)) \\ &= \sum_i \exp(-\|\boldsymbol{L}\boldsymbol{x} - \boldsymbol{r}_i^+\|^2) - \sum_j \exp(-\|\boldsymbol{L}\boldsymbol{x} - \boldsymbol{r}_j^-\|^2). \end{aligned} \tag{5.10}$$

For a classifier h with convex constraints on parameters, the parameter $\boldsymbol{w}$ will be restricted to be inside a convex set, as illustrated in Section 1.6.2. Then based on Corollary 1 in Chapter 1, a sufficient condition for bounded $\text{lip}(\frac{\partial h}{\partial \boldsymbol{w}} \leftarrow \boldsymbol{w})$ is to have finite values of the first and second partial derivatives. The first partial derivatives of the classifier (5.10) are as follows:

$$\begin{aligned} \frac{\partial h(\boldsymbol{x};\Theta)}{\partial \boldsymbol{r}_i^+} =& - \exp(-\|\boldsymbol{L}\boldsymbol{x} - \boldsymbol{r}_i^+\|^2)(2\boldsymbol{r}_i^+ - 2\boldsymbol{L}\boldsymbol{x}) \\ \frac{\partial h(\boldsymbol{x};\Theta)}{\partial \boldsymbol{r}_j^-} =& \exp(-\|\boldsymbol{L}\boldsymbol{x} - \boldsymbol{r}_j^-\|^2)(2\boldsymbol{r}_j^- - 2\boldsymbol{L}\boldsymbol{x}) \\ \frac{\partial h(\boldsymbol{x};\Theta)}{\partial \boldsymbol{L}_{[a,b]}} =& \sum_i -2(\boldsymbol{L}\boldsymbol{x} - \boldsymbol{r}_i^+)_{[a]}\boldsymbol{x}_{[b]} \exp(-\|\boldsymbol{L}\boldsymbol{x} - \boldsymbol{r}_i^+\|^2) \\ &+ \sum_j 2(\boldsymbol{L}\boldsymbol{x} - \boldsymbol{r}_j^-)_{[a]}\boldsymbol{x}_{[b]} \exp(-\|\boldsymbol{L}\boldsymbol{x} - \boldsymbol{r}_j^-\|^2), \end{aligned}$$

where $\boldsymbol{L}_{[i,j]}$ denotes the ith row and jth column element of matrix $\boldsymbol{L}$ and $\boldsymbol{x}_{[j]}$ denotes the jth element of the vector $\boldsymbol{x}$; $(\boldsymbol{L}\boldsymbol{x} - \boldsymbol{r})_{[a]} = \sum_k \boldsymbol{L}_{[ak]}\boldsymbol{x}_{[k]} - \boldsymbol{r}_{[a]}$. $\frac{\partial h(\boldsymbol{x};\Theta)}{\partial \boldsymbol{r}_i^+}$ and $\frac{\partial h(\boldsymbol{x};\Theta)}{\partial \boldsymbol{r}_j^-}$ are bounded by $2\,\text{diam}(\boldsymbol{r}_a^-) + 2\,\text{diam}(\boldsymbol{L})\,\text{diam}(\boldsymbol{x})$, where $\text{diam}(\mathcal{V}) = \sup_{\boldsymbol{v}_i,\boldsymbol{v}_j \in \mathcal{V}} \|\boldsymbol{v}_i - \boldsymbol{v}_j\|$ and vector-2 norm or Frobenius norm is used for a set of vectors or a set of matrices respectively; $\frac{\partial h(\boldsymbol{x};\Theta)}{\partial \boldsymbol{L}}$ is bounded by $4m(\text{diam}(\boldsymbol{r}) + \text{diam}(\boldsymbol{L})\,\text{diam}(\boldsymbol{x}))\,\text{diam}(\boldsymbol{x})$, where m denotes the number of extracted instances. All first partial derivatives have finite values as long as $\text{diam}(\boldsymbol{L})$,$\text{diam}(\boldsymbol{x})$ and $\text{diam}(\boldsymbol{r})$ are bounded.

The second partial derivatives are as follows:

$$
\begin{aligned}
\frac{\partial^2 h(\boldsymbol{x};\Theta)}{\partial \boldsymbol{r}_i^{+2}} =& \exp(-\|\boldsymbol{L}\boldsymbol{x}-\boldsymbol{r}_i^+\|^2)(2\boldsymbol{r}_i^+-2\boldsymbol{L}\boldsymbol{x})(2\boldsymbol{r}_i^+-2\boldsymbol{L}\boldsymbol{x})^T \\
&-2\exp(-\|\boldsymbol{L}\boldsymbol{x}-\boldsymbol{r}_i^+\|^2)\boldsymbol{I}; \\
\frac{\partial^2 h(\boldsymbol{x};\Theta)}{\partial \boldsymbol{r}_j^{-2}} =& -\exp(-\|\boldsymbol{L}\boldsymbol{x}-\boldsymbol{r}_j^-\|^2)(2\boldsymbol{r}_j^--2\boldsymbol{L}\boldsymbol{x})(2\boldsymbol{r}_j^--2\boldsymbol{L}\boldsymbol{x})^T \\
&+2\exp(-\|\boldsymbol{L}\boldsymbol{x}-\boldsymbol{r}_j^-\|^2)\boldsymbol{I}; \\
\frac{\partial^2 h(\boldsymbol{x};\Theta)}{\partial \boldsymbol{L}_{[a,b]}{}^2} =& \sum_i 4(\boldsymbol{L}\boldsymbol{x}-\boldsymbol{r}_i^+)^2_{[a]}\boldsymbol{x}^2_{[b]}\exp(-\|\boldsymbol{L}\boldsymbol{x}-\boldsymbol{r}_i^+\|^2) \\
&-2\sum_i \boldsymbol{x}^2_{[b]}\exp(-\|\boldsymbol{L}\boldsymbol{x}-\boldsymbol{r}_i^+\|^2) \\
&-\sum_j 4(\boldsymbol{L}\boldsymbol{x}-\boldsymbol{r}_j^-)^2_{[a]}\boldsymbol{x}^2_{[b]}\exp(-\|\boldsymbol{L}\boldsymbol{x}-\boldsymbol{r}_j^-\|^2) \\
&+2\sum_j \boldsymbol{x}^2_{[b]}\exp(-\|\boldsymbol{L}\boldsymbol{x}-\boldsymbol{r}_j^-\|^2),
\end{aligned}
$$

where $\boldsymbol{I}$ is the identity matrix. All second partial derivatives have finite values as long as $\mathrm{diam}(\boldsymbol{L})$,$\mathrm{diam}(\boldsymbol{x})$ and $\mathrm{diam}(\boldsymbol{r})$ are bounded.

5.3.2 Loss Function

Hinge loss is not second-order differentiable. Similar to Huber loss for regression [29], the following loss is defined by combining a quadratic and a linear function

$$
l(a) = \begin{cases} 1-a & \text{if } a \le 0 \\ \frac{1}{4}(a-2)^2 & \text{if } 0 < a \le 2 \\ 0 & \text{if } a > 2 \end{cases} \tag{5.11}
$$

The derivative of $l(a)$ is as follows

$$
l'(a) = \begin{cases} -1 & \text{if } a \le 0 \\ \frac{a-2}{2} & \text{if } 0 < a \le 2 \\ 0 & \text{if } a > 2 \end{cases}
$$

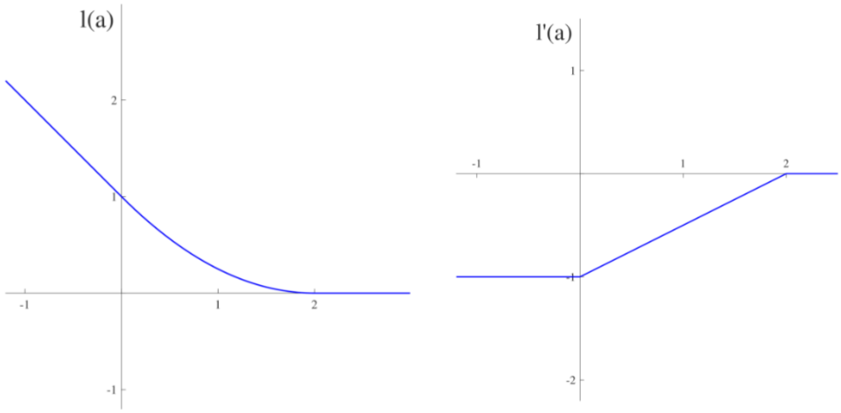

Figure 5.1: An illustration of the ‘smooth’ loss function and its derivative. Left: loss function $l(a)$; right: the derivative of loss function $l'(a)$.

The loss function and its derivative are illustrated in Figure 5.1. The Lipschitz constant of $l'(a)$ is bounded by $\frac{1}{2}$ and the proposed loss (5.11) is a smooth loss.

5.3.3 Objective Function

Using the classifier defined in (5.10), the loss function defined in (5.11) and the convex regularisation terms $\sum_i \|\boldsymbol{r}_i^-\|_2^2 + \|\boldsymbol{r}_j^+\|_2^2 + \|\boldsymbol{L}\|_F^2$, the following optimisation problem is proposed:

$$\begin{aligned} \min_{\Theta} \quad & \tfrac{1}{n} \textstyle\sum_k l(y_k h(\boldsymbol{x}_k; \boldsymbol{r}^m, \boldsymbol{L})) + \lambda \Big(\textstyle\sum_i \|\boldsymbol{r}_i^-\|_2^2 + \|\boldsymbol{r}_j^+\|_2^2 + \|\boldsymbol{L}\|_F^2 \Big) \\ & i = 1, \dots, n, \end{aligned} \tag{5.12}$$

where $\Theta = \{\boldsymbol{r}^m, \boldsymbol{L}\}$ denotes the set of parameters to be optimised; $\boldsymbol{L}$ denotes the linear mapping to be learned and the learning of $\boldsymbol{L}$ is equivalent to the learning of the Mahalanobis distance $\boldsymbol{M}$; $\boldsymbol{r}^m = \{\boldsymbol{r}_i^-, \boldsymbol{r}_j^+; i = 1, \dots, m_+, j = 1, \dots, m_-\}$ denotes the set of all extracted instances; λ is a trade-off parameter which balances the loss term and the regularisation term.

5.3.4 Gradient Descent

The optimisation function is not a convex problem due to the non-convexity of $h(\boldsymbol{x}; \boldsymbol{r}^m, \boldsymbol{M})$. Thus the gradient descent algorithm is used:

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

where α is the learning rate; the superscript t denotes the time step during optimisation; g denotes the objective function.

The gradient of the objective function g with respect to each set of parameters is

$$\frac{\partial g}{\partial \Theta}|_{\Theta^t} = \Big(\frac{1}{n}\sum_k y_k \frac{\partial l}{\partial h(\boldsymbol{x}_k;\Theta)}\frac{\partial h(\boldsymbol{x}_k;\Theta)}{\partial \Theta} + 2\lambda\Theta\Big)|_{\Theta^t}.$$

The final updating equations are as follows:

$$\begin{aligned}
\boldsymbol{r}_i^{+,t+1} =& \boldsymbol{r}_i^{+,t} - 2\lambda\alpha\boldsymbol{r}_i^{+,t} \\
&+\frac{\alpha}{n}\sum_{k=1}^{n} y_k l'[h(\boldsymbol{x}_k;\Theta)]\exp(-\|\boldsymbol{L}\boldsymbol{x}_k - \boldsymbol{r}_i^{+}\|^2)(2\boldsymbol{r}_i^{+} - 2\boldsymbol{L}\boldsymbol{x}_k)|_{\Theta^t}; \\
\boldsymbol{r}_j^{-,t+1} =& \boldsymbol{r}_j^{-,t} - 2\lambda\alpha\boldsymbol{r}_j^{-,t} \\
&-\frac{\alpha}{n}\sum_{k=1}^{n} y_k l'[h(\boldsymbol{x}_k;\Theta)]\exp(-\|\boldsymbol{L}\boldsymbol{x}_k - \boldsymbol{r}_j^{-}\|^2)(2\boldsymbol{r}_j^{-} - 2\boldsymbol{L}\boldsymbol{x}_k)|_{\Theta^t}; \\
\boldsymbol{L}^{t+1} =& \boldsymbol{L}^t - 2\lambda\alpha\boldsymbol{L}^t \\
&+\frac{\alpha}{n}\sum_{k=1}^{n} y_k l'[h(\boldsymbol{x}_k;\Theta)]\sum_i \exp(-\|\boldsymbol{L}\boldsymbol{x}_k - \boldsymbol{r}_i^{+}\|^2)2(\boldsymbol{L}\boldsymbol{x}_k - \boldsymbol{r}_i^{+})\boldsymbol{x}_k^T|_{\Theta^t} \\
&-\frac{\alpha}{n}\sum_{k=1}^{n} y_k l'[h(\boldsymbol{x}_k;\Theta)]\sum_j \exp(-\|\boldsymbol{L}\boldsymbol{x}_k - \boldsymbol{r}_j^{-}\|^2)2(\boldsymbol{L}\boldsymbol{x}_k - \boldsymbol{r}_j^{-})\boldsymbol{x}_k^T|_{\Theta^t}.
\end{aligned}$$

5.4 Experiments

The proposed algorithm is compared with nine established metric learning algorithms from two categories: 1) The most cited algorithms, including large margin nearest neighbor (LMNN) [76], information theoretic metric learning (ITML) [9], neighbourhood component analysis (NCA) [17] and metric learning by collapsing classes (MCML) [16]; (2) the most state-of-the-art algorithms, including geometric mean metric learning (GMML) [83], regressive virtual metric learning (RVML) [52], stochastic neighbor compression (SNC) [35], sparse compositional metric learning (SCML) [59] and reduced-rank local distance metric learning (R2LML) [26]. LMNN and ITML are implemented by using the metric-learn tool-

Table 5.1: Experiment results of smooth metric learning with instance extraction. Mean accuracy (percentage) and standard deviations are reported with the best ones in bold; '# of best' indicates the number of data sets that an algorithm performs the best.

Data set	LMNN	ITML	MCML	NCA	RVML
Australian	78.8 ±2.57	77.17 ±1.94	78.77 ±1.70	79.96 ±1.63	83.01 ±1.58
Breastcancer	95.91 ±0.69	96.39 ±1.04	96.35 ±0.77	95 ±1.52	95.77 ±1.09
Diabetes	69.16 ±1.44	69.09 ±1.24	69.19 ±1.18	68.47 ±2.46	71.04 ±2.60
Fourclass	72.06 ±2.31	72.09 ±2.22	72.06 ±2.43	72.06 ±2.46	70.46 ±1.40
German	67.85 ±1.54	66.95 ±2.05	67.67 ±1.48	69.95 ±2.88	71.65 ±1.78
Haberman	67.89 ±3.34	67.97 ±4.05	67.56 ±2.75	67.4 ±3.33	66.67 ±2.30
Heart	76.2 ±3.82	76.94 ±3.30	77.22 ±3.66	75.56 ±2.01	77.69 ±4.05
ILPD	66.97 ±2.13	68.67 ±2.83	67.48 ±2.58	66.8 ±1.19	67.95 ±2.90
Liverdisorders	61.01 ±4.80	57.17 ±4.01	60.65 ±5.12	59.78 ±3.44	64.64 ±3.93
Pima	68.54 ±1.64	67.95 ±2.01	68.31 ±2.33	65.91 ±3.04	69.45 ±1.68
Voting	94.83 ±0.77	90.75 ±1.44	92.64 ±1.58	94.77 ±0.92	95.75 ±1.26
WDBC	96.58 ±1.12	94.91 ±0.92	95.7 ±0.90	96.58 ±0.85	96.58 ±1.34
# of best	0	0	0	0	0

Dataset	GMML	SCML	R2LML	SNC	Smooth MLIE
Australian	84.35 ±1.04	82.25 ±1.40	84.67 ±1.32	81.78 ±8.8	**85.52 ±1.98**
Breastcancer	**97.26 ±0.81**	97.01 ±0.91	97.01 ±0.66	96.65 ±0.69	96.98 ±0.79
Diabetes	74.16 ±2.58	71.49 ±2.21	73.8 ±1.37	**75.32 ±2.74**	75.22 ±2.49
Fourclass	76.12 ±1.87	75.54 ±1.42	**76.12 ±1.91**	73.39 ±8.7	74.53 ±2.93
German	71.55 ±1.12	70.9 ±2.65	72.9 ±1.83	70.13 ±3.33	**73.03 ±1.79**
Haberman	71.22 ±3.35	69.19 ±2.47	71.06 ±3.39	71.98 ±5.2	**72.35 ±4.02**
Heart	81.2 ±2.69	78.98 ±3.24	82.04 ±3.81	77.04 ±5.32	**82.31 ±2.92**
ILPD	67.14 ±2.17	68.03 ±2.90	65.85 ±2.22	68.91 ±2.67	**69.12 ±2.72**
Liverdisorders	63.84 ±5.43	61.74 ±4.57	**66.81 ±3.68**	63.31 ±5.18	66.66 ±4.71
Pima	72.95 ±1.84	71.14 ±2.64	72.34 ±1.54	73.99 ±2.59	**74.91 ±2.86**
Voting	95.17 ±1.88	95 ±1.30	**96.32 ±1.19**	94.45 ±1.2	95.11 ±1.25
WDBC	96.71 ±0.78	96.97 ±0.89	96.93 ±1.67	96.93 ±0.85	**97.63 ±1.22**
# of best	1	1	2	1	7

box[3]; NCA and MCML are implemented by using the drToolbox[4]; and GMML, RVML, SCML, R2LML and SNC are implemented by using the authors' code.

The experiment is focused on binary classification of 12 publicly available data sets from the websites of UCI[5] and LibSVM[6], namely Australian, Breastcancer, Diabetes, Fourclass, Germannumber, Haberman, Heart, ILPD, Liverdisorders, Pima, Voting and WDBC. All data sets are pre-processed by firstly subtracting the mean and dividing by the standard deviation, and then normalising the L_2-norm of each instance to 20.

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. 10-fold cross-validation is used to select the trade-off parameters in the compared algorithms, namely the regularisation 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 $\{0.1, 0.3, 0.5, 0.7, 0.9\}$), λ in RVML (from $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$), Ratio in SNC (from $\{0.01, 0.02, 0.04, 0.08, 0.16\}$). All other parameters are set as default. For the proposed algorithm, the parameters are set as follows: the initial weight of $\boldsymbol{L}$ is set as the identity matrix $\boldsymbol{I}$; the initial values of $\boldsymbol{r}^m$ are set as the k-means clustering (Matlab kmeans function with random initial values) centres of the positive and negative classes; the number of extracted instances for each class is set as 2; the trade-off parameter λ is set as 1 and the learning rate α is set as 0.001. The maximum number of iterations is set as 5000 and the final result is based on the parameters at time t, which is the earliest time when the smallest training error is obtained.

As shown in Table 5.1, with only two instances extracted from each class, the proposed algorithm achieves the best accuracy on 7 data sets out of the 12 data sets. None of the other algorithms performs the best in more than 3 data sets. These experiment results show the proposed algorithm enjoys competitive performance

[3] https://all-umass.github.io/metric-learn/

[4] https://lvdmaaten.github.io/drtoolbox/

[5] https://archive.ics.uci.edu/ml/datasets.html

[6] https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/binary.html

against state-of-the-art metric learning algorithms.

5.5 Conclusion

In this chapter, the generalisation PAC bound is used to cover the factors related to the optimisation process. Based on the resultant bound, a smooth classifier and a smooth loss function are used for MLIE. Compared to the experiment result in last chapter, considering the smoothness property has improved the performance of the classifier.

5.6 Appendix

5.6.1 Proof of Theorem 10

The definition of PAC and agnostic PAC learnable is reviewed. After that, the required Lemma of McDiarmid's inequality is introduced. Then one proposition is proved and Finally Theorem 10 is proved.

Definition 38. [57, 65] A hypothesis class $\mathcal{H}$ is *Probably Approximately Correct (PAC) learnable* if there exist a function $n_{\mathcal{H}}^{L} : (0,1)^2 \rightarrow \mathbb{N}$ and a learning algorithm with the following property: For every $\epsilon, \delta \in (0,1)$, for every distribution $\mathcal{D}_{\mathcal{X}}$ over $\mathcal{X}$, and for every target function $g \in \mathcal{G}$, if there exists an $h^* \in \mathcal{H}$ which returns the same classification result as g, then when running the learning algorithm on $n \geq n_{\mathcal{H}}^{L}(\epsilon, \delta)$ independent and identically distributed (i.i.d.) instances generated by $\mathcal{D}_{\mathcal{X}}$ and labelled by g, the algorithm returns a hypothesis $\hat{h}$, such that, with probability at least $1-\delta$, $R(\hat{h}) \leq \epsilon$, which can be equivalently written as

$$\mathbb{P}_{\boldsymbol{x}^n}\Big(R(\hat{h}) \leq \epsilon\Big) \geq 1-\delta,$$

or

$$\mathbb{P}_{\boldsymbol{x}^n}\Big(\mathbb{E}_{\boldsymbol{x}'}\big[l\big(\hat{h}(\boldsymbol{x}'); g(\boldsymbol{x}')\big)\big] \leq \epsilon\Big) \geq 1-\delta,$$

where the probability is over $\boldsymbol{x}_n$ and $\hat{h}$ is a random variable related to $\boldsymbol{x}_n$.

Definition 39. [57, 23] A hypothesis class $\mathcal{H}$ is *agnostic PAC learnable* or has *agnostic PAC learnability* if there exist a function $n_{\mathcal{H}}^{AL} : (0,1)^2 \rightarrow \mathbb{N}$ and a learning algorithm with the following property: For every $\epsilon, \delta \in (0,1)$ and for every distribution $\mathcal{D}_{\mathcal{Z}}$ over $\mathcal{Z}$, when running the learning algorithm on $n \geq n_{\mathcal{H}}^{AL}(\epsilon, \delta)$ i.i.d. instances generated by $\mathcal{D}_{\mathcal{Z}}$, the algorithm returns a hypothesis $\hat{h}$ which satisfies the following *agnostic PAC learning bound*: with probability at least $1-\delta$,

$$R(\hat{h}) - \min_{h \in \mathcal{H}} R(h) \leq \epsilon.$$

The above agnostic PAC learning bound can be equivalently written as

$$\mathbb{P}_{\boldsymbol{z}^n}\Big(R(\hat{h}) - \min_{h \in \mathcal{H}} R(h) \leq \epsilon\Big) \geq 1 - \delta,$$

or more explicitly

$$\mathbb{P}_{\boldsymbol{z}^n}\Big(\mathbb{E}_{\boldsymbol{z}'}\big[l\big(\hat{h}(\boldsymbol{x}'); y\big)\big] - \min_{h \in \mathcal{H}} \mathbb{E}_{\boldsymbol{z}'}\big[l\big(h(\boldsymbol{x}'); y\big)\big] \leq \epsilon\Big) \geq 1 - \delta.$$

where the probability is over $\boldsymbol{z}_n$ and $\hat{h}$ is a random variable related to $\boldsymbol{z}_n$.

Lemma 11. [46] Let $\boldsymbol{z}^n = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_{i-1}, \boldsymbol{z}_i, \boldsymbol{z}_{i+1}, \ldots, \boldsymbol{z}_n\}$ be n independent samples; Let $\boldsymbol{z}^{n,i} = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_{i-1}, \boldsymbol{z}_i^{'}, \boldsymbol{z}_{i+1}, \ldots, \boldsymbol{z}_n\}$, where the replacement example $\boldsymbol{z}_i^{'}$ is assumed to be drawn from the same distribution of $\boldsymbol{z}_i$ and it is independent from $\boldsymbol{z}^n$. Furthermore, let $m : \mathcal{Z}^n \to \mathbb{R}$ be a function of $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$ that satisfies $\forall i, \forall \boldsymbol{z}^n, \forall \boldsymbol{z}^{n,i}$

$$|m(\boldsymbol{z}^n) - m(\boldsymbol{z}^{n,i})| \leq c_i, \tag{5.13}$$

for some constant c_i. Then for all $\epsilon > 0$, *McDiarmid's Inequality* states that

$$\mathbb{P}_{\boldsymbol{z}^n}(m(\boldsymbol{z}^n) - \mathbb{E}_{\boldsymbol{z}^n}(m(\boldsymbol{z}^n)) \geq \epsilon) \leq \exp(\frac{-2\epsilon^2}{\sum_{i=1}^n c_i^2})$$

$$\mathbb{P}_{\boldsymbol{z}^n}(\mathbb{E}_{\boldsymbol{z}^n}(m(\boldsymbol{z}^n)) - m(\boldsymbol{z}^n) \geq \epsilon) \leq \exp(\frac{-2\epsilon^2}{\sum_{i=1}^n c_i^2})$$

i.e.

$$\mathbb{P}_{\boldsymbol{z}^n}(|m(\boldsymbol{z}^n) - \mathbb{E}_{\boldsymbol{z}^n}(m(\boldsymbol{z}^n))| \geq \epsilon) \leq 2\exp(\frac{-2\epsilon^2}{\sum_{i=1}^n c_i^2}) \tag{5.14}$$

Proposition 17. Suppose the range of the risk function $r(\boldsymbol{z}, h)$ is bounded by $[0, C_r]$, then

$$\mathbb{P}_{\boldsymbol{z}^n}\Big(\min_{h\in\mathcal{H}} R_n(\boldsymbol{z}^n, h) - E_{\boldsymbol{z}^n}[\min_{h\in\mathcal{H}} R_n(\boldsymbol{z}^n, h)] \geq \epsilon\Big) \leq \exp(\frac{-2n\epsilon^2}{C_r^2}).$$

Proof. Given $\boldsymbol{z}^n$ and fixed hypothesis set of $\mathcal{H}$, then the value of $a(\boldsymbol{z}^n) = \min_{h\in\mathcal{H}} R_n(\boldsymbol{z}^n, h)$ is fixed and this mapping $a : \mathcal{Z}^n \to \mathbb{R}$ is a function. So we can use Lemma 11 and we need to bound $|\min_{h\in\mathcal{H}} R_n(\boldsymbol{z}^n, h) - \min_{h\in\mathcal{H}} R_n(\boldsymbol{z}^{n,i}, h)|$ as follows,

$$\begin{aligned}
&\min_{h\in\mathcal{H}} R_n(\boldsymbol{z}^{n,i}, h)\\
=&\min_{h\in\mathcal{H}}\Big(R_n(\boldsymbol{z}^n, h) - \frac{r(z_i, h)}{n} + \frac{r(z_i', h)}{n}\Big)\\
\leq&\min_{h\in\mathcal{H}}\Big(R_n(\boldsymbol{z}^n, h) - 0 + \frac{C_r}{n}\Big)\\
=&\min_{h\in\mathcal{H}}\Big(R_n(\boldsymbol{z}^n, h)\Big) + \frac{C_r}{n}
\end{aligned}$$

Similarly,

$$\min_{h\in\mathcal{H}} R_n(\boldsymbol{z}^n, h) \leq \min_{h\in\mathcal{H}}\Big(R_n(\boldsymbol{z}^{n,i}, h)\Big) + \frac{C_r}{n}.$$

Therefore

$$|\min_{h\in\mathcal{H}} R_n(\boldsymbol{z}^n, h) - \min_{h\in\mathcal{H}} R_n(\boldsymbol{z}^{n,i}, h)| \leq \frac{C_r}{n}.$$

The result is obtained by substitute the above $\frac{C_r}{n}$ into Lemma 11. □

Then Theorem 10 is proved as follows.

Proof. Let $\hat{h} \in \operatorname{argmin}_{h\in\mathcal{H}} R_n(\boldsymbol{z}^n, h)$, we have

$$R_n(\boldsymbol{z}^n, \hat{h}) = \min_{h\in\mathcal{H}} R_n(\boldsymbol{z}^n, h).$$

Suppose

$$\mathbb{P}_{\boldsymbol{z}^n}\Big(R(\hat{h}) - R_n(\boldsymbol{z}^n, \hat{h}) \leq \epsilon/2\Big) \geq 1 - \delta/2,$$

$$\mathbb{P}_{\boldsymbol{z}^n}\Big(R_n(\boldsymbol{z}^n,\hat{h}) - E_{\boldsymbol{z}^n}[R_n(\boldsymbol{z}^n,\hat{h})] \leq \epsilon/2\Big) \geq 1-\delta/2.$$

Let $E1 = \{\boldsymbol{z}^n | R(\hat{h}) - R_n(\boldsymbol{z}^n,\hat{h}) \leq \epsilon/2\}$; $E2 = \{\boldsymbol{z}^n | R_n(\boldsymbol{z}^n,\hat{h}) - E_{\boldsymbol{z}^n}[R_n(\boldsymbol{z}^n,\hat{h})] \leq \epsilon/2\}$. $\forall \boldsymbol{z}^n \in E1 \cap E2$, we have

$$\begin{aligned}
&R(\hat{h})\\
(a)\quad &\leq R_n(\boldsymbol{z}^n,\hat{h}) + \frac{\epsilon}{2}\\
(b)\quad &\leq \mathbb{E}_{\boldsymbol{z}^n}[R_n(\boldsymbol{z}^n,\hat{h})] + \epsilon\\
(c)\quad &= \mathbb{E}_{\boldsymbol{z}^n} \min_{h\in\mathcal{H}} \frac{\sum_{i=1}^n r(\boldsymbol{z}_i,h)}{n} + \epsilon\\
(d)\quad &\leq \min_{h\in\mathcal{H}} \mathbb{E}_{\boldsymbol{z}^n} \frac{\sum_{i=1}^n r(\boldsymbol{z}_i,h)}{n} + \epsilon\\
(e)\quad &= \min_{h\in\mathcal{H}} \mathbb{E}_{\boldsymbol{z}} r(\boldsymbol{z},h) + \epsilon\\
(f)\quad &= \min_{h\in\mathcal{H}} R(h) + \epsilon,
\end{aligned}$$

where inequality (a) is due to $R(\hat{h}) - R_n(\boldsymbol{z}^n,\hat{h}) \leq \epsilon/2$; inequality (b) is due to $R_n(\boldsymbol{z}^n,\hat{h}) - E_{\boldsymbol{z}^n}[R_n(\boldsymbol{z}^n,\hat{h})] \leq \epsilon/2$; equality (c) is due to the definitions of $R_n(\boldsymbol{z}^n,h)$ and $\hat{h}$; inequality (d) is due to change the order of $E_{\boldsymbol{z}^n}$ and $\min_{h\in\mathcal{H}}$; equality (e) is due to the identical assumption of $\boldsymbol{z}^n$; equality (f) is due to the definition of $R(h)$.

Therefore

$$\begin{aligned}
&\mathbb{P}_{\boldsymbol{z}^n}\Big(R(\hat{h}) \leq \min_{h\in\mathcal{H}} R(h) + \epsilon\Big)\\
(a)\quad &\geq \mathbb{P}_{\boldsymbol{z}^n}\Big(E1 \cap E2\Big)\\
(b)\quad &\geq 1 - \delta_1 - \delta_2,
\end{aligned}$$

where inequality (a) is due to the relationship between $E1 \cap E2$ and $R(\hat{h}) \leq \min_{h\in\mathcal{H}} R(h) + \epsilon$; inequality (b) is due to the probability of union of sets.

Based on Proposition 17, to guarantee $\mathbb{P}_{\boldsymbol{z}^n}\Big(R_n(\boldsymbol{z}^n,\hat{h}) - E_{\boldsymbol{z}^n}[R_n(\boldsymbol{z}^n,\hat{h})] \leq \epsilon/2\Big) \geq 1-\delta/2$ is satisfied, $\frac{2C_r^2}{\epsilon^2}\ln\frac{4}{\delta}$ instances are required.

At the same time, based on Definition 35, to guarantee $\mathbb{P}_{\boldsymbol{z}^n}\Big(R(\hat{h}) - R_n(\boldsymbol{z}^n,\hat{h}) \leq \epsilon/2\Big) \geq 1-\delta/2$ is satisfied, $m_{\mathcal{H}}^{G}(\epsilon/2,\delta/2)$ instances are required. Therefore, with more than $\max[m^G_{\mathcal{H}}(\epsilon/2, \delta/2), \frac{2C_r^2}{\epsilon^2} \ln \frac{4}{\delta}]$ instances, $\mathbb{P}_{\boldsymbol{z}^n}\Big(R(\hat{h}) \leq \epsilon\Big) \geq 1 - \delta$ is satisfied. Based on Definition 39, the hypothesis set is (agnostic) PAC learnable and the agnostic PAC learner for $\mathcal{H}$ is $\mathrm{ERM}_{\mathcal{H}}$. □

5.6.2 Theorem 13

Theorem 13. Let $r(\boldsymbol{z}, \boldsymbol{w}) = l(h(\boldsymbol{x}, \boldsymbol{w}))$ be the loss function of a parameterized function h and $\boldsymbol{w} \in \mathbb{R}^Q$. Suppose $\mathrm{lip}(r \leftarrow \boldsymbol{w}) \leq L$ and and $\mathrm{diam}(\mathcal{W}, \|\cdot\|_2) \leq B$, then the following inequality holds

$$\mathbb{P}_{\boldsymbol{z}^n}\Big[\max_{\boldsymbol{w}\in\mathcal{W}} R(\boldsymbol{w}) - R_n(\boldsymbol{z}^n, \boldsymbol{w}) \leq CLB\sqrt{\frac{Q}{n}} + \sqrt{\frac{\ln 1/\delta}{2n}}\Big] \geq 1 - \delta,$$

where C is a universal constant and $\mathrm{lip}(r \leftarrow \boldsymbol{w})$ is the Lipschitz constant of function r with respect to $\boldsymbol{w}$, which is defined as follows

$$\mathrm{lip}(r \leftarrow \boldsymbol{w}) = \max_{\boldsymbol{w}_1, \boldsymbol{w}_2 \in \mathcal{W}, \boldsymbol{w}_1 \neq \boldsymbol{w}_2, \boldsymbol{z} \in \mathcal{Z}} \frac{|r(\boldsymbol{z}; \boldsymbol{w}_1) - r(\boldsymbol{z}; \boldsymbol{w}_2)|}{\|\boldsymbol{w}_1 - \boldsymbol{w}_2\|}.$$

5.6.3 Proof of Lemma 9

Proof. Let E_1 be the set of events of $R(\hat{h}) - R_n(\boldsymbol{z}^n, \hat{h}) \leq \epsilon$ and E_2 be the set of events of $\max_{h\in\mathcal{H}}[R(h) - R_n(\boldsymbol{z}^n, h)] \leq \epsilon$

$$\mathbb{P}_{\boldsymbol{z}^n}(E_1) = \int \Bigg(f(\boldsymbol{z}^n)\mathbb{1}[E_1] \Bigg) d\boldsymbol{z}^n$$
$$\mathbb{P}_{\boldsymbol{z}^n}(E_2) = \int \Bigg(f(\boldsymbol{z}^n)\mathbb{1}[E_2] \Bigg) d\boldsymbol{z}^n.$$

1. At the points $\boldsymbol{z}^n$ where $\mathbb{1}[E_2] = 1$, we have $\mathbb{1}[E_1] = 1$, thus

$$f(\boldsymbol{z}^n)\mathbb{1}[E_1] = f(\boldsymbol{z}^n)\mathbb{1}[E_2].$$

2. At the points $\boldsymbol{z}^n$ where $\mathbb{1}[E_2] = 0$, we have

$$f(\boldsymbol{z}^n)\mathbb{1}[E_1] \geq 0 = f(\boldsymbol{z}^n)\mathbb{1}[E_2].$$

Therefore, at each $\boldsymbol{z}^n$, $\left(\int f(\boldsymbol{z}^n)\mathbb{1}[E_1]\right) \geq \left(f(\boldsymbol{z}^n)\mathbb{1}[E_2]\right)$. Thus $\mathbb{P}_{\boldsymbol{z}^n}(E_1) \geq \mathbb{P}_{\boldsymbol{z}^n}(E_2)$. □

5.6.4 Proof of Theorem 11

Proof. Let E_1 denote the set of events $R(\hat{\boldsymbol{w}}) - R_n(\boldsymbol{z}^n, \hat{\boldsymbol{w}}) \leq \epsilon$; let E_2 denote the set of events $\boldsymbol{m}(\boldsymbol{z}^n) \in \hat{\mathcal{W}}$; let E_3 denotes the set of events $\max_{\boldsymbol{w}\in\hat{\mathcal{W}}} R(\boldsymbol{w}) - R_n(\boldsymbol{z}^n, \boldsymbol{w}) \leq \epsilon$.

$$\begin{aligned}
& \mathbb{P}_{\boldsymbol{z}^n}[\neg E_1] \\
& = \mathbb{P}_{\boldsymbol{z}^n}[\neg E_1, E_2] + \mathbb{P}_{\boldsymbol{z}^n}[\neg E_1, \neg E_2] \\
(d) \quad & \leq \mathbb{P}_{\boldsymbol{z}^n}[\neg E_1, E_2] + \delta_1 \\
(e) \quad & \leq \mathbb{P}_{\boldsymbol{z}^n}[\neg E_3] + \delta_1 \\
& = \delta_2 + \delta_1;
\end{aligned}$$

where inequality (d) is due to $\mathbb{P}_{\boldsymbol{z}^n}[\neg E_1, \neg E_2] \leq \mathbb{P}_{\boldsymbol{z}^n}[\neg E_2] = 1 - \mathbb{P}_{\boldsymbol{z}^n}[E_2] \leq \delta_1$; inequality (e) is based on the relationship between $\mathbb{1}[E_2]\mathbb{1}[\neg E_1]$ and $\mathbb{1}[E_3]$. At the points $\boldsymbol{z}^n$ that satisfy $\boldsymbol{m}(\boldsymbol{z}^n) \in \hat{\mathcal{W}}$, $\mathbb{1}[\neg E_1] = 1 \Rightarrow \mathbb{1}[\neg E_3] = 1$, thus $\mathbb{1}[E_2]\mathbb{1}[\neg E_1] \leq \mathbb{1}[\neg E_3]$ and $\mathbb{P}_{\boldsymbol{z}^n}[\neg E_1, E_2] \leq \mathbb{P}_{\boldsymbol{z}^n}[\neg E_3]$. □

5.6.5 Proof of Lemma 10

Proof. $\boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s})$ and $\boldsymbol{z}^n$ satisfy the $\mathcal{Z}^n \to \mathbb{R}$ function and independent assumptions of Lemma 11. To concentrate the difference between $\boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s})$ and its expectation, $|\boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s}) - \boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^{n,i}; \boldsymbol{s})|$ need to be bounded. $\forall \boldsymbol{s}, \forall q$, $|\boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s}) - \boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^{n,i}; \boldsymbol{s})| \leq \|\boldsymbol{m}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s}) - \boldsymbol{m}^{(T)}(\boldsymbol{z}^{n,i}; \boldsymbol{s})\|$, where $\|\cdot\|$ denotes the vector l_2 norm[7]. We will now discuss the bound of $\|\boldsymbol{m}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s}) - \boldsymbol{m}^{(T)}(\boldsymbol{z}^{n,i}; \boldsymbol{s})\|$. $\boldsymbol{m}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s})$ and $\boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^n; \boldsymbol{s})$ are temporarily simplified to $\boldsymbol{m}^{(T)}(\boldsymbol{z}^n)$ and $\boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^n)$ respectively.

(1) Decompose $\boldsymbol{m}^{(t)}(\boldsymbol{z}^n)$

To see the influence of $\boldsymbol{z}_i$, the updating equation of $\boldsymbol{m}^{(t)}(\boldsymbol{z}^n)$ could be divided into

[7] In the cases of $\boldsymbol{w}$ being matrix, the matrix would be reshaped into a vector and then vector l_2 norm would be used, which is equivalent to using matrix Frobenius norm directly.

two parts

$$\boldsymbol{m}^{(t)}(\boldsymbol{z}^n) = \left(\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n) - \sum_{j\in[n]/i} \frac{\alpha^{(t)}}{n}\frac{\partial r(\boldsymbol{z}_j,\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n)}\right) - \frac{\alpha^{(t)}}{n}\frac{\partial r(\boldsymbol{z}_i,\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n)}.$$

The updating process represented with operator G is as follows

$$\boldsymbol{m}^{(t)}(\boldsymbol{z}^n) = G(\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n)) - \frac{\alpha^{(t)}}{n}\frac{\partial r(\boldsymbol{z}_i,\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n)}.$$

For both $\boldsymbol{z}^n$ and $\boldsymbol{z}^{n,i}$, $G(\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n)) = G(\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^{n,i}))$ because all training instances considered in G are the same. Then

$$\begin{aligned}
&\|\boldsymbol{m}^{(t)}(\boldsymbol{z}^n) - \boldsymbol{m}^{(t)}(\boldsymbol{z}^{n,i})\| \\
=&\left\| G(\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n)) - \frac{\alpha^{(t)}}{n}\frac{\partial r(\boldsymbol{z}_i,\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n)} - \right. \\
&\left. G(\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^{n,i})) + \frac{\alpha^{(t)}}{n}\frac{\partial r(\boldsymbol{z}_{i'},\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^{n,i})} \right\| \\
\leq&\left\| \frac{\alpha^{(t)}}{n}\frac{\partial r(\boldsymbol{z}_i,\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n)} - \frac{\alpha^{(t)}}{n}\frac{\partial r(\boldsymbol{z}_{i'},\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^{n,i})} \right\| \text{ (Term 1)+} \\
&\left\| G(\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n)) - G(\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^{n,i})) \right\| \text{ (Term 2)}
\end{aligned}$$

Term 1 and term 2 in the inequality can be bounded by using the Lipschitz constant of function r with respect to $\boldsymbol{w}$ and the Lipschitz constant of G with respect to $\boldsymbol{w}$ respectively.

(2) Bound Term 1: with the following definition of $\operatorname{lip}(r \leftarrow \boldsymbol{w})$

$$\operatorname{lip}(r \leftarrow \boldsymbol{w}) = \max_{\boldsymbol{w}_1,\boldsymbol{w}_2\in\mathcal{W},\boldsymbol{w}_1\neq\boldsymbol{w}_2,\boldsymbol{z}\in\mathcal{Z}} \frac{|r(\boldsymbol{z},\boldsymbol{w}_1) - r(\boldsymbol{z},\boldsymbol{w}_2)|}{\|\boldsymbol{w}_1 - \boldsymbol{w}_2\|},$$

term 1 is bounded as follows

$$\begin{aligned}
&\left\|\frac{\alpha^{(t)}}{n}\frac{\partial r(\boldsymbol{z}_i,\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n)} - \frac{\alpha^{(t)}}{n}\frac{\partial r(\boldsymbol{z}_{i'},\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^{n,i})}\right\| \\
\leq&\left\|\frac{\alpha^{(t)}}{n}\frac{\partial r(\boldsymbol{z}_i,\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n)}\right\| + \left\|\frac{\alpha^{(t)}}{n}\frac{\partial r(\boldsymbol{z}_{i'},\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^{n,i})}\right\| \\
\leq&\frac{2\alpha^{(t)}}{n}\operatorname{lip}(r \leftarrow \boldsymbol{w}).
\end{aligned}$$

(3) Bound Term 2: the Lipschitz constant of the operator is illustrated via the following definition of η-expansive. Term 2 is bounded as follows

$$\left\|G(\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n)) - G(\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^{n,i}))\right\| \leq \eta\|\boldsymbol{m}^{(t-1)}(\boldsymbol{z}^n) - \boldsymbol{m}^{(t-1)}(\boldsymbol{z}^{n,i})\right\|$$

(4) Iterations

$t = 1$

$$\begin{aligned}
&\|\boldsymbol{m}^{(1)}(\boldsymbol{z}^n) - \boldsymbol{m}^{(1)}(\boldsymbol{z}^{n,i})\| \\
\leq&\|\frac{\alpha^{(1)}}{n}\frac{\partial r(\boldsymbol{z}_i,\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{w}^0} - \frac{\alpha^{(1)}}{n}\frac{\partial r(\boldsymbol{z}_{i'},\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{w}^0}\| + \|G(\boldsymbol{w}^0) - G(\boldsymbol{w}^0)\| \\
\leq&\frac{2\alpha^{(1)}}{n}\operatorname{lip}(r \leftarrow \boldsymbol{w})
\end{aligned}$$

$t = 2$

$$\begin{aligned}
&\|\boldsymbol{m}^{(2)}(\boldsymbol{z}^n) - \boldsymbol{m}^{(2)}(\boldsymbol{z}^{n,i})\| \\
\leq&\|\frac{\alpha^{(2)}}{n}\frac{\partial r(\boldsymbol{z}_i,\boldsymbol{w})}{\partial \boldsymbol{w}}|_{m^{(1)}(\boldsymbol{z}^n)} - \frac{\alpha^{(2)}}{n}\frac{\partial r(\boldsymbol{z}_{i'},\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{m}^{(1)}(\boldsymbol{z}^{n,i})}\|+ \\
&\|G(\boldsymbol{m}^{(1)}(\boldsymbol{z}^n)) - G(\boldsymbol{m}^{(1)}(\boldsymbol{z}^{n,i}))\| \\
\leq&\frac{2\alpha^{(2)}}{n}\operatorname{lip}(r \leftarrow \boldsymbol{w}) + \eta\frac{2\alpha^{(1)}}{n}\operatorname{lip}(r \leftarrow \boldsymbol{w}) \\
=&\frac{2(\eta\alpha^{(1)} + \alpha^{(2)})\operatorname{lip}(r \leftarrow \boldsymbol{w})}{n}
\end{aligned}$$

$\vdots$

$t = T$

$$\begin{aligned}
&\|\boldsymbol{m}^{(T)}(\boldsymbol{z}^n) - \boldsymbol{m}^{(T)}(\boldsymbol{z}^{n,i})\| \\
\leq&\|\frac{\alpha^{(T)}}{n}\frac{\partial r(\boldsymbol{z}_i, \boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{m}^{(T-1)}(\boldsymbol{z}^n)} - \frac{\alpha^{(T)}}{n}\frac{\partial r(\boldsymbol{z}_{i'}, \boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{m}^{(T-1)}(\boldsymbol{z}^{n,i})}\| + \\
&\|G(\boldsymbol{m}^{(T-1)}(\boldsymbol{z}^n)) - G(\boldsymbol{m}^{(T-1)}(\boldsymbol{z}^{n,i}))\| \\
\leq&\frac{2\Big(\sum_{t=1}^T \eta^{T-t}\alpha^{(t)}\Big)\operatorname{lip}(r \leftarrow \boldsymbol{w})}{n}
\end{aligned}$$

(5) Concentration

$$\begin{aligned}
&|\boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^n) - \boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^{n,'})| \\
\leq&\|\boldsymbol{m}^{(T)}(\boldsymbol{z}^n) - \boldsymbol{m}^{(T)}(\boldsymbol{z}^{n,i})\| \\
\leq&\frac{2\Big(\sum_{t=1}^T \eta^{T-t}\alpha^{(t)}\Big)\operatorname{lip}(r \leftarrow \boldsymbol{w})}{n} \\
=&\frac{C}{n}
\end{aligned}$$

where $C = 2\Big(\sum_{t=1}^T \eta^{T-t}\alpha^{(t)}\Big)\operatorname{lip}(r \leftarrow \boldsymbol{w})$. In the case of $\eta \leq 1$, $C \leq 2\sum_{t=1}^T \alpha^{(t)}\operatorname{lip}(r \leftarrow \boldsymbol{w})$; in the case of $\eta > 1$, $C \leq 2\Big(\sum_{t=1}^T \eta^{T-t}\Big)\Big(\max_t \alpha^{(t)}\Big)\operatorname{lip}(r \leftarrow \boldsymbol{w}) = 2\frac{\eta^T-1}{\eta-1}\Big(\max_t \alpha^{(t)}\Big)\operatorname{lip}(r \leftarrow \boldsymbol{w})$.

Therefore, based on Lemma 11, $\forall s$, $\boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^n, \boldsymbol{a}; \boldsymbol{s})$ can be bounded as

$$\begin{aligned}
\mathbb{P}_{\boldsymbol{z}^n}[|\boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^n) - \mathbb{E}_{\boldsymbol{z}^n}\boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^n)| \leq \frac{\epsilon}{\sqrt{Q}}] &\geq 1 - \exp(\frac{-2\epsilon^2}{Q\sum_{i=1}^n c_i^2}) \\
&= 1 - \exp(\frac{-2\epsilon^2 n}{QC^2})
\end{aligned}$$

Therefore,

$$\begin{aligned}
&\mathbb{P}_{\boldsymbol{z}^n}[\|\boldsymbol{m}^{(T)}(\boldsymbol{z}^n) - \mathbb{E}_{\boldsymbol{z}^n}\boldsymbol{m}^{(T)}(\boldsymbol{z}^n)\| \leq \epsilon] \\
(a)\quad \geq&\mathbb{P}_{\boldsymbol{z}^n}[\bigcap_{q=1}^{Q}\|\boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^n) - \mathbb{E}_{\boldsymbol{z}^n}\boldsymbol{m}_{[q]}^{(T)}(\boldsymbol{z}^n)\| \leq \frac{\epsilon}{\sqrt{Q}}] \\
(b)\quad \geq&1 - Q\exp(\frac{-2\epsilon^2 n}{QC^2}).
\end{aligned}$$

where inequality (a) is due the relationship between the events; inequality (b) is due to the probability of the union of events. □

5.6.6 Proof of Theorem 12

Proof. Let $\mathrm{ball}(E,B) := \mathrm{ball}\left(\mathbb{E}_{\boldsymbol{z}^n}\boldsymbol{m}^{(T)}(\boldsymbol{z}^n), B\right)$ denote the ball with the centre at $\mathbb{E}_{\boldsymbol{z}^n}\boldsymbol{m}^{(T)}(\boldsymbol{z}^n)$ and radius of B. In the result of Corollary 7,

$$\mathbb{P}_{\boldsymbol{z}^n}\Big[\boldsymbol{m}(\boldsymbol{z}^n) \in \mathrm{ball}(E,B)\Big] \geq 1-\delta_1, \tag{5.15}$$

where $\delta_1 = Q\exp(\frac{-2B^2n}{QC_2^2L^2})$ i.e. $B = C_2L_1\sqrt{\frac{Q}{2n}\ln\frac{Q}{\delta_1}}$.

Based on the result of Theorem 13,

$$\mathbb{P}_{\boldsymbol{z}^n}\Big[\max_{\boldsymbol{w}\in ball(E,B)} R(\boldsymbol{w}) - R_n(\boldsymbol{z}^n,\boldsymbol{w}) \leq C_1L_2B\sqrt{\frac{Q}{n}} + \sqrt{\frac{\ln 1/\delta_2}{2n}}\Big] \geq 1-\delta_2.$$

Substitute $B = C_2\,\mathrm{lip}(r \leftarrow \boldsymbol{w})\sqrt{\frac{Q}{2n}\ln\frac{Q}{\delta_1}} \leq C_2L_1L_l\sqrt{\frac{Q}{2n}\ln\frac{Q}{\delta_1}}$ into the above formula

$$\mathbb{P}_{\boldsymbol{z}^n}\Big[\max_{\boldsymbol{w}\in \mathrm{ball}(\boldsymbol{a})} R(\boldsymbol{w}) - R_n(\boldsymbol{z}^n,\boldsymbol{w}) \leq \epsilon\Big] \geq 1-\delta_2. \tag{5.16}$$

$$\epsilon = \frac{C_1C_2L_1^2L_l^2Q\sqrt{\frac{1}{2}\ln(Q/\delta_1)}}{n} + \frac{\sqrt{\frac{1}{2}\ln(1/\delta_2)}}{\sqrt{n}}.$$

Based on (5.15) and (5.16), the final result is obtained using Theorem 11

$$\mathbb{P}_{\boldsymbol{z}^n}[R(\boldsymbol{m}(\boldsymbol{z}^n)) - R_n(\boldsymbol{z}^n, \boldsymbol{m}(\boldsymbol{z}^n)) \leq \epsilon] \geq 1-\delta_1-\delta_2.$$

□

Chapter 6

Summary and Future Work

6.1 Summary

In this thesis, some intuitive metric learning algorithms and their PAC bounds have been proposed. The intuitive explanations help us understand the terms in the PAC bounds, such as the intuition of large margin ratio in Chapter 2. Meanwhile, the PAC bounds can theoretically guarantee the performance of the algorithms and the requirement for 'better' PAC bounds help propose better algorithms, such as the modification from Chapter 4 to Chapter 5 has improved the performance of the algorithm.

The learnable conditions and the terms required to be bounded in each chapter are summarised in Table 6.1. The PAC bounds in different chapters are suitable for different classifiers. For classifiers without a bounded partial derivative value $\frac{\partial h}{\partial \boldsymbol{\theta}}$ but with bounded $\mathrm{lip}(h \leftarrow \boldsymbol{x})$, the bound for $\mathrm{lip}(h \leftarrow \boldsymbol{x})$ can be used, such as the local metric classifier used in Chapter 3. However, the exponential term of $(16C)^{\mathrm{ddim}(\mathcal{X})}$ means that the bound may be useful only when the sample size is very large or the doubling dimension of $\mathcal{X}$ is very small. For classifiers with a bounded gradient, a Lipschitz continuous loss and a bounded parametric space Θ, the bound proposed in Chapter 4 can be used. Furthermore, for the classifiers which enjoy the (Lipschitz) smoothness property besides the conditions mentioned in Chapter 4, there exists a bound with terms related to the optimisation process.

Chapter	Properties	
2	task intuition learnable condition bounded terms	classification large margin ratio uniform convergence $\mathrm{lip}(h \leftarrow \boldsymbol{x}), \mathrm{lip}(r \leftarrow h), \mathrm{diam}(\mathcal{X})$
3	task intuition learnable condition bounded terms	classification local metric regions uniform convergence $\mathrm{lip}(h \leftarrow \boldsymbol{x}), \mathrm{lip}(r \leftarrow h), \mathrm{diam}(\mathcal{X})$
4	task intuition learnable condition bounded terms	classification and compression local linear classifier uniform convergence $\mathrm{lip}(h \leftarrow \boldsymbol{\theta}), \mathrm{lip}(r \leftarrow h), \mathrm{diam}(\Theta)$
5	task intuition learnable condition bounded terms	classification and compression a ‘smooth’ edition of Chapter 4 generalisation PAC bound $\mathrm{lip}(h \leftarrow \boldsymbol{\theta}), \mathrm{lip}(\frac{\partial h}{\partial \boldsymbol{\theta}} \leftarrow \boldsymbol{\theta}), \mathrm{lip}(r \leftarrow h), \mathrm{diam}(\Theta)$

Table 6.1: A summary of metric learning problems and PAC learning bounds discussed in the thesis.

6.2 Future Work

6.2.1 More Regularisation Terms and Classifiers

In this thesis, the regularisation term is restricted to be vector L_2-norm or matrix Frobenius norm and the type of classifiers is limited to those discussed in the four chapters. Meanwhile, in metric learning, a large number of regularisation terms and distance-based classifiers have been proposed and verified by experiments whereas the intuition and learnablilty of these methods are rarely discussed. Therefore, it would be valuable to extend our work to more regularisation terms and distance-based classifiers.

Some deep neural networks (DNNs) can be regarded as Lipschitz functions when all layers use Lipschitz functions and hence the hierarchical structure is a composition of Lipschitz functions. Some attempts have recently been made to find the PAC bounds of DNNs, such as [3, 1], and further theoretical and intuitive explanations of more types of DNNs would be interesting work. Moreover, as metric learning classifiers have much stronger classification ability than the softmax classifier, the latter of which is widely adopted in DNNs, incorporating metric learning into DNNs may reduce the required number of layers for feature extraction or the number of nodes in each layer. Hence, it is interesting to study hierarchical metric learning and its learnability.

6.2.2 Considering Optimisation in PAC Bounds

Lipschitz properties are crucial to both learning theory and optimisation. Research on learning theory and optimisation are complementary. Besides the extensive discussions on the rate of convergence toward the minimal empirical risk, optimisation algorithms should consider the generalisation performance at the same time, which requires the help of learning theory. Meanwhile, as illustrated in [22, 47, 48], the traditional generalisation bound cannot explain many important observations of the optimisation algorithms and further work is required to better understand deep learning and many other classifiers [84]. Therefore, it is important to combine the results in these two research areas. The explanation of the generalisation ability for more advanced optimisation techniques would be another interesting future work and Lipschitz properties would be an important factor to consider.

6.2.3 Reinforcement Learning Problems

As mentioned in Chapter 1, besides classification, there exist other important learning tasks, such as reinforcement learning (RL) [64]. RL learns to take sequential actions in an environment so as to maximise the cumulative reward and it has been applied to various practical problems [45]. Lipschitz functions are used in some state-of-the-art reinforcement learning models, such as deep reinforcement learning models. Future work may focus on learnability problems and intuitions behind effective reinforcement learning algorithms.

Meanwhile, reinforcement learning may help solve many hard classification problems. For example, active learning, which designs an algorithm on selecting a subset of unlabelled instances for additional labelling so as to improve the classification performance, may be understood as a sequential decision problem and hence solved by reinforcement learning [50]. Structure learning in DNNs, which aims to learn a suitable network structure for a specific classification task, may also be viewed as a sequential decision task and the policy network is learned to add/delete nodes or layers in a DNN given current information on data and network [85]. These interesting problems, as well as their theoretical and intuitive explanations, would also be a future work.

Bibliography

[1] Radu Balan, Maneesh Singh, and Dongmian Zou. Lipschitz properties for deep convolutional networks. *arXiv preprint arXiv:1701.05217*, 2017.

[2] Peter Bartlett. Statistical learning theory course, 2016.

[3] Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems*, pages 6241–6250, 2017.

[4] Aurélien Bellet. Supervised metric learning with generalization guarantees. *arXiv preprint arXiv:1307.4514*, 2013.

[5] Julien Bohné, Yiming Ying, Stéphane Gentric, and Massimiliano Pontil. Large margin local metric learning. In *European Conference on Computer Vision*, pages 679–694. Springer, 2014.

[6] Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of machine learning research*, 2(Mar):499–526, 2002.

[7] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[8] Qiong Cao, Zheng-Chu Guo, and Yiming Ying. Generalization bounds for metric and similarity learning. *Machine Learning*, 102(1):115–132, 2016.

[9] Jason V Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th international conference on Machine learning*, pages 209–216. ACM, 2007.

[10] Dick De Ridder, Olga Kouropteva, Oleg Okun, Matti Pietikäinen, and Robert PW Duin. Supervised locally linear embedding. In *Artificial Neural Networks and Neural Information ProcessingICANN/ICONIP 2003*, pages 333–341. Springer, 2003.

[11] Huyen Do and Alexandros Kalousis. Convex formulations of radius-margin based support vector machines. In *ICML (1)*, pages 169–177, 2013.

[12] Yanni Dong, Bo Du, Lefei Zhang, Liangpei Zhang, and Dacheng Tao. LAM3L: Locally adaptive maximum margin metric learning for visual data classification. *Neurocomputing*, 235:1–9, 2017.

[13] Rémi Flamary, Marco Cuturi, Nicolas Courty, and Alain Rakotomamonjy. Wasserstein discriminant analysis. *arXiv preprint arXiv:1608.08063*, 2016.

[14] Andrea Frome, Yoram Singer, Fei Sha, and Jitendra 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*, pages 1–8. IEEE, 2007.

[15] Amir Globerson and Sam Roweis. Metric learning by collapsing classes. In *Advances in Neural Information Processing Systems*, volume 18, pages 451–458, 2005.

[16] Amir Globerson and Sam T Roweis. Metric learning by collapsing classes. In *Advances in neural information processing systems*, pages 451–458, 2006.

[17] Jacob Goldberger, Geoffrey E Hinton, Sam T Roweis, and Ruslan R Salakhutdinov. Neighbourhood components analysis. In *Advances in neural information processing systems*, pages 513–520, 2005.

[18] Lee-Ad Gottlieb, Aryeh Kontorovich, and Robert Krauthgamer. Efficient classification for metric data. *Information Theory, IEEE Transactions on*, 60(9):5750–5759, 2014.

[19] Lee-Ad Gottlieb, Aryeh Kontorovich, and Pinhas Nisnevitch. Near-optimal sample compression for nearest neighbors. In *Advances in Neural Information Processing Systems*, pages 370–378, 2014.

[20] Zheng-Chu Guo and Yiming Ying. Guaranteed classification via regularized similarity learning. *Neural computation*, 26(3):497–522, 2014.

[21] Chirag Gupta, Arun Sai Suggala, Ankit Goyal, Harsha Vardhan Simhadri, Bhargavi Paranjape, Ashish Kumar, Saurabh Goyal, Raghavendra Udupa, Manik Varma, and Prateek Jain. Protonn: Compressed and accurate knn for resource-scarce devices. In *International Conference on Machine Learning*, pages 1331–1340, 2017.

[22] Moritz Hardt, Benjamin Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. *ICML*, 2016.

[23] David Haussler. Decision theoretic generalizations of the pac model for neural net and other learning applications. *Information and Computation*, page 78150, 1992.

[24] Junlin Hu, Jiwen Lu, and Yap-Peng Tan. Discriminative deep metric learning for face verification in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1875–1882, 2014.

[25] Junlin Hu, Jiwen Lu, and Yap-Peng Tan. Sharable and individual multi-view metric learning. *IEEE transactions on pattern analysis and machine intelligence*, 2017.

[26] Yinjie Huang, Cong Li, Michael Georgiopoulos, and Georgios C Anagnostopoulos. Reduced-rank local distance metric learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 224–239. Springer, 2013.

[27] Zhiwu Huang, Ruiping Wang, Shiguang Shan, Luc Van Gool, and Xilin Chen. Cross Euclidean-to-Riemannian metric learning with application to face

recognition from video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[28] John H Hubbard and Barbara Burke Hubbard. *Vector calculus, linear algebra, and differential forms: a unified approach*. Matrix Editions, 2015.

[29] Peter J Huber. Robust estimation of a location parameter. *The annals of mathematical statistics*, pages 73–101, 1964.

[30] Jing Huo, Yang Gao, Yinghuan Shi, and Hujun Yin. Cross-modal metric learning for AUC optimization. *IEEE Transactions on Neural Networks and Learning Systems*, 2018.

[31] Tony Jebara and Pannagadatta K Shivaswamy. Relative margin machines. In *Advances in Neural Information Processing Systems*, pages 1481–1488, 2009.

[32] Hong Jia, Yiu-ming Cheung, and Jiming Liu. A new distance metric for unsupervised learning of categorical data. *IEEE transactions on neural networks and learning systems*, 27(5):1065–1079, 2016.

[33] Rong Jin, Shijun Wang, and Yang Zhou. Regularized distance metric learning: Theory and algorithm. In *Advances in neural information processing systems*, pages 862–870, 2009.

[34] Dor Kedem, Stephen Tyree, Fei Sha, Gert R Lanckriet, and Kilian Q Weinberger. Non-linear metric learning. In *Advances in Neural Information Processing Systems*, pages 2573–2581, 2012.

[35] Matt Kusner, Stephen Tyree, Kilian Q Weinberger, and Kunal Agrawal. Stochastic neighbor compression. In *Proceedings of the 31st international conference on machine learning (ICML-14)*, pages 622–630, 2014.

[36] Lubor Ladicky and Philip Torr. Locally linear support vector machines. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 985–992, 2011.

[37] Jun Li, Xun Lin, Xiaoguang Rui, Yong Rui, and Dacheng Tao. A distributed approach toward discriminative distance metric learning. *IEEE transactions on neural networks and learning systems*, 26(9):2111–2122, 2015.

[38] Chenghao Liu, Teng Zhang, Peilin Zhao, Jianling Sun, and Steven CH Hoi. Unified locally linear classifiers with diversity-promoting anchor points. In *AAAI*, 2018.

[39] Weiwei Liu, Donna Xu, Ivor Tsang, and Wenjie Zhang. Metric learning for multi-output tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[40] Jiwen Lu, Gang Wang, Weihong Deng, Pierre Moulin, and Jie Zhou. Multi-manifold deep metric learning for image set classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1137–1145, 2015.

[41] Yong Luo, Yonggang Wen, and Dacheng Tao. Heterogeneous multitask metric learning across multiple domains. *IEEE transactions on neural networks and learning systems*, 2017.

[42] Ulrike von Luxburg and Olivier Bousquet. Distance-based classification with Lipschitz functions. *The Journal of Machine Learning Research*, 5:669–695, 2004.

[43] Edward James McShane. Extension of range of functions. *Bulletin of the American Mathematical Society*, 40(12):837–842, 1934.

[44] H Quang Minh and Thomas Hofmann. Learning over compact metric spaces. In *International Conference on Computational Learning Theory*, pages 239–254. Springer, 2004.

[45] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidje-

land, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[46] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. 2012.

[47] Wenlong Mou, Liwei Wang, Xiyu Zhai, and Kai Zheng. Generalization bounds of sgld for non-convex learning: Two theoretical viewpoints. *arXiv preprint arXiv:1707.05947*, 2017.

[48] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems*, pages 5949–5958, 2017.

[49] YK Noh, BT Zhang, and DD Lee. Generative local metric learning for nearest neighbor classification. *IEEE transactions on pattern analysis and machine intelligence*, 40(1):106, 2018.

[50] Kunkun Pang, Mingzhi Dong, and Timothy Hospedales. Meta-learning transferable active learning policies by deep reinforcement learning, 2018.

[51] Neal Parikh, Stephen P Boyd, et al. Proximal algorithms. *Foundations and Trends in optimization*, 1(3):127–239, 2014.

[52] Michaël Perrot and Amaury Habrard. Regressive virtual metric learning. In *Advances in Neural Information Processing Systems*, pages 1810–1818, 2015.

[53] Qi Qian, Rong Jin, Jinfeng Yi, Lijun Zhang, and Shenghuo Zhu. Efficient distance metric learning by adaptive sampling and mini-batch stochastic gradient descent (SGD). *Machine Learning*, 99(3):353–372, 2015.

[54] Stuart Russell and Peter Norvig. A modern approach. *Artificial Intelligence*, 25, 1995.

[55] Shreyas Saxena and Jakob Verbeek. Coordinated local metric learning. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 127–135, 2015.

[56] Matthew Schultz and Thorsten Joachims. Learning a distance metric from relative comparisons. *Advances in Neural Information Processing Systems*, page 41, 2004.

[57] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[58] Chunhua Shen, Junae Kim, Fayao Liu, Lei Wang, and Anton Van Den Hengel. Efficient dual approach to distance metric learning. *IEEE transactions on neural networks and learning systems*, 25(2):394–406, 2014.

[59] Yuan Shi, Aurélien Bellet, and Fei Sha. Sparse compositional metric learning. In *AAAI*, pages 2078–2084, 2014.

[60] Phil Simon. *Too Big to Ignore: The Business Case for Big Data*. John Wiley & Sons, 2013.

[61] N Srebro and K Sridharan. Note on refined dudley integral covering number bound. *Unpublished results. http://ttic. uchicago. edu/karthik/dudley. pdf*, 2010.

[62] Joseph St Amand and Jun Huan. Sparse compositional local metric learning. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1097–1104. ACM, 2017.

[63] Jos F Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11(1-4):625–653, 1999.

[64] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[65] Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[66] Vladimir Naumovich Vapnik and Vlamimir Vapnik. *Statistical learning theory*, volume 1. Wiley New York, 1998.

[67] Nakul Verma and Kristin Branson. Sample complexity of learning mahalanobis distance metrics. In *Advances in Neural Information Processing Systems*, pages 2584–2592, 2015.

[68] JM Wainwright. High-dimensional statistics: A non-asymptotic viewpoint. *preparation. University of California, Berkeley*, 2015.

[69] Bing Wang, Gang Wang, Kap Luk Chan, and Li Wang. Tracklet association by online target-specific metric learning and coherent dynamics estimation. *IEEE transactions on pattern analysis and machine intelligence*, 39(3):589–602, 2017.

[70] Jun Wang, Alexandros Kalousis, and Adam Woznica. Parametric local metric learning for nearest neighbor classification. In *Advances in Neural Information Processing Systems*, pages 1601–1609, 2012.

[71] Wei Wang, Bao-Gang Hu, and Zeng-Fu Wang. Globality and locality incorporation in distance metric learning. *Neurocomputing*, 129:185–198, 2014.

[72] Wei Wang, Hao Wang, Chen Zhang, and Yang Gao. Cross-domain metric and multiple kernel learning based on information theory. *Neural computation*, (Early Access):1–36, 2018.

[73] Wenlin Wang, Changyou Chen, Wenlin Chen, Piyush Rai, and Lawrence Carin. Deep metric learning with data summarization. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 777–794. Springer, 2016.

[74] Nik Weaver. *Lipschitz algebras*. World Scientific, 1999.

[75] Kilian Q Weinberger, John Blitzer, and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. In *Advances in neural information processing systems*, pages 1473–1480, 2006.

[76] Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *The Journal of Machine Learning Research*, 10:207–244, 2009.

[77] Hassler Whitney. Analytic extensions of differentiable functions defined in closed sets. *Transactions of the American Mathematical Society*, 36(1):63–89, 1934.

[78] Eric P Xing, Michael I Jordan, Stuart Russell, and Andrew Y Ng. Distance metric learning with application to clustering with side-information. In *Advances in neural information processing systems*, pages 505–512, 2002.

[79] Haibin Yan, Jiwen Lu, Weihong Deng, and Xiuzhuang Zhou. Discriminative multimetric learning for kinship verification. *IEEE Transactions on Information forensics and security*, 9(7):1169–1178, 2014.

[80] Liu Yang, Rong Jin, Lily Mummert, Rahul Sukthankar, Adam Goode, Bin Zheng, Steven CH Hoi, and Mahadev 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*, 32(1):30–44, 2010.

[81] Gui-Bo Ye, Yifei Chen, and Xiaohui Xie. Efficient variable selection in support vector machines via the alternating direction method of multipliers. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 832–840, 2011.

[82] Shihui Ying, Zhijie Wen, Jun Shi, Yaxin Peng, Jigen Peng, and Hong Qiao. Manifold preserving: An intrinsic approach for semisupervised distance metric learning. *IEEE transactions on neural networks and learning systems*, 2017.

[83] Pourya Zadeh, Reshad Hosseini, and Suvrit Sra. Geometric mean metric learning. In *International Conference on Machine Learning*, pages 2464–2471, 2016.

[84] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *ICLR*, 2016.

[85] Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. 2017.