Essays on Distance Metric Learning

Xiaochen Yang

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

October 15, 2020

Abstract

Many machine learning methods, such as the k-nearest neighbours algorithm, heavily depend on the distance measure between data points. As each task has its own notion of distance, distance metric learning has been proposed. It learns a distance metric to assign a small distance to semantically similar instances and a large distance to dissimilar instances by formulating an optimisation problem.

While many loss functions and regularisation terms have been proposed to improve the discrimination and generalisation ability of the learned metric, the metric may be sensitive to a small perturbation in the input space. Moreover, these methods implicitly assume that features are numerical variables and labels are deterministic. However, categorical variables and probabilistic labels are common in real-world applications.

This thesis develops three metric learning methods to enhance robustness against input perturbation and applicability for categorical variables and probabilistic labels.

In Chapter 3, I identify that many existing methods maximise a margin in the feature space and such margin is insufficient to withstand perturbation in the input space. To address this issue, a new loss function is designed to penalise the input-space margin for being small and hence improve the robustness of the learned metric.

In Chapter 4, I propose a metric learning method for categorical data. Classifying categorical data is difficult due to high feature ambiguity, and to this end, the technique of adversarial training is employed. Moreover, the generalisation bound of the proposed method is established, which informs the choice of the regularisation term.

In Chapter 5, I adapt a classical probabilistic approach for metric learning to utilise information on probabilistic labels. The loss function is modified for training stability, and new evaluation criteria are suggested to assess the effectiveness of different methods.

At the end of this thesis, two publications on hyperspectral target detection are appended as additional work during my PhD.

Impact Statement

Classification is one of the most important tasks in the statistics and machine learning communities. The developed methods have been used in numerous applications, such as financial credit approval, face recognition, medical diagnosis, and malware filtering. This thesis focuses on one of the most classical classifiers, namely the nearest neighbour algorithm. As the method is highly interpretable, improvements on it can contribute to a more explainable and trusted machine learning model. More specifically, the impact of this thesis are threefolds.

1. The algorithms presented in Chapters 4 and 5 can improve the accuracy of the nearest neighbour algorithm for two practical problem settings. The first setting considers datasets consisting of categorical features. This type of data is prevalent in social and health sciences. The second setting considers datasets whose class labels are provided in the form of probability. This is common in medical diagnosis and appears more often nowadays with the advent of crowdsourcing platforms, which enable large datasets to be labelled by multiple annotators.

2. The algorithm presented in Chapter 3 can improve the test-time robustness of the nearest neighbour algorithm. Such robustness is an important measure to assess the reliability of classification methods.

3. Insight and theory presented in the thesis may be beneficial to researchers working on metric learning and adversarial robustness. Two geometric interpretations are provided to help understand the concept of adversarial robustness with respect to the nearest neighbour classifier (Chapter 3) and the effect

of adversarial training (Chapter 4). A finite-sample guarantee on the adversarially trained distance metric is shown, which specifies the requirement for generalisation (Chapter 4). New evaluation criteria are suggested to measure the effectiveness of metric learning algorithms in the setting of probabilistic labels (Chapter 5).

Acknowledgements

First and foremost, I would like to express my deepest and sincere gratitude to my supervisor Dr Jing-Hao Xue. Over the past four years, he has been sharing thoughts and providing detailed feedback on all my oral presentations and written reports, guiding me on different aspects of research, and giving me numerous training and opportunities to develop as an independent researcher. I am deeply grateful for his full support on my research topics and career development. Without his guidance and encouragement, I would never have made it this far.

I am thankful to Dr Ardo van den Hout for kindly being as my second supervisor. I would like to express my sincere gratitude to the Department of Statistical Science for funding my PhD and creating a supportive research environment.

I would like to thank my collaborators Dr Lefei Zhang, Dr Lianru Gao, Dr Xiaoxu Li, Dr Rui Zhu and Dr Ziyu Wang for their helpful advice on research and constructive comments on the co-authored papers. My thanks also go to Dr Wenming Yang and Dr Yang Wu for hosting my visit at Tsinghua SIGS and NAIST.

I am grateful to my colleague Dr Mingzhi Dong for his suggestions on the thesis topic, fruitful discussions about research, and patience to explain complicated ideas more than once. I am thankful to Dr Yiwen Guo for his supervision at the Intel Labs, selfless share of knowledge and research experience, valuable suggestions on research ideas, and encouragement.

On a personal note, I would like to thank my friends Ain Toha, Marco Molinari, Marcel Hirt, Yurong Ling, Yiyue Qian, Yizhuo Li and Zhuo Shen. Special thanks to Shi Chen for the support through thick and thin. Lastly, I am forever in debt to my parents and grandparents, for their unconditional love, support and encouragement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and scope of the thesis

The distance measure between data points is important for many classification, clustering and ranking methods. For example, the k-nearest neighbours (kNN) algorithm assigns the test instance to the most common class among the k nearest training instances. While metrics such as the Euclidean distance can be used, they are not specific to the classification task and thus may not produce the optimal performance. As it is important yet difficult to handcraft a distance metric for each problem, metric learning has attracted considerable research attention since its proposal [1]. Metric learning algorithms take training data as input and output a distance metric such that the resulting distance could accord well with the semantic notion of similarity.

Existing metric learning methods can be mainly divided into two categories based on the learning paradigm, namely supervised metric learning and unsupervised metric learning; the latter is more well known as manifold learning [2; 3]. Within the supervised learning paradigm, methods can be further divided depending on the type of learned metrics, such as the Mahalanobis distance, χ^2 histogram distance and string edit distance.

In this thesis, we study metric learning in a supervised setting and focus on learning the Mahalanobis distance. Compared with the conventional classification setting where the label information is available to each instance, the supervision in metric learning is often provided in the form of pairwise or triplet constraints. The pairwise constraints specify which two training instances should be similar or dissimilar, and the triplet constraints specify the relative relationship between two pairs of instances such as 'A is more similar to B than is to C'.

The Mahalanobis distance has been most widely studied in the metric learning community. For any two instances $\boldsymbol{x}_i, \boldsymbol{x}_j$, the (generalised) Mahalanobis distance is defined as $d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{M} (\boldsymbol{x}_i - \boldsymbol{x}_j)}$; the parameter matrix $\boldsymbol{M}$ is constrained to be positive semi-definite. The metric can be easily optimised by formulating a constrained convex programming problem. The loss function embeds the supervision information, which assigns a large penalty if the parameter matrix violates pairwise or triplet constraints. For example, imposing a hinge loss on triplet constraints encourage the metric to separate two instance pairs by a margin. The regularisation term, such as the Frobenius norm of the matrix, constrains the complexity of the metric and promotes its generalisation to unseen data.

1.2 Research objectives

Many metric learning methods have been proposed to improve the discriminability, generalisability and robustness of the metric, as well as the efficiency and scalability of the optimisation procedure. Nevertheless, most of them lack robustness to perturbation in the instance space. A tiny perturbation of the test instance may be magnified by the learned metric and cause a large change in the distance. Consequently, the nearest neighbour may change from the correct class to the incorrect one, and even worse, result in misclassification if 1NN classifier is used. Motivated by this issue, the first research objective of this thesis is to enhance the robustness of distance metric against instance perturbation.

Most metric learning methods implicitly assume that input data are numerical variables, but they are evaluated on datasets with categorical variables. Encoding categorical variables as integers and then treating them as real-valued numerical variables is problematic since, for nominal variables, the difference between two integers is meaningless for two nominal levels and, for ordinal variables, the difference between two integers does not accurately reflect the distance between the two ordinal levels. Considering categorical data are prevalent in health and social sciences, developing a suitable method for this data type is the second objective of this thesis.

The pairwise and triplet constraints used for supervision are often constructed based on class labels; for example, instances of the same class are considered to be similar and the ones of different classes are dissimilar. This class information is deterministic, meaning that the class membership of instances has no ambiguity. In some real-world applications, such as medical diagnosis, ambiguity exists naturally and hence labels are associated with probabilities. Adapting metric learning methods for probabilistic labels is the third research objective.

1.3 Contributions and outline

Three contributions are presented in this thesis for each of the above objectives and summarised as follows.

Contribution 1: Metric learning towards certified robustness

Chapter 3 builds on the insight that the margin in the transformed feature space, which is widely used in existing methods for separating similar and dissimilar pairs of instances, is insufficient to withstand a small perturbation of test instance in the original instance space. To enhance robustness against instance perturbation, we propose a simple yet effective solution through enlarging the adversarial margin, which is defined as the distance between a training instance and its nearest adversarial example [4] in the instance space. A closed-form solution to the adversarial margin is first derived by drawing on an intuitive geometric insight into the nearest neighbour classifier. A new perturbation loss is then defined to penalise the margin for being small. The proposed loss is flexible and can be optimised jointly with any triplet-based metric learning methods. It is also beneficial to the generalisation ability of the learned metric, which is shown by using the technique of algorithmic robustness. Extensive experiments demonstrate the superiority of the proposed method over the state-of-the-arts in terms of both discrimination accuracy and robustness to noise.

Contribution 2: Metric learning for categorical and ambiguous features

Chapter 4 develops a new method that addresses two challenges of learning the distance metric from categorical features, namely high feature ambiguity and small sample size. More specifically, ambiguity arises as the boundaries between ordinal or nominal levels are not always sharply defined. To mitigate the impact of feature ambiguity, we propose to consider the worst-case perturbation of each instance within a deliberately designed constraint set and learn the Mahalanobis distance through adversarial training [5]. Moreover, we provide a geometric interpretation of the proposed method, showing that the method dynamically divides the instance space into three regions and exploits the information on the "adversarially vulnerable" region. This information, which has not been considered in previous triplet-based methods, is useful for small-sized data. Furthermore, we establish the generalisation bound for a general form of adversarial training. It suggests that the sample complexity rate remains at the same order only if the Mahalanobis distance is regularised with the elementwise 1-norm. Experiments on ordinal and mixed ordinal-and-nominal datasets demonstrate the effectiveness of the proposed method when encountering the problems of high feature ambiguity and small sample size.

Contribution 3: Metric learning for probabilistic labels

Chapter 5 proposes to adopt a probabilistic framework for learning the metric from probabilistic labels. Such framework can sidestep the challenge of constructing pairwise or triplet constraints from probabilistic labels by directly using these labels as supervision. More specifically, we revise the existing estimation of class membership probabilities by taking into account probabilistic labels. Moreover, a new loss function based on the Jensen-Shannon divergence is proposed to encourage learning a metric such that the estimated probability, which is a function of the metric, is similar to the observed probability. Furthermore, since predicted probabilities indicate the confidence of outcome and may be useful to downstream applications, we advocate new evaluation measures to assess the classification performance of metric learning algorithms and suggest three statistical distances. Experiments on datasets with synthetic and real probabilistic labels demonstrate the benefit of utilising label probabilities to classification.

The works presented in Chapter 3 and 4 lead to the following submission and publication:

- Xiaochen Yang*, Yiwen Guo*, Mingzhi Dong, and Jing-Hao Xue. Metric learning through maximizing the adversarial margin. Submitted to *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD) journal track*, 2021.

- Xiaochen Yang*, Mingzhi Dong*, Yiwen Guo, and Jing-Hao Xue. Metric learning for categorical and ambiguous features: An adversarial method. *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, 2020.

The rest of the thesis is organised as follows. Chapter 2 provides a formal introduction to metric learning and a brief background theory on the generalisation bound and adversarial robustness. Chapters 3, 4 and 5 present Contributions 1, 2 and 3, respectively. Each chapter is self-contained with necessary preliminaries. Chapter 6 summarises the thesis and outlines future works. Appendices A and B include the following two additional works completed during the PhD. The papers present new approaches to target detection in hyperspectral images, addressing the problem of unequal noise variances over spectral bands and the issue of scarce target spectra.

- Xiaochen Yang, Lefei Zhang, Lianru Gao, and Jing-Hao Xue. MSDH: Matched subspace detector with heterogeneous noise. *Pattern Recognition Letters*, vol. 125, pp. 701-707, 2019.

- Xiaochen Yang, Mingzhi Dong, Ziyu Wang, Lianru Gao, Lefei Zhang, and Jing-Hao Xue. Data-augmented matched subspace detector for hyperspectral subpixel target detection. *Pattern Recognition*, vol. 106, Article 107464, 2020.

* Equal contribution.

Chapter 2

Background

The first part of this chapter introduces metric learning, covering the definition of a distance metric, the optimisation formulation of metric learning, and seminal algorithms from which this thesis is developed. The second part provides preliminaries on generalisation and robustness, covering some key concepts in the statistical learning theory and two theorems on the generalisation bound which will be used in Chapters 3 and 4.

2.1 Distance metric learning

2.1.1 Definition of a metric

A metric, which is also termed distance function or simply distance, is defined as follows.

Definition 1. [6] A *metric* over a vector space $\mathcal{X}$ is a function $d : \mathcal{X} \times \mathcal{X} \to [0, \infty)$ such that the following properties hold for all $\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_k \in \mathcal{X}$:

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

A function that only satisfies the first three properties is called a *pseudometric*. In the classification literature, pseudometric is often referred to as metric; we follow this simplification in the thesis.

The Mahalanobis distance [7] is the foundation of the majority of metric learning methods, owing to its nice interpretation of learning a linear transformation.

Definition 2. [6] For any $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathbb{R}^p$, the *generalised Mahalanobis distance* is defined as

$$d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{M} (\boldsymbol{x}_i - \boldsymbol{x}_j)}, \tag{2.1}$$

and parameterised by $\boldsymbol{M} \in \mathbb{S}_+^p$, where $\mathbb{S}_+^p$ denotes the cone of symmetric positive semi-definite $p \times p$ real-valued matrices.

In the original Mahalanobis distance, $\boldsymbol{M}$ is defined as the inverse covariance matrix. With a slight abuse of terminology, in the metric learning literature, any metric in the form of Eq. 2.1 is termed the Mahalanobis distance. Since $\boldsymbol{M}$ is positive semi-definite, it can be decomposed as $\boldsymbol{M} = \boldsymbol{L}^T \boldsymbol{L}$, and the Mahalanobis distance can be rewritten as:

$$\begin{aligned} d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{M} (\boldsymbol{x}_i - \boldsymbol{x}_j)} \\ &= \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{L}^T \boldsymbol{L} (\boldsymbol{x}_i - \boldsymbol{x}_j)} \\ &= \sqrt{(\boldsymbol{L}\boldsymbol{x}_i - \boldsymbol{L}\boldsymbol{x}_j)^T (\boldsymbol{L}\boldsymbol{x}_i - \boldsymbol{L}\boldsymbol{x}_j)}. \end{aligned}$$

Therefore, computing the Mahalanobis distance in the original space is same as computing the Euclidean distance in the transformed space after linearly mapping the data by the transformation matrix $\boldsymbol{L}$. If $\boldsymbol{M}$ is low rank, the data is mapped into a space of lower dimension. Based on this equivalence, metric learning can be approached by either learning the distance matrix $\boldsymbol{M}$ or learning the linear transformation matrix $\boldsymbol{L}$.

2.1.2 Formulation of supervised Mahalanobis distance learning

Research on selecting optimal distance metrics for the nearest neighbour classifier dates back to the 1980s [8]. The recent resurgence of research interest in this field attributes to [1], which first proposes to formulate the metric learning task as a convex optimisation problem. Most later works on supervised Mahalanobis distance learning fall into this optimisation framework; they vary in loss functions and regularisation terms to better align with the side information on the ideal distance.

Two common forms of side information are pairwise constraints and triplet constraints [6]:

- pairwise constraints[1]:

$$\mathcal{S} = \{(\boldsymbol{x}_i, \boldsymbol{x}_j) : \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ should be similar}\},$$
$$\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{x}_j) : \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ should be dissimilar}\}.$$

- triplet constraints:

$$\mathcal{R} = \{(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_l) : \boldsymbol{x}_i \text{ should be more similar to } \boldsymbol{x}_j \text{ than to } \boldsymbol{x}_l\}.$$

The optimisation formulation of supervised Mahalanobis distance learning consists of a loss function that encodes the supervision and a regulariser on the metric:

$$\min_{\boldsymbol{M} \in \text{dom}(\boldsymbol{M})} \ell(\boldsymbol{M}; \mathcal{S}, \mathcal{D}, \mathcal{R}) + \lambda R(\boldsymbol{M}). \tag{2.2}$$

dom$(\boldsymbol{M})$ specifies the domain of the metric, which is generally defined as the space of positive semi-definite matrices. The loss function $\ell(\boldsymbol{M}; \mathcal{S}, \mathcal{D}, \mathcal{R})$ will assign a penalty if the learned metric violates the given pairwise or triplet constraints. The regularisation term $R(\boldsymbol{M})$ will constrain the complexity of the learned metric so that it can generalise well to the unseen data. λ is the trade-off parameter between the loss and the regulariser.

2.1.3 Representative metric learning algorithms

A large number of metric learning algorithms have been proposed since the proposal of [1] to improve the discriminability, generalisability and robustness of the metric, efficiency for large-scale and high-dimensional data, and applicability to specific classification tasks, such as multi-label problems and multi-task problems.

[1]The notation $\mathcal{D}$ is used only in this section for dissimilar constraints to accord with the metric learning literature; it will be used to denote the distribution in the rest of the thesis.

A comprehensive survey of methods is provided in [9]. Here, we present three seminal algorithms, which are representatives of learning the metric from pairwise constraints, learning from triplet constraints, and learning in a probabilistic framework; the latter two algorithms are the foundations of Chapters 3, 4 and Chapter 5, respectively.

In the following discussion, we denote a set of training examples by $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ with instances $\boldsymbol{x}_i \in \mathbb{R}^p$ and labels $y_i \in \{1, 2, \dots, C\}$.

Mahalanobis metric for clustering (MMC) [1] MMC aims to minimise the distances between pairs of similar instances and maximises the distance between pairs of dissimilar instances:

$$\begin{aligned} \min_{\boldsymbol{M}} & \sum\nolimits_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ \text{s.t.} & \sum\nolimits_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{D}} d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 1, \\ & \boldsymbol{M} \in \mathbb{S}^p_+. \end{aligned} \tag{2.3}$$

By using the squared Mahalanobis distance, the objective function is linear in $\boldsymbol{M}$. Moreover, the two constraints are both convex. Therefore, the optimisation problem is convex. The similar set is generated by randomly sampling pairs of instances with the same class label, i.e. $\mathcal{S} = \{(\boldsymbol{x}_i, \boldsymbol{x}_j) : y_i = y_j\}$; the dissimilar set is generated from instances of different labels, i.e. $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{x}_j) : y_i \neq y_j\}$.

Large margin nearest neighbor (LMNN) [10] MMC and other early approaches minimises the distance between all similar pairs, whereas the subsequent k-nearest neighbour (kNN) classifier uses only the nearest k instances for prediction. This local property of kNN is taken account into LMNN, which has the following objective function:

$$\min_{\boldsymbol{M}\in\mathbb{S}^P_+} (1-\mu) \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) + \mu \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j,\boldsymbol{x}_l)\in\mathcal{R}} \left[1 + d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l)\right]_+, \tag{2.4}$$

where $[a]_+ = \max(a, 0)$ for $a \in \mathbb{R}$. The pairwise and triplet constraints are defined as follows:

$$\mathcal{S} = \left\{(\boldsymbol{x}_i, \boldsymbol{x}_j) : j \in \underset{a=1,\cdots,n}{\arg\min}_k\{d_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_a) : y_i = y_a\}\right\},$$
$$\mathcal{R} = \left\{(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_l) : (\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{S}, y_i \neq y_l\right\},$$

where $\arg\min_k$ denotes the k minimal elements of a set and $d_{\boldsymbol{E}}$ denotes the Euclidean distance.

The first part of Eq. 2.4 is designed to pull together k nearest neighbours with the same label. The second part is designed to push all instances with different labels outside the local neighbourhood plus a unit *margin*; the margin acts a safeguard against noise. $\mu \in [0, 1]$ is the trade-off parameter, balancing the effects of shrinking and expanding distances. $\boldsymbol{x}_j$ in the set $\mathcal{S}$ is termed the *target neighbour* of $\boldsymbol{x}_i$ and $\boldsymbol{x}_l$ is termed the *impostor*. An illustration of LMNN is given in Figure 2.1.

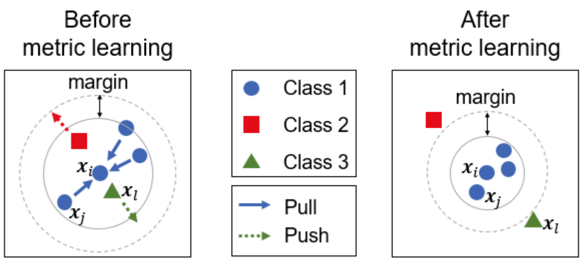


Figure 2.1: Illustration of LMNN. The objective is to bring the target neighbour $\boldsymbol{x}_j$ closer to the instance $\boldsymbol{x}_i$ and push the impostor $\boldsymbol{x}_l$ outside the local neighbourhood of $\boldsymbol{x}_i$ (indicated by the circle) plus a margin. The figure is reproduced from [10].

Neighbourhood components analysis (NCA) [11] NCA is a probabilistic approach to metric learning with a direct objective of maximising the test accuracy of NN classifier. Since the true data distribution is unknown, the method instead maximises a smooth approximation of the leave-one-out classification accuracy on the training data:

$$\max_{\boldsymbol{L}} \sum_i \sum_j p_{ij} \mathbb{1}[y_i = y_j], \tag{2.5}$$

where

$$p_{ij} = \frac{\exp(-\|\boldsymbol{L}\boldsymbol{x}_i - \boldsymbol{L}\boldsymbol{x}_j\|_2^2)}{\sum_{k \neq i} \exp(-\|\boldsymbol{L}\boldsymbol{x}_i - \boldsymbol{L}\boldsymbol{x}_k\|_2^2)}, \quad p_{ii} = 0, \tag{2.6}$$

and $\mathbb{1}[\cdot]$ denotes the indicator function. p_{ij} defines the probability of selecting instance $\boldsymbol{x}_j$ as the nearest neighbour of $\boldsymbol{x}_i$ by applying a softmax function to the Euclidean distances in the linearly transformed space. In addition to the above objective which maximises the expected number of correct classification, [11] proposes an alternative objective which maximises the log-likelihood that all instances select same-class points as their neighbours:

$$\max_{\boldsymbol{L}} \sum_i \log \Big(\sum_j p_{ij} \mathbb{1}[y_i = y_j] \Big). \tag{2.7}$$

Eqs. 2.5 and 2.7 can also be interpreted as minimising the L_1-norm and the Kullback-Leibler divergence from the estimated probability distribution built on p_{ij} to the observed distribution represented as a one-hot vector, respectively.

2.2 Generalisation and robustness

2.2.1 The statistical learning framework

In this section, we review some basic concepts in the statistical learning framework; the terminology follows [12].

In a classification setting, a learning algorithm, or simply a learner, takes as input a training set and outputs a classifier. The training set $S = \{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_n, y_n)\}$ consists of n training examples that are independently and identically distributed (i.i.d.) according to an unknown probability distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$. $\mathcal{X}$ is set of of all possible *instances* and is referred as the *instance space*; each instance will be represented by a vector of *features*. $\mathcal{Y}$ is the set of all possible *labels*. The *classifier*, or more often known as the hypothesis in the literature, is a function from $\mathcal{X}$ to $\mathcal{Y}$, i.e. $h : \mathcal{X} \to \mathcal{Y}$. The set of all possible hypotheses is termed the hypothesis class and denoted by $\mathcal{H}$.

The goal of the learner is to find a classifier that minimises the expected risk. Let $\boldsymbol{z} = (\boldsymbol{x}, y) \in \mathcal{Z}$ denote an instance-label pair and ℓ denote a loss function. The *expected risk* (also known as the generalisation error or the true error) is defined as:

$$R(h) = \mathbb{E}_{\boldsymbol{z}\sim\mathcal{D}}[\ell(h, \boldsymbol{z})]. \tag{2.8}$$

Since the expected risk cannot be evaluated due to the unknown distribution $\mathcal{D}$, the learner will instead minimise the *empirical risk* (also known as the training error):

$$R_n(h) = \frac{1}{n}\sum_{i=1}^{n}[\ell(h, \boldsymbol{z}_i)]. \tag{2.9}$$

This learning paradigm is called the empirical risk minimisation (ERM).

Minimising the empirical risk does not necessarily lead to a low expected risk. The difference between these two quantities, termed the *generalisation gap*, can be bounded based on a number of techniques, some of which are presented in the following section.

2.2.2 Generalisation bound

The generalisation bound provides a finite sample guarantee on the deviation of the expected risk from the empirical risk. It can be established through tools such as the complexity of the hypothesis class [13], stability of an algorithm [14], and robustness of an algorithm [15]. In this section, we present two ways of bounding the generalisation gap – one based on the Rademacher complexity, which is one of the complexity measures, and one based on the algorithmic robustness.

2.2.2.1 Rademacher complexity-based generalisation bound [16]

The definition of Rademacher complexity is given as follows.

Definition 3. [12] Let $\mathcal{F}$ be a function class and $\mathcal{F} \circ S$ be the set of all possible evaluations a function f can achieve on a sample S of size n:

$$\mathcal{F} \circ S = \{(f(\boldsymbol{z}_1), \cdots, f(\boldsymbol{z}_n)) : f \in \mathcal{F}\}.$$

Let the variables in $\boldsymbol{\sigma}$ be i.i.d. according to $P(\sigma_i = 1) = P(\sigma_i = -1) = \frac{1}{2}$. Then, the *empirical Rademacher complexity* of $\mathcal{F}$ with respect to S is defined as:

$$\hat{\mathcal{R}}_n(\mathcal{F} \circ S) = \frac{1}{n} \mathop{\mathbb{E}}_{\boldsymbol{\sigma} \sim \{\pm 1\}^n} \left[\sup_{f \in \mathcal{F}} \sum_{i=1}^{n} \sigma_i f(\boldsymbol{z}_i) \right]. \tag{2.10}$$

The *Rademacher complexity* of $\mathcal{F}$ is the expectation of the empirical Rademacher complexity over all samples of size n drawn according to the distribution $\mathcal{D}$:

$$\mathcal{R}_n(\mathcal{F}) = \mathop{\mathbb{E}}_{S \sim \mathcal{D}^n} \left[\hat{\mathcal{R}}_n(\mathcal{F} \circ S) \right]. \tag{2.11}$$

The empirical Rademacher complexity can be intuitively understood as measuring how well a function class can fit random noise on average. A more complex function class can return an evaluation on a sample S that has a large inner product with a random sign vector on average.

The following theorem states that the generalisation gap is upper bounded by the Rademacher complexity of the hypothesis class.

Theorem 1. Assume $|\ell(h, \boldsymbol{z})| \leq c$ for all $\boldsymbol{z}$ and h. Then, with probability at least $1 - \delta$, for all $h \in \mathcal{H}$,

$$R(h) - R_n(h) \leq 2c\, \mathcal{R}_n(\mathcal{H}) + c\sqrt{\frac{2\ln(2/\delta)}{n}}, \tag{2.12}$$

$$R(h) - R_n(h) \leq 2c\, \hat{\mathcal{R}}_n(\mathcal{H} \circ S) + 4c\sqrt{\frac{2\ln(4/\delta)}{n}}. \tag{2.13}$$

In particular, this holds for $h = \text{ERM}_{\mathcal{H}}(S)$.

A merit of bounding the generalisation gap via the empirical Rademacher complexity, i.e. Eq. 2.13, is that the bound is data-dependent; the training set S is used both for learning the classifier and evaluating the generalisation performance. Such dependency can lead to a much tighter bound than Eq. 2.12 in practice.

2.2.2.2 Algorithmic robustness-based generalisation bound [15]

Originating from robust optimisation [17], algorithmic robustness is a relatively new technique to prove the generalisation bound. Robustness and a weaker notion of robustness termed pseudo-robustness are defined as follows.

Definition 4. An algorithm $\mathcal{A}$ is $(K, \epsilon(\cdot))$ robust for $K \in \mathbb{N}$ and $\epsilon(\cdot) : \mathcal{Z}^n \to \mathbb{R}$ if $\mathcal{Z}$ can be partitioned into K disjoint sets, denoted by $\{C_i\}_{i=1}^K$, such that the following holds for all $\boldsymbol{z}^n = \{\boldsymbol{z}_1, \dots, \boldsymbol{z}_n : \boldsymbol{z}_i \in \mathcal{Z}\}$:

$$\forall \boldsymbol{z} \in \boldsymbol{z}^n, \forall \boldsymbol{z}' \in \mathcal{Z}, \forall i = 1, \cdots, K : \text{if } \boldsymbol{z}, \boldsymbol{z}' \in C_i, \text{ then } |\ell(\mathcal{A}_{\boldsymbol{z}^n}, \boldsymbol{z}) - \ell(\mathcal{A}_{\boldsymbol{z}^n}, \boldsymbol{z}')| \leq \epsilon(\boldsymbol{z}^n),$$

where $\mathcal{A}_{\boldsymbol{z}^n}$ denotes the classifier learned on the set $\boldsymbol{z}^n$.

Definition 5. Let $\mathrm{card}(\cdot)$ denote the cardinality of a set. An algorithm $\mathcal{A}$ is $(K, \epsilon(\cdot), \hat{n}(\cdot))$ pseudo-robust for $K \in \mathbb{N}$, $\epsilon(\cdot) : \mathcal{Z}^n \to \mathbb{R}$ and $\hat{n}(\cdot) : \mathcal{Z}^n \to \{1, \dots, n\}$ if $\mathcal{Z}$ can be partitioned into K disjoint sets, denoted by $\{C_i\}_{i=1}^K$, such that for all $\boldsymbol{z}^n = \{\boldsymbol{z}_i\}_{i=1}^n$, there exists a subset of samples $\hat{\boldsymbol{z}}^n$ with $\mathrm{card}(\hat{\boldsymbol{z}}^n) = \hat{n}(\boldsymbol{z}^n)$ that the following holds:

$$\forall \boldsymbol{z} \in \hat{\boldsymbol{z}}^n, \forall \boldsymbol{z}' \in \mathcal{Z}, \forall i = 1, \cdots, K : \text{ if } \boldsymbol{z}, \boldsymbol{z}' \in C_i, \text{ then } |\ell(\mathcal{A}_{\boldsymbol{z}^n}, \boldsymbol{z}) - \ell(\mathcal{A}_{\boldsymbol{z}^n}, \boldsymbol{z}')| \leq \epsilon(\boldsymbol{z}^n).$$

Robustness requires a classifier, trained on *every* set of samples, to obtain similar performances between a training instance and a nearby test instance, while pseudo-robust requires the condition to hold for a *subset* of samples. One merit of studying the (pseudo) robustness property is that the above definitions do not restrict the samples $\boldsymbol{z}^n$ to be independent, and thus the analysis can be conducted in a non-i.i.d. setting.

Several theorems have been proved in [15]; here, we only present the following generalisation bound based on the pseudo-robustness in an i.i.d. setting.

Theorem 2. Assume $|\ell(h, \boldsymbol{z})| \leq c$ for all $\boldsymbol{z}$ and h. If a learning algorithm $\mathcal{A}$ is $(K, \epsilon(\cdot), \hat{n}(\cdot))$ pseudo-robust and the training set $\boldsymbol{z}^n$ is generated by n i.i.d. draws from $\mathcal{D}$, then with probability at least $1 - \delta$ we have

$$|R(\mathcal{A}_{\boldsymbol{z}^n}) - R_n(\mathcal{A}_{\boldsymbol{z}^n})| \leq \frac{\hat{n}(\boldsymbol{z}^n)}{n} \epsilon(\boldsymbol{z}^n) + c\left(\frac{n - \hat{n}(\boldsymbol{z}^n)}{n} + \sqrt{\frac{2K \ln 2 + 2\ln(1-\delta)}{n}}\right).$$

2.2.3 Adversarial robustness

While deep neural networks have achieved the state-of-the-art performance in many applications such as computer vision and speech recognition, they are shown to be vulnerable to adversarial perturbations, which are well-thought imperceptible perturbations of the input that cause erroneous prediction. Adversarial perturbations can only modify the legitimate test inputs; they cannot modify the training procedure. The perturbed inputs are termed *adversarial examples* [4]. To improve robustness against adversarial examples, *adversarial training* [5] has been shown to be one of the most effective methods. It learns a model by considering the worst-case perturbation of each training instance.

More mathematically, an adversarial example of size r for a given L_p-norm is said to exist if there exists $\boldsymbol{\delta}$ such that $\|\boldsymbol{\delta}\|_p \leq r$ and $h(\boldsymbol{x} + \boldsymbol{\delta}) \neq h(\boldsymbol{x})$. If no such $\boldsymbol{\delta}$ exists, then the model is said to achieve *certified robustness* [18]. The goal of adversarial training is to learn the parameters $\boldsymbol{\theta}$ that minimise the expected adversarial risk:

$$\min_{\boldsymbol{\theta}} \mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim\mathcal{D}} \Big[\max_{\boldsymbol{\delta}:\|\boldsymbol{\delta}\|_p \leq r} \ell(\boldsymbol{x} + \boldsymbol{\delta}, y; \boldsymbol{\theta}) \Big]. \tag{2.14}$$

Analogous to empirical risk minimisation, the empirical adversarial risk is minimised in practice.

We make a remark that the above definitions refer to norm-bounded adversarial examples. Other types of perturbations, such as change in light condition [19] and spatial transformation [20], have their corresponding definitions.

Chapter 3

Metric Learning Towards Certified Robustness

3.1 Introduction

Distance metric learning (DML) focuses on learning similarity or dissimilarity between data and it has been actively researched in classification and clustering [2; 3; 6], as well as domain-specific applications such as information retrieval [21; 22], computer vision [23; 24; 25] and bioinformatics [26]. A commonly studied distance metric is the generalised Mahalanobis distance, which defines the distance between any two instances $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathbb{R}^p$ as

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

where $\boldsymbol{M}$ is a positive semi-definite (PSD) matrix. Owing to its PSD property, $\boldsymbol{M}$ can be decomposed into $\boldsymbol{L}^T\boldsymbol{L}$ with $\boldsymbol{L} \in \mathbb{R}^{d\times p}$; thus the Mahalanobis distance is equivalent to the Euclidean distance $||\boldsymbol{L}\boldsymbol{x}_i - \boldsymbol{L}\boldsymbol{x}_j||_2^2$ in the linearly transformed feature space.

To learn a specific distance metric for each task, prior knowledge on instance similarity and dissimilarity should be provided as side information. Metric learning methods differ by the form of side information they use and the supervision encoded in similar and dissimilar pairs. For example, pairwise constraints enforce the distance between instances of the same class to be small (or smaller than a

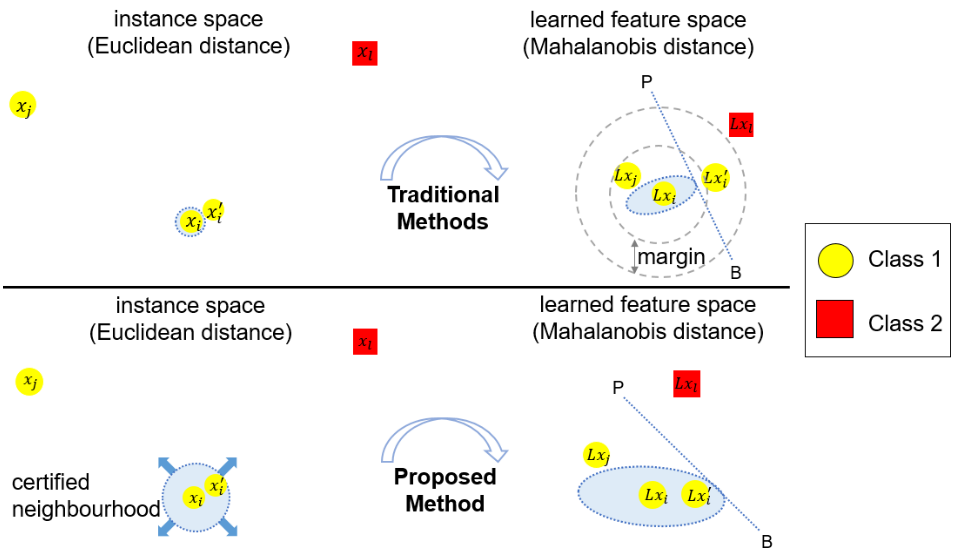


Figure 3.1: Comparison of traditional metric learning methods and the proposed method. While classical methods separate similar and dissimilar pairs by a margin (indicated by the gap between gray dashed circles), a small perturbation from $\boldsymbol{x}_i$ to $\boldsymbol{x}_i'$ in the instance space may change its nearest neighbour (NN) from $\boldsymbol{x}_j$ to $\boldsymbol{x}_l$ in the learned feature space. Our method aims to expand the certified neighbourhood (indicated by blue dotted circle), defined as the largest hypersphere in which $\boldsymbol{x}_i$ could be perturbed without any label change on its NN in the learned feature space. Points on line PB are equidistant from $\boldsymbol{x}_j$ and $\boldsymbol{x}_l$ with respect to the learned distance.

threshold value) and the distance between instances of different classes to be large (or larger than a threshold value) [1; 27; 28; 29; 30]. The thresholds could be either pre-defined or learned for similar and dissimilar pairs [31; 32]. In triplet constraints $(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_l)$, distance between the different-class pair $(\boldsymbol{x}_i, \boldsymbol{x}_l)$ should be larger than distance between the same-class pair $(\boldsymbol{x}_i, \boldsymbol{x}_j)$, and typically, plus a margin [10; 33; 34; 35]. More recently, quadruplet constraints are proposed, which require the difference in the distance of two pairs of instances to exceed a margin [36], and $(N+1)$-tuplet extends the triplet constraint for multi-class classification [37; 38].

The gap between thresholds in pairwise constraints and the margin in triplet and quadruplet constraints are both designed to learn a distance metric that could ensure good generalisation of the subsequent k-nearest neighbour (NN) classifier. However, such a separating margin imposed at the distance and decision level does not necessarily produce a robust metric – indeed it may be sensitive to a small perturbation at the instance level. As illustrated in Figure 3.1 (upper), a tiny perturbation from $\boldsymbol{x}_i$ to $\boldsymbol{x}'_i$ in the instance space can be magnified by the learned distance metric, leading to a change in its NN from $\boldsymbol{x}_j$ to $\boldsymbol{x}_l$ in the feature space, and even worse, an incorrect label prediction if 1NN is used.

In this chapter, we propose a simple yet effective method to enhance robustness of the learned distance metric against instance perturbation. The principal idea is to expand a *certified neighbourhood*, defined as the largest hypersphere in which a training instance could be perturbed without changing the label of its nearest neighbour (or k nearest neighbours if required) in the feature space.

Our contributions are mainly fourfold. Firstly, we derive an analytically elegant solution to the radius of certified neighbourhood (Section 3.2.1). It is equivalent to the distance between a training instance $\boldsymbol{x}_i$ and its nearest adversarial example [4] termed *support point*. Building on a geometric insight, the support point can be easily identified as the closest point to $\boldsymbol{x}_i$ in the instance space that lies on the decision boundary in the feature space. Secondly, we define a new perturbation loss that penalises the radius for being small, or equivalently, encourages an expansion of certified neighbourhood (Section 3.2.1), which can be optimised jointly with any existing triplet-based metric learning methods (Section 3.2.2). The optimisation problem suggests that our method learns a discriminative metric in a weighted manner and simultaneously imposes a data-dependent regularisation. Thirdly, because learning a distance metric for high-dimensional data may suffer from overfitting, we extend the perturbation loss so that the metric could be learned based on PCA transformed data in a low-dimensional subspace while retaining the ability to withstand perturbation in the original high-dimensional instance space (Section 3.2.3). Fourthly, we show the benefit of expanding a certified neighbourhood to the generalisation ability of the learned distance metric by using the theoretical technique of algorithmic robustness [15] (Section 3.2.4, Theorem 3). Experiments in noise-free and noisy settings show that the proposed method outperforms existing robust metric learning methods in terms of classification accuracy and validate its robustness to noise (Section 3.3).

Notation Let $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$ denote the set of training instance and label pairs, where $\boldsymbol{x}_i \in \mathcal{X} \subseteq \mathbb{R}^p$ and $y_i \in \mathcal{Y} = \{1, \ldots, C\}$; $\mathcal{X}$ is called the instance space. Our framework is based on triplet constraints $\{\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_l\}$ and we adopt the following strategy for generating triplets [10]:

$$\mathcal{S} = \big\{(\boldsymbol{x}_i, \boldsymbol{x}_j) : j \in \underset{a=1,\cdots,n}{\arg\min}_k \{d_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_a) : y_i = y_a\}\big\},$$
$$\mathcal{R} = \big\{(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_l) : (\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{S}, y_i \neq y_l\big\},$$

where $\arg\min_k$ denotes the k minimal elements of a set and $d_{\boldsymbol{E}}$ denotes the Euclidean distance. $\boldsymbol{x}_j$ is termed the target neighbour of $\boldsymbol{x}_i$ and $\boldsymbol{x}_l$ is termed the impostor. $d_{\boldsymbol{M}}$ denotes the Mahalanobis distance; $\boldsymbol{M} \in \mathbb{S}_+^p$, where $\mathbb{S}_+^p$ is the cone of $p \times p$ real-valued PSD matrices. $\boldsymbol{M}^2 = \boldsymbol{M}\boldsymbol{M}$. $|\mathcal{A}|$ denotes the cardinality of a set $\mathcal{A}$. $\mathbb{1}[\cdot]$ denotes the indicator function. $[a]_+ = \max(a, 0)$ for $a \in \mathbb{R}$.

3.2 Methodology

In this section, we will firstly derive an explicit formula for the support point and provide the rationale behind the advocated perturbation loss. Secondly, we will present how the introduced loss term can be used in conjunction with triplet-based metric learning methods such as large (distance) margin nearest neighbor (LMNN). Thirdly, we will incorporate the linear transformation induced by PCA into our framework, making the method suitable for high-dimensional data. Fourthly, we will show the benefit of the proposed method to the generalisation ability of the learned distance metric. Lastly, we will discuss the relationship between the proposed method and existing (robust) metric learning methods. Figure 3.2 illustrates the main concepts discussed in this chapter and Table 3.1 provides a detailed terminology list.

3.2.1 Support point and perturbation loss

As mentioned in the introduction, a learned distance metric may be sensitive to perturbation in the sense that a small change of the instance could alter its nearest neighbour in the learned feature space, from an instance of the same class to one

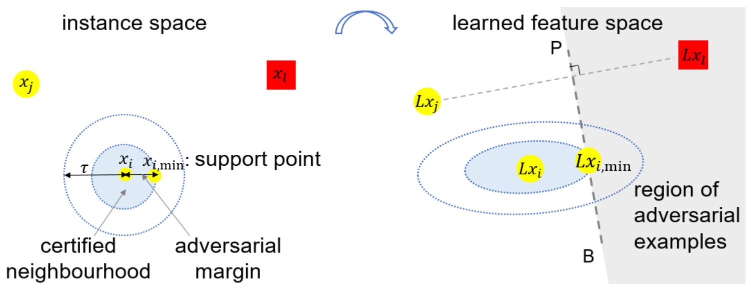


Figure 3.2: Explanation of main concepts: given a triplet constraint $(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_l)$, the decision boundary for $\boldsymbol{x}_i$ is the perpendicular bisector of $\boldsymbol{Lx}_j$ and $\boldsymbol{Lx}_l$, i.e. line PB. Points on the right-hand side of PB are adversarial examples. The support point is defined as the nearest adversarial example, i.e. the closest point to $\boldsymbol{x}_i$ in the instance space that its $\boldsymbol{L}$-transformed point lies on PB. The Euclidean distance between $\boldsymbol{x}_i$ and its support point is called adversarial margin, which is equivalent to the radius of certified neighbourhood; it will be enlarged to τ through penalising the proposed perturbation loss.

Table 3.1: Terminology list.

adversarial example	a perturbed instance; the perturbation changes the label of the instance's nearest neighbour (NN) in the feature space from being the same class to being a different class. In other words, the perturbation forces the NN classifier to produce an incorrect prediction [4].
support point ($\boldsymbol{x}_{i,\min}$)	the adversarial example that is closest to the training instance in the original instance space
certified neighbourhood	the largest hypersphere that a training instance could be perturbed while keeping its NN in the feature space to be an instance of the same class
adversarial margin ($d_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\min})$)	the Euclidean distance between a training instance and its associated support point. It defines the radius of certified neighbourhood.

of a different class, and consequently, increasing the risk of misclassification from kNN. A perturbed point, that causes a change in the nearest neighbours and thus prediction, is termed an *adversarial example* [4]; if the adversarial examples of an instance are all far away from the instance itself, a high degree of robustness is expected. Based on this reasoning, we will construct a loss function to penalise the small distance between a training instance $\boldsymbol{x}_i$ and its closest adversarial example (i.e. support point), and therefore, allowing $\boldsymbol{x}_i$ to retain prediction correctness even when perturbed to a larger extent.

We start by building a geometric insight into the support point: for any instance $\boldsymbol{x}_i$ associated with the triplet constraint $(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_l)$, the support point $\boldsymbol{x}_{i,\text{min}}$ is the closest point to $\boldsymbol{x}_i$ in the instance space that lies on the decision boundary formed by $\boldsymbol{x}_j$ and $\boldsymbol{x}_l$ in the feature space. Note that closeness is defined in the instance space and will be calculated using the Euclidean distance since we target at changes on the original feature of an instance; and that the decision boundary is found in the feature space since kNNs are identified by using the Mahalanobis distance. Mathematically, we can formulate the support point $\boldsymbol{x}_{i,\text{min}}$ as follows:

$$\begin{aligned} &\boldsymbol{x}_{i,\text{min}} = \arg\min_{\boldsymbol{x}'_i \in \mathbb{R}^p} (\boldsymbol{x}'_i - \boldsymbol{x}_i)^T \boldsymbol{A}_0 (\boldsymbol{x}'_i - \boldsymbol{x}_i) \\ &\text{s.t. } (\boldsymbol{L}\boldsymbol{x}'_i - \frac{\boldsymbol{L}\boldsymbol{x}_j + \boldsymbol{L}\boldsymbol{x}_l}{2})^T (\boldsymbol{L}\boldsymbol{x}_l - \boldsymbol{L}\boldsymbol{x}_j) = 0. \end{aligned} \tag{3.1}$$

With a pre-given positive definite matrix $\boldsymbol{A}_0$, the objective function of Eq. 3.1 defines an arbitrarily oriented hyperellipsoid, representing any heterogeneous and correlated perturbation. Without prior knowledge on the perturbation, we simplify $\boldsymbol{A}_0$ as the identity matrix. In this case, the objective function defines a hypersphere, representing perturbation of equal magnitude in all directions. It can also be interpreted as minimising the Euclidean distance from the training instance $\boldsymbol{x}_i$. Unless otherwise stated, we always refer the certified neighbourhood as the largest hypersphere; the hyperellipsoid case is discussed in Appendix 3.5.1. The constraint defines the decision boundary, which is the perpendicular bisector of points $\boldsymbol{L}\boldsymbol{x}_j$ and $\boldsymbol{L}\boldsymbol{x}_l$. In other words, it is a hyperplane that is perpendicular to the line joining points $\boldsymbol{L}\boldsymbol{x}_j$ and $\boldsymbol{L}\boldsymbol{x}_l$ and passes their midpoint $\frac{\boldsymbol{L}\boldsymbol{x}_j+\boldsymbol{L}\boldsymbol{x}_l}{2}$; all points on the hyperplane are equidistant from $\boldsymbol{L}\boldsymbol{x}_j$ and $\boldsymbol{L}\boldsymbol{x}_l$.

Since Eq. 3.1 minimises a convex quadratic function with an equality constraint, we can find an explicit formula for the support point $\boldsymbol{x}_{i,\text{min}}$ by using the method of Lagrangian multipliers; the detailed derivation is given in Appendix 3.5.1:

$$\boldsymbol{x}_{i,\text{min}} = \boldsymbol{x}_i + \frac{(\frac{\boldsymbol{x}_j+\boldsymbol{x}_l}{2} - \boldsymbol{x}_i)^T \boldsymbol{M} (\boldsymbol{x}_l - \boldsymbol{x}_j)}{(\boldsymbol{x}_l - \boldsymbol{x}_j)^T \boldsymbol{M}^2 (\boldsymbol{x}_l - \boldsymbol{x}_j)} \boldsymbol{M} (\boldsymbol{x}_l - \boldsymbol{x}_j). \tag{3.2}$$

With a closed-form solution of $\boldsymbol{x}_{i,\text{min}}$, we can now calculate the squared Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i,\text{min}}$:

$$d^2_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\text{min}}) = \frac{\left(d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)\right)^2}{4d^2_{\boldsymbol{M}^2}(\boldsymbol{x}_j, \boldsymbol{x}_l)}. \tag{3.3}$$

For clarity, we will call $d_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\text{min}})$ the *adversarial margin*, in contrast to the distance margin as in LMNN. It defines the radius of the certified neighbourhood.

To improve robustness of distance metric, we design a perturbation loss to encourage an expansion of certified neighbourhood. Two situations need to be distinguished here. Firstly, when the nearest neighbour of $\boldsymbol{x}_i$ is an instance from the same class, we will penalise a small adversarial margin by using the hinge loss $[\tau^2 - d^2_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\text{min}})]_+$. The reasons are that (a) the adversarial margin is generally smaller for hard instances that are close to the class boundary in contrast to those locating far away and (b) it is these hard instances that are more vulnerable to perturbation and demand an improvement in their robustness. Therefore, we introduce τ for directing attention to hard instances and controlling the desired margin. Secondly, in the other situation where the nearest neighbour of $\boldsymbol{x}_i$ belongs to a different class, metric learning should focus on satisfying the distance requirement specified in the triplet constraint. In this case, we simply assign a large penalty of τ^2 to promote a non-increasing loss function. Integrating these two situations leads to the proposed perturbation loss:

$$\begin{aligned} J_{\text{P}} = \frac{1}{|\mathcal{R}|} \sum_{\mathcal{R}} \Big\{ & [\tau^2 - \tilde{d}^2_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\text{min}})]_+ \mathbb{1}[d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) > d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)] \\ & + \tau^2 \mathbb{1}[d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) \leq d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)] \Big\}, \end{aligned} \tag{3.4}$$

where $\sum_{\mathcal{R}}$ is an abbreviation for $\sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_l) \in \mathcal{R}}$. To prevent the denominator of Eq. 3.3 from being zero, which may happen when different-class instances $\boldsymbol{x}_j$ and $\boldsymbol{x}_l$ are close to each other, we add a small constant ϵ ($\epsilon = 10^{-10}$) to the denominator; that is, $\tilde{d}^2_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\text{min}}) = \frac{\left(d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)\right)^2}{4\left(d^2_{\boldsymbol{M}^2}(\boldsymbol{x}_j, \boldsymbol{x}_l) + \epsilon\right)}$.

3.2.2 Metric learning towards certified robustness

As support points are derived from triplet constraints, it would be natural and straightforward to embed the proposed perturbation loss into a metric learning method that is also based on triplet constraints. LMNN is thus adopted as an example for its wide use and effective classification performance.

The objective function of the proposed LMNN towards certified robustness (LMNN-CR) is as follows:

$$\begin{aligned}&\min_{\boldsymbol{M}\in\mathbb{S}_+^p} J = J_{\text{LMNN}} + \lambda J_{\text{P}},\\ &J_{\text{LMNN}} = (1-\mu)\frac{1}{|\mathcal{S}|}\sum_{\mathcal{S}} d_{\boldsymbol{M}}^2(\boldsymbol{x}_i,\boldsymbol{x}_j) + \mu\frac{1}{|\mathcal{R}|}\sum_{\mathcal{R}}\left[1 + d_{\boldsymbol{M}}^2(\boldsymbol{x}_i,\boldsymbol{x}_j) - d_{\boldsymbol{M}}^2(\boldsymbol{x}_i,\boldsymbol{x}_l)\right]_+,\end{aligned} \tag{3.5}$$

where $\sum_{\mathcal{S}}$ stands for $\sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}}$. The weight parameter $\lambda > 0$ controls the importance of perturbation loss (J_{P}) relative to the loss function of LMNN (J_{LMNN}). $\mu \in (0,1)$ balances the impacts between pulling together target neighbours and pushing away impostors.

We adopt the projected gradient descent algorithm to solve the optimisation problem (Eq. 3.5). The gradient of J_{P} and J_{LMNN} are given as follows:

$$\begin{aligned}\frac{\partial J_{\text{P}}}{\partial \boldsymbol{M}} &= \frac{1}{|\mathcal{R}|}\sum_{\mathcal{R}}\alpha_{ijl}\Big\{\frac{d_{\boldsymbol{M}}^2(\boldsymbol{x}_i,\boldsymbol{x}_l) - d_{\boldsymbol{M}}^2(\boldsymbol{x}_i,\boldsymbol{x}_j)}{2\big(d_{\boldsymbol{M}^2}^2(\boldsymbol{x}_j,\boldsymbol{x}_l)+\epsilon\big)}(\boldsymbol{X}_{ij} - \boldsymbol{X}_{il})\\ &\qquad + \frac{\left(d_{\boldsymbol{M}}^2(\boldsymbol{x}_i,\boldsymbol{x}_l) - d_{\boldsymbol{M}}^2(\boldsymbol{x}_i,\boldsymbol{x}_j)\right)^2}{4\big(d_{\boldsymbol{M}^2}^2(\boldsymbol{x}_j,\boldsymbol{x}_l)+\epsilon\big)^2}(\boldsymbol{M}\boldsymbol{X}_{jl} + \boldsymbol{X}_{jl}\boldsymbol{M})\Big\},\\ \frac{\partial J_{\text{LMNN}}}{\partial \boldsymbol{M}} &= \frac{1-\mu}{|\mathcal{S}|}\sum_{\mathcal{S}}\boldsymbol{X}_{ij} + \frac{\mu}{|\mathcal{R}|}\sum_{\mathcal{R}}\beta_{ijl}(\boldsymbol{X}_{ij} - \boldsymbol{X}_{il}),\end{aligned}$$

where $\alpha_{ijl} = \mathbb{1}[d_{\boldsymbol{M}}^2(\boldsymbol{x}_i,\boldsymbol{x}_l) > d_{\boldsymbol{M}}^2(\boldsymbol{x}_i,\boldsymbol{x}_j), \tilde{d}_{\boldsymbol{E}}(\boldsymbol{x}_i,\boldsymbol{x}_{i,\text{min}}) \leq \tau], \beta_{ijl} = \mathbb{1}[1 + d_{\boldsymbol{M}}^2(\boldsymbol{x}_i,\boldsymbol{x}_j) - d_{\boldsymbol{M}}^2(\boldsymbol{x}_i,\boldsymbol{x}_l) \geq 0]$; $\boldsymbol{X}_{ij} = (\boldsymbol{x}_i - \boldsymbol{x}_j)(\boldsymbol{x}_i - \boldsymbol{x}_j)^T$ and $\boldsymbol{X}_{il}, \boldsymbol{X}_{jl}$ are defined similarly. The gradient of J_{P} is a sum of two descent directions. The first direction $\boldsymbol{X}_{ij} - \boldsymbol{X}_{il}$ agrees with LMNN, indicating that our method updates the metric toward better discrimination in a weighted manner. The second direction $\boldsymbol{M}\boldsymbol{X}_{jl} + \boldsymbol{X}_{jl}\boldsymbol{M}$ controls the scale of $\boldsymbol{M}$; the metric will descend at a faster pace in the direction of a larger correlation between $\boldsymbol{M}$ and $\boldsymbol{X}_{jl}$. This suggests our method functions as a data-dependent regularisation. Let $\boldsymbol{M}^t$ denote the Mahalanobis matrix learned at the tth iteration. The distance matrix will be updated as:

$$\boldsymbol{M}^{t+1} = \boldsymbol{M}^t - \gamma\Big(\frac{\partial J_{\text{LMNN}}}{\partial \boldsymbol{M}^t} + \lambda\frac{\partial J_{\text{P}}}{\partial \boldsymbol{M}^t}\Big),$$

where γ denotes the learning rate. To guarantee the PSD property, we factorize $\boldsymbol{M}^{t+1}$ as $\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^T$ via eigendecomposition and truncate all negative eigenvalues to zero, i.e. $\boldsymbol{M}^{t+1} = \boldsymbol{V}\max(\boldsymbol{\Lambda}, 0)\boldsymbol{V}^T$.

The proposed perturbation loss is a generic approach to improving robustness to perturbation. In Appendix 3.5.3, we give another example which incorporates the perturbation loss into sparse compositional metric learning (SCML) [39]; the new method is termed SCML towards certified robustness (SCML-CR). SCML is a recent triplet-based method that reduces the number of parameters and avoids the projection onto the PSD cone by representing the Mahalanobis distance metric as a sparse and non-negative combination of rank-one basis elements. The solution to the support point and the form of perturbation loss remain the same; the learning of the Mahalanobis distance is replaced by learning the sparse coefficients, and the optimisation problem is solved via the accelerated proximal gradient descent algorithm.

3.2.3 Extension to high-dimensional data

Learning a distance metric for extremely high-dimensional data will result in a large number of parameters to be estimated and potentially suffer from overfitting. In order to reduce the input dimensionality, PCA is often applied to pre-process the data prior to metric learning [10; 40]. In this subsection, we will extend the proposed method so that the distance metric learned in the low-dimensional PCA subspace could still achieve robustness against perturbation in the original high-dimensional instance space.

Defining perturbation loss in conjunction with PCA is realisable as our derivation builds on the linear transformation induced by the distance metric and PCA also performs a linear transformation to map data onto a lower dimension subspace. Let $\boldsymbol{D} \in \mathbb{R}^{d\times p}$ denote the linear transformation matrix obtained from PCA; p is the original feature dimension and d is the reduced feature dimension. With a slight abuse of terminology, datasets with $p \leq n$ is referred as high-dimensional so long as p is relatively large (e.g. $p^2 > 100n$). Following the same principle as before, the support point $\boldsymbol{x}_{i,\min}^{\mathrm{PCA}}$ should be the closest point to $\boldsymbol{x}_i$ in the original high-dimensional instance space and lie on the perpendicular bisector of points $\boldsymbol{LDx}_j$ and $\boldsymbol{LDx}_l$, i.e. after first mapping the data to a low-dimensional subspace by $\boldsymbol{D}$ and then mapping it to a feature space by $\boldsymbol{L}$. The mathematical formulation is as follows:

$$\begin{aligned} &\boldsymbol{x}_{i,\min}^{\mathrm{PCA}} = \arg\min_{\boldsymbol{x}_i'} (\boldsymbol{x}_i - \boldsymbol{x}_i')^T(\boldsymbol{x}_i - \boldsymbol{x}') \\ &\text{s.t. } (\boldsymbol{LDx}_i' - \frac{\boldsymbol{LDx}_j + \boldsymbol{LDx}_l}{2})^T(\boldsymbol{LDx}_l - \boldsymbol{LDx}_j) = 0 \end{aligned} \tag{3.6}$$

As shown in Appendix 3.5.2, $\boldsymbol{x}_{i,\min}^{\mathrm{PCA}}$ again has a closed-form solution and equations on the adversarial margin and perturbation loss can be extended accordingly.

3.2.4 Generalisation benefit

From the perspective of algorithmic robustness [15], enlarging the adversarial margin could potentially improve the generalisation ability of triplet-based metric learning methods. The following generalisation bound, i.e. the gap between the expected risk and the empirical risk, follows from the pseudo-robust theorem of [41]. Preliminaries and derivations are given in Appendix 3.5.4.

Theorem 3. Let $\boldsymbol{M}^*$ be the optimal solution to Eq. 3.5. Then with probability at least $1-\delta$ we have:

$$|R(\boldsymbol{M}^*) - R_n(\boldsymbol{M}^*)| \leq \frac{\hat{n}(t_{\boldsymbol{s}})}{n^3} + c\Big(\frac{n^3 - \hat{n}(t_{\boldsymbol{s}})}{n^3} + 3\sqrt{\frac{2K\ln 2 + 2\ln 1/\delta}{n}}\Big), \tag{3.7}$$

where $R(\boldsymbol{M}) = \mathbb{E}_{(\boldsymbol{z}_i,\boldsymbol{z}_j,\boldsymbol{z}_l)\sim\mathcal{D}}\ell(\boldsymbol{M};\boldsymbol{z}_i,\boldsymbol{z}_j,\boldsymbol{z}_l)$ denotes the expected risk, $R_n(\boldsymbol{M}) = \frac{1}{n(n-1)(n-2)}\sum_{i\neq j\neq l}\ell(\boldsymbol{M};\boldsymbol{z}_i,\boldsymbol{z}_j,\boldsymbol{z}_l)$ denotes the empirical risk, $\ell(\boldsymbol{M};\boldsymbol{z}_i,\boldsymbol{z}_j,\boldsymbol{z}_l) = [1 + d_{\boldsymbol{M}}^2(\boldsymbol{x}_i,\boldsymbol{x}_j) - d_{\boldsymbol{M}}^2(\boldsymbol{x}_i,\boldsymbol{x}_l)]_+ \cdot \mathbb{1}_{\{y_i=y_j\neq y_l\}}$ is the classical triplet loss, c is a constant denoting the upper bound of the loss function ℓ, $\hat{n}(t_{\boldsymbol{s}})$ denotes the number of triplets whose adversarial margins are larger than τ, and $K = |\mathcal{Y}|(1 + \frac{2}{\tau})^p$.

Enlarging the desired adversarial margin τ will reduce the value of K and $\hat{n}(t_{\boldsymbol{s}})$ in Eq. 3.7. On the one hand, K decreases with τ at a polynomial rate of the input dimensionality p and hence the upper bound of generalisation gap reduces at a rate of $p^{1/2}$. On the other hand, the reduction in $\hat{n}(t_{\boldsymbol{s}})$ increases the upper bound. However, $\hat{n}(t_{\boldsymbol{s}})$ remains relatively stable when τ increases as long as most instances in the dataset do not have a small margin in the original instance space. Therefore, for this type of dataset, we expect an improvement in the generalisation ability of the learned distance metric from enlarging the adversarial margin.

3.2.5 Relationship with other metric learning methods

In this subsection, we will discuss the connection between the introduced adversarial margin and the (distance) margin in conventional metric learning methods, followed by a comparison between our method and some other robust metric learning methods.

Our method imposes a safety margin in the original instance space while traditional metric learning methods target at a margin in the feature space. These two objectives can be linked by the Lipschitz constant of the Mahalanobis distance. The involved Lipschitz constant, as formally defined below, bounds the magnitude of change in the output $f(u)$ for any change in the input u:

Definition 6. [42] Given two metric spaces $(\mathcal{U}, \rho_{\mathcal{U}})$ and $(\mathcal{V}, \rho_{\mathcal{V}})$, the Lipschitz constant of a function $f : \mathcal{U} \to \mathbb{R}$ is defined to be the smallest $L > 0$ that satisfies $|f(u) - f(v)| \leq L\rho(u, v)$ for all $u, v \in \mathcal{U}$.

Recall that the objective function associated with the triplet loss in LMNN is

$$d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) \geq d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) + 1.$$

Let $\Delta\boldsymbol{x}_i$ denote a perturbation of $\boldsymbol{x}_i$ and $f(\boldsymbol{x}) = d^2_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}_l) - d^2_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}_j)$. When the Lipschitz constant of the Mahalanobis distance and hence of $f(\boldsymbol{x})$ is large, a small change from $\boldsymbol{x}_i$ to $\boldsymbol{x}_i + \Delta\boldsymbol{x}_i$ is likely to cause a large difference between

$f(\boldsymbol{x}_i)$ and $f(\boldsymbol{x}_i + \Delta\boldsymbol{x}_i)$. In other words, the fixed margin 1 imposed in the feature space may be insufficient to offset instance perturbation and guarantee classification performance. Therefore, we advocate maximising the margin in the instance space directly and thus propose the perturbation loss.

To improve robustness to perturbation that is likely to exist in practice, many robust metric learning methods have been proposed, which can be categorised into three main types. The first type of methods imposes structural assumption or regularisation over $\boldsymbol{M}$ so as to avoid overfitting [43; 44; 45; 46; 40; 47; 48]. Methods with structural assumption are proposed for classifying images and achieve robustness by exploiting the structural information of images; however, such information is generally unavailable in the symbolic datasets that will be studied in this chapter. Regularisation-based methods are proposed to reduce the risk of overfitting to feature noise. Our proposal, which is aimed to withstand perturbation, does not conflict with these methods and can be combined with them to learn a more effective and robust distance metric; an example is shown in Section 3.3.3. The second type of methods explicitly models the perturbation distribution or identifies clean latent examples [49; 50]. The expected Mahalanobis distance is then used to adjust the value of separating margin. The third type of methods generates hard instances through adversarial perturbation and trains a metric to fare well in the new hard problem [51; 52]. Although sharing the aim of improving metric robustness, these methods approach the task at a data-level by synthesising real examples that incur large losses, while our method tackles perturbation at a model-level by designing a loss function that considers the definition of robustness with respect to the decision maker kNN. By preventing change in the nearest neighbour in a strict manner, our method can obtain a certification on the adversarial margin.

Finally, we note that a large margin in the instance space has been studied in deep neural networks for enhancing robustness and generalisation ability [53; 54; 55; 56]. In contrast, our paper investigates such margin in the framework of metric learning, defines it specifically with respect to the NN classifier, and provides an exact and analytical solution to the margin.

3.3 Experiments

In this section, we first visualise the effect of the perturbation loss on a synthetic dataset. Next, we evaluate the generalisation performance and robustness of proposed method on 12 benchmark datasets (10 low/medium-dimensional and two high-dimensional). Finally, we discuss the computational aspect of the proposed method.

3.3.1 Synthetic data

We synthesise a two-dimensional dataset to illustrate the effect of the perturbation loss on the adversarial margin. The dataset includes two classes of ten instances each, simulated from Gaussian distributions. The mean vectors of the positive and negative classes are set as $[0.4, 0.4]$ and $[-0.4, -0.4]$ respectively, and the covariance matrices for both classes are set as $[1, -0.5; -0.5, 1]$. Mean centring and standardisation are performed prior to metric learning. Experimental settings such as the number of target neighbours and the ranges of hyperparameters are described in the subsequent section. Hyperparameters are selected on a separate validation set of 10,000 instances.

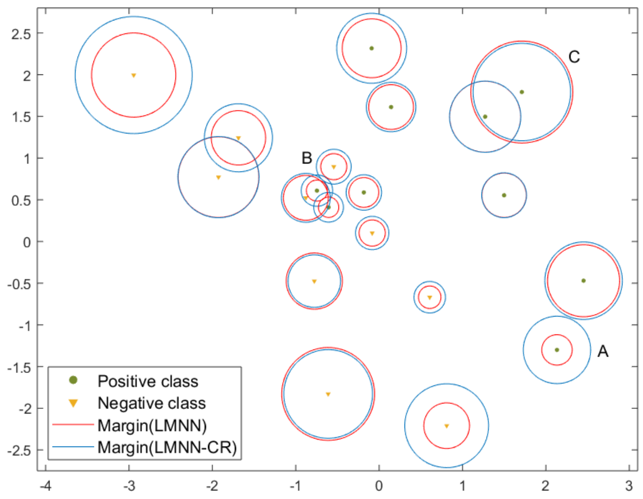


Figure 3.3: Visualisation and comparison of adversarial margins of LMNN (without) and LMNN-CR (with the proposed perturbation loss).

In Figure 3.3, we plot the adversarial margins of training instances after learning the Mahalanobis distance via LMNN or LMNN-CR; that is, $d_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{\text{min}})$ calculated with respect to different $\boldsymbol{M}$. For clarity, the margin size is shrunk by 70 percent of its original size for all instances. For easy instances whose nearest neighbours are instances of the same class (e.g. A), incorporating the perturbation loss enlarges the margin by a large magnitude. For hard instances which are located close to instances of the opposite class (e.g. B), the margin still gets enlarged although to a less extent. For instances whose margin is already quite large under LMNN (e.g. C), LMNN-CR may not further enlarge their margins, and even shrink the margins in some cases, which agrees with the design of the desired margin τ.

Appendix 3.5.5.2 presents two more simulation studies to illustrate the difference in the learning mechanism of LMNN and LMNN-CR.

3.3.2 Experiments on UCI data

3.3.2.1 Data description and experimental setting

We evaluate the proposed LMNN-CR and SCML-CR on 10 UCI datasets [57]. Information on sample size, feature dimension and class information is listed in Table 3.2. All datasets are pre-processed with mean-centring and standardisation, followed by L_2 normalisation to unit length. We use 70-30% training-test partitions

Table 3.2: Characteristics of the datasets.

Dataset	#Instances (training,test)	#Features (reduced dimensions)	#Classes	#Rounds
		UCI data		
Australian	690	14	2	20
Breast cancer	683	9	2	20
Fourclass	862	2	2	20
Haberman	306	3	2	20
Iris	150	4	3	20
Segment	2310	19	7	20
Sonar	208	60	2	20
Voting	435	16	2	20
WDBC	569	30	2	20
Wine	178	13	3	20
		High-dimensional data		
Isolet	7797 (1560,1558)	617 (170)	26	4
MNIST	4000 (2000,2000)	784 (141)	10	1

and report the average result over 20 rounds of random split.

The proposed methods are compared with two types of methods. First, we consider different regularisers on $\boldsymbol{M}$. Specifically, we replace the regulariser in LMNN from $\sum_{S} d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ to the log-determinant divergence (LDD) [31], which encourages learning a metric toward the identity matrix, and to the capped trace norm (CAP) [40], which encourages a low-rank matrix. Second, we compare with the method DRIFT [49], which models the perturbation distribution explicitly. We also report the performance of adversarial metric learning (AML) [51]. However, it is not directly comparable to our method as it learns from pairwise constraints and is based on a different backbone of geometric mean metric learning [58].

Hyperparameters of our methods are tuned via random search [59]. We randomly sample 50 sets of values from the following ranges: $\mu \in U(0.1, 0.9)$, $\tau \in U\left(0, P_{90\%}\{d_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\text{min}})\}\right)$, $\lambda \in U(0, 4/\tau^2)$. $U(a, b)$ denotes the uniform distribution. $P_{k\%}\{d_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\text{min}})\}$ denotes the kth percentile of $d_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\text{min}})$, where the distance is calculated for all i in the triplet constraint with respect to the Euclidean distance. Setting the upper bound of the adversarial margin τ via the percentile avoids unnecessary large values, matching our intention to expand the certified neighbourhood primarily for hard instances. The upper bound of the weight parameter λ depends on the realisation of τ to ensure that magnitudes of perturbation loss and LMNN loss are at the same level. SCML-CR is tuned in the same manner. The optimal hyperparameters from five-fold cross-validation on the training data are used to learn the metric. More details on the training procedure of the proposed and other methods are given in Appendix 3.5.5.1. The MATLAB code for our method is available at `http://github.com/xyang6/LMNNPL.`

In all experiments, triplet constraints are generated from 3 target neighbours and 10 nearest impostors, calculated under the Euclidean distance. We use 3NN as the classifier and accuracy as the evaluation criterion.

3.3.2.2 Evaluation on classification performance

Table 3.3 reports the mean value and standard deviation of classification accuracy. LMNN-CR outperforms LMNN on 9 out of 10 datasets. Among the methods with

Table 3.3: Classification accuracy (mean±standard deviation) of 3NN on clean datasets.

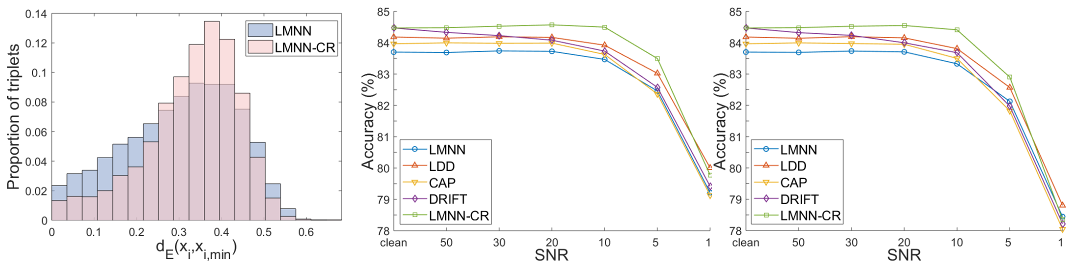
		LMNN-based					SCML-based	
Dataset	AML	LMNN	LDD	CAP	DRIFT	LMNN-CR	SCML	SCML-CR
Australian	83.25±2.59	83.70±2.43	84.18±2.37	83.97±2.45	**84.47±2.02**	**84.47±1.63**	**84.76±2.08**	84.42±2.18
Breast cancer	97.10±1.21	**97.12±1.25**	96.95±1.51	97.00±1.08	96.98±1.16	97.02±1.30	97.00±1.09	**97.07±1.24**
Fourclass	75.12±2.35	75.10±2.31	**75.15±2.32**	75.02±2.48	75.08±2.34	75.12±2.35	75.10±2.27	**75.12±2.35**
Haberman	72.58±4.00	72.19±3.89	72.42±3.95	71.52±3.54	72.02±3.94	**72.64±4.29**	**72.75±3.79**	72.36±4.38
Iris	87.00±5.41	87.11±5.08	**87.67±4.70**	86.67±5.49	85.89±4.46	87.33±4.73	86.89±6.40	**87.44±5.31**
Segment	95.21±0.72	95.31±0.89	95.58±0.81	95.51±0.70	**95.75±0.65**	95.64±0.83	92.61±6.65	**93.95±1.47**
Sonar	84.13±4.86	86.67±4.10	87.22±3.90	87.22±4.38	86.19±4.43	**87.78±3.53**	82.38±4.15	**84.13±4.61**
Voting	95.34±1.64	95.80±1.78	95.80±1.41	95.92±1.45	95.31±1.32	**96.15±1.56**	95.84±1.58	**96.26±1.28**
WDBC	96.93±1.39	96.99±1.30	96.96±1.43	96.99±1.51	96.70±1.16	**97.13±1.33**	**97.25±1.30**	**97.25±1.52**
Wine	97.13±1.75	97.31±1.94	96.67±1.76	96.85±2.26	**97.69±1.79**	**97.69±1.89**	**97.69±1.79**	97.22±2.04
#Outperform	-	9	8	10	9	-	7	-

For methods with LMNN as the backbone, the best ones are shown in bold and the second best ones are underlined; for methods with SCML as the backbone, the best ones are shown in bold. '#Outperform' counts the number of datasets where LMNN-CR (SCML-CR, resp.) outperforms or performs equally well with LMNN-based (SCML, resp.) methods.

LMNN as the backbone, our method achieves the highest accuracy on 6 datasets and second highest accuracy on the remaining 4 datasets. SCML-CL outperforms or performs equally well with SCML on 7 datasets. These experimental results demonstrate the benefit of perturbation loss to generalisation of the learned distance metric.

3.3.2.3 Investigation into robustness

In this section, we evaluate robustness of metric learning methods against instance perturbation. To start with, we conduct an in-depth experiment on the Australian dataset to investigate the relationship between the proposed perturbation loss, adversarial margin and robustness. First, we compare the adversarial margins obtained from LMNN and LMNN-CR. Figure 3.4a provides a histogram of adversarial margin for all triplets that satisfy $d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) < d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l)$. Triplets that do not satisfy the constraint are not taken into account since in this case, by design of our perturbation loss, LMNN-CR will focus on correcting the nearest neighbour and improving classification accuracy rather than on enlarging the adversarial margin and improving robustness. Instances whose values of $d_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\text{min}})$ are close to zero are incapable of defending perturbation. After learning with the proposed loss (LMMN-CR), we see that nearly half of these vulnerable instances have a larger adversarial margin. Next, we test the robustness performance by adding Gaussian noise to test data. We consider the following two types of zero-mean Gaussian noise



(a) Histogram of adversarial margins after metric learning from LMNN and LMNN-CR. **(b)** Performance of LMNN-based methods under different levels of spherical Gaussian noise. **(c)** Performance of LMNN-based methods under different levels of Gaussian noise.

Figure 3.4: Investigation into robustness on the Australian dataset.

– Gaussian with a diagonal covariance matrix and equal variances (abbreviated to spherical Gaussian), and Gaussian with a diagonal covariance matrix and unequal variances (Abbr. Gaussian). The noise intensity is controlled via the signal-to-noise ratio (SNR). In addition, considering the small sample size of UCI datasets, we augment test data by adding multiple rounds of random noise until its size reaches 10,000. Figure 3.4b plots the classification accuracy of LMNN-based methods under different levels of spherical Gaussian noise. When the noise intensity is low, the performance of LMNN and LMNN-CR remain stable. When the noise intensity increases to the SNR of 10 dB or 5 dB, the performances of both method degrade. Owing to the enlarged adversarial margin, the influence on LMNN-CR is slightly smaller than that on LMNN. When the SNR equals 1 dB, the performance gain from using LMNN-CR becomes smaller. This result is reasonable as the desired margin τ is selected according to the criterion of classification accuracy and hence may be too small to withstand a high level of noise. LMNN-CR surpasses all other LMNN-based methods until the noise intensity is very large. Figure 3.4c plots the accuracy under Gaussian noise. Compared with the case of spherical Gaussian, the degradation of all methods is more pronounced in this case, but the pattern remains similar.

We now turn to test robustness on all datasets. Table 3.4 reports the classification accuracy after contaminating the test data by the Gaussian noise of 5 dB. LMNN-CR and SCML-CR improve the robustness of the corresponding baselines

Table 3.4: Classification accuracy of 3NN on datasets contaminated with Gaussian noise (SNR=5 dB).

		LMNN-based					SCML-based	
Dataset	AML	LMNN	LDD	CAP	DRIFT	LMNN-CR	SCML	SCML-CR
Australian	82.26±1.62	82.13±1.52	82.57±1.55	81.82±1.52	81.97±1.53	**82.90±1.53**	82.59±1.70	**82.84±1.64**
Breast cancer	96.70±1.01	96.24±1.06	96.66±1.07	96.27±1.03	96.61±0.97	**96.69±1.07**	96.34±1.03	**96.63±1.03**
Fourclass	69.00±1.06	67.74±1.25	68.84±1.14	67.84±1.19	**69.13±1.02**	69.04±1.11	68.22±1.10	**68.96±1.11**
Haberman	70.21±1.84	70.21±1.84	**70.25±1.82**	69.39±2.06	69.31±2.50	**70.25±1.90**	69.98±1.64	**70.24±1.85**
Iris	79.07±3.25	78.75±2.96	79.04±3.17	77.90±3.31	78.57±3.09	**79.20±3.08**	78.32±3.60	**79.18±3.13**
Segment	85.87±0.70	79.03±3.37	83.49±1.17	82.77±2.49	**83.88±1.33**	82.13±2.70	61.28±9.78	**62.86±8.76**
Sonar	83.50±3.38	83.54±4.30	**86.18±2.93**	85.44±2.79	84.65±3.30	84.99±3.13	76.91±4.32	**79.49±3.80**
Voting	94.10±1.07	94.01±1.00	94.24±1.13	94.37±1.17	93.94±1.12	**94.64±1.21**	93.99±1.15	**94.65±1.09**
WDBC	96.47±1.12	92.01±1.65	**96.30±0.94**	96.14±1.11	96.02±0.88	96.07±0.89	95.75±1.29	**96.22±1.14**
Wine	95.03±1.14	93.27±1.62	93.97±1.38	93.87±1.49	**94.55±1.15**	94.44±1.21	93.92±1.55	**94.52±1.33**
#Outperform	-	10	6	7	7	-	10	-

on all datasets. This clearly demonstrates the efficacy of adding perturbation loss for improving robustness against instance perturbation. Moreover, LMNN-CR is superior to the existing robust metric learning methods CAP and DRIFT on 7 datasets. The method LDD is also quite robust to perturbation. However, this should not be surprising as it encourages learning a metric close to the Euclidean distance, and the Euclidean distance is less sensitive to perturbation than the discriminative Mahalanobis distance. The performance under spherical Gaussian noise is very similar to the Gaussian noise and is reported in Appendix 3.5.5.3.

3.3.3 Experiments on high-dimensional data

In Section 3.2.3, we extend LMNN-CR for high-dimensional data. To verify its effectiveness, we test it on the following two datasets:

1) MNIST-2k [60]: The dataset includes the first 2,000 training images and first 2,000 test images of the MNIST database. We apply PCA to reduce the feature dimension from 784 to 141, accounting for 95% of total variance. All methods are trained and tested once on the pre-given training/test partition.

2) Isolet [57]: The dataset is a spoken letter database and is available from UCI. It includes 7,797 instances, grouped into four training sets and one test set. The original feature dimension is 617, and the leading 170 principal components which explain 95% of total variance are retained. All methods are trained four times, one time on each training set, and evaluated on the pre-

Table 3.5: Generalisation and robustness of DML methods on high-dimensional datasets.

Isolet						
Method	Clean	SG,SNR=20 (0.0809)	SG,SNR=5 (0.4233)	G,SNR=20 (0.0588)	G,SNR=5 (0.3181)	Adv. margin
LMNN	90.14±4.45	90.09±4.15	86.02±3.48	90.17±4.03	87.81±3.87	0.1095
LMNN-CR	91.08±3.71	91.02±3.77	87.91±3.30	91.05±3.73	89.40±3.76	0.1249
CAP	91.05±3.66	91.13±3.85	88.97±4.00	91.10±3.73	89.90±3.87	0.1514
CAP-CR	91.58±3.96	91.52±3.86	89.91±3.74	91.47±3.91	90.65±3.73	0.1559
SCML	90.73±4.10	90.33±4.21	86.50±4.18	90.51±4.14	88.50±3.71	0.0683
SCML-CR	90.83±4.16	90.67±4.12	86.55±3.75	90.83±4.16	88.41±4.07	0.0822

MNIST						
Method	Clean	SG,SNR=20 (0.0540)	SG,SNR=5 (0.2939)	G,SNR=20 (0.0649)	G,SNR=5 (0.3482)	Adv. margin
LMNN	90.55	90.00	88.40	90.10	88.40	0.1528
LMNN-CR	91.15	91.35	90.80	91.45	90.35	0.2235
CAP	91.65	91.80	91.40	91.80	90.70	0.2219
CAP-CR	92.00	91.90	90.85	91.95	90.65	0.2264
SCML	88.95	88.75	87.35	88.85	86.45	0.1217
SCML-CR	89.15	89.20	88.50	89.35	88.05	0.1432

Columns 3-6 report methods' robustness against spherical Gaussian noise (SG) and Gaussian noise (G). Values in brackets give the average perturbation size, calculated as the mean value of the L_2-norm of noises ($\|\Delta \boldsymbol{x}_i\|_2$).

given test set.

In addition to aforementioned methods, we introduce CAP-CR, which comprises the triplet loss of LMNN, the proposed perturbation loss, and the regulariser of CAP. CAP enforces $\boldsymbol{M}$ to be low-rank, which is a suitable constraint for high-dimensional data. With the inclusion of the perturbation loss, we expect the learned compact metric to be more robust to perturbation. For a fair comparison, in CAP-CR, we use the same rank and regularisation weight as CAP, and tune τ, λ from 10 randomly sampled sets of values.

Table 3.5 compares the generalisation and robustness performance of LMNN, CAP, SCML and our method; the generalisation performance of other methods are inferior to LMNN-CR and are reported in Appendix 3.5.5.3. First, on both datasets, our method achieves higher clean accuracy than the baseline methods, validating its efficacy in improving the generalisation ability of the learned distance metric. Second, when the SNR is 20 dB, the average perturbation size is smaller than the average adversarial margin. In this case, our method maintains its superiority. When the SNR is 5 dB, the average perturbation size is larger than the average adversarial margin. Nonetheless, our method produces even larger gain in accuracy for LMNN on both datasets, for CAP on Isolet, and for SCML on MNIST. These results demonstrate that adversarial margin is indeed a contributing factor in achieving certified robustness. Third, CAP-CR obtains higher accuracy on both clean and noise-contaminated data than LMNN-CR. This supports our discussion in Section 3.2.5 that regularisation and perturbation loss impose different requirements on $\boldsymbol{M}$ and combining them has the potential for learning a more effective distance metric.

3.3.4 Computational cost

We now analyse the computational complexity of LMNN-CR. According to Eq. 3.6, our method requires additional calculations on $d^2_{\boldsymbol{M}^2}(\boldsymbol{x}_j, \boldsymbol{x}_l)$ and $\boldsymbol{M}\boldsymbol{X}_{jl}$. Given n training instances, k target neighbours and p features, the computational complexities of $d^2_{\boldsymbol{M}^2}(\boldsymbol{x}_j, \boldsymbol{x}_l)$ and $\boldsymbol{M}\boldsymbol{X}_{jl}$ are $O(np^2 + n^2p)$ and $O(n^2p^2)$, respectively. The total complexity of our method is $O(p^3 + n^2p^2 + kn^2p)$, same as that of LMNN.

Table 3.6 compares the running time of LMNN-based methods on four UCI datasets that are large in sample size or in dimensionality and two high-dimensional datasets. The computational cost of our method is comparable to LMNN.

Table 3.6: Average training time (in seconds) of LMNN-based methods.

	LMNN	LDD	CAP	DRIFT	LMNN-CR
Australian	13.44	0.83	3.07	1.00	2.15
Segment	27.48	10.45	11.47	5.12	19.54
Sonar	4.93	4.08	4.65	0.92	6.75
WDBC	9.38	2.94	5.22	5.12	8.17
Isolet	339.57	207.69	176.50	N/A	190.55
MNIST	369.55	68.98	180.68	37.51	391.04

3.3.5 Parameter sensitivity

The proposed LMNN-CR includes three hyperparameters – μ for the weight of similarity constraints, λ for the weight of the perturbation loss, and τ for the desired ad-

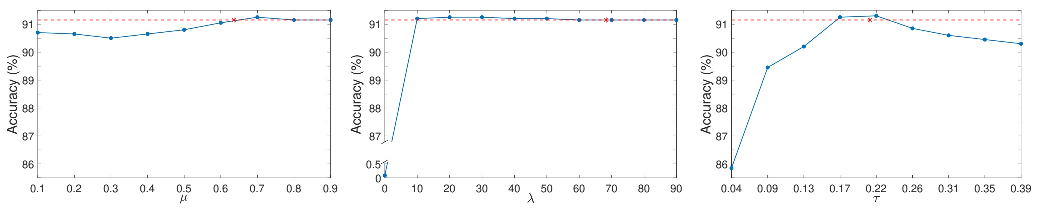


Figure 3.5: Sensitivity of LMNN-CR to hyperparameters (indicated by the straight line). The optimal accuracy and parameter value found via CV are indicated by the dashed line and asterisk, respectively.

versarial margin. We investigate their influences on the classification performance by varying one hyperparameter and fixing the other two at their optimal values. Figure 3.5 shows the accuracy on MNIST evaluated over the range of the hyperparameter. The performance changes smoothly with respect to μ. It is stable over a wide range of λ. When λ equals 0, LMNN-CR fails to learn a metric and returns a zero matrix. The performance is most affected by τ. Indeed, τ plays the central role in LMNN-CR as it determines the distribution of adversarial margins. Therefore, we shall strive to search for its optimal value.

3.4 Conclusions and future work

In this chapter, we demonstrate that robustness and generalisation of distance metrics can be enhanced by enforcing a larger margin in the instance space. By taking advantage of the linear transformation induced by the Mahalanobis distance, we obtain an explicit formula for the support points and push them away from training instances through penalising the perturbation loss. Extensive experiments verify that our method effectively enlarges the adversarial margin, achieves certified robustness, and sustains classification excellence.

The perturbation is assumed to be spherically distributed in the current study. Since this assumption does not always hold in practice, we may consider jointly learning the elliptical distribution and distance metric from the data. Another interesting future work will be extending the idea to nonlinear metric learning methods, as these methods generally have higher expressive power.

3.5 Appendix

3.5.1 Derivation of support point, adversarial margin, and gradient of perturbation loss

We define the hyperellipsoid via the quadratic form. An arbitrarily oriented hyperellipsoid, centred at $\boldsymbol{\mu} \in \mathbb{R}^p$, is defined by the solutions to the equation

$$\{\boldsymbol{x} \in \mathbb{R}^p : (\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{A}_0 (\boldsymbol{x} - \boldsymbol{\mu}) = r^2\},$$

where $\boldsymbol{A}_0$ is a positive definite matrix. By the Cholesky decomposition, $\boldsymbol{A}_0 = \boldsymbol{A}\boldsymbol{A}^T$. Therefore, finding the support point of $\boldsymbol{x}_i$ on the hyperellipsoid is equivalent to finding the point $\boldsymbol{x}'_i$ that defines the smallest hypersphere given by $(\boldsymbol{A}^T(\boldsymbol{x}'_i - \boldsymbol{x}_i))^T(\boldsymbol{A}^T(\boldsymbol{x}'_i - \boldsymbol{x}_i)) = r^2$. The optimisation problem of Eq. 3.1 is equivalent to the following problem:

$$\begin{aligned}
\boldsymbol{x}_{i,\min} &= \arg \min_{\boldsymbol{x}'_i \in \mathbb{R}^p} \left(\boldsymbol{A}^T(\boldsymbol{x}'_i - \boldsymbol{x}_i)\right)^T \left(\boldsymbol{A}^T(\boldsymbol{x}'_i - \boldsymbol{x}_i)\right) \\
\text{s.t. } &(\boldsymbol{L}\boldsymbol{x}'_i - \frac{\boldsymbol{L}\boldsymbol{x}_j + \boldsymbol{L}\boldsymbol{x}_l}{2})^T(\boldsymbol{L}\boldsymbol{x}_l - \boldsymbol{L}\boldsymbol{x}_j) = 0.
\end{aligned}$$

Applying the method of Lagrangian multiplier, we transform the above problem to the following Lagrangian function by introducing the Lagrangian multiplier λ and solve it by setting the first partial derivatives to zero:

$$\begin{aligned}
&\min_{\boldsymbol{x}'_i} \left(\boldsymbol{A}^T(\boldsymbol{x}'_i - \boldsymbol{x}_i)\right)^T \left(\boldsymbol{A}^T(\boldsymbol{x}'_i - \boldsymbol{x}_i)\right) - \lambda(\boldsymbol{L}\boldsymbol{x}'_i - \frac{\boldsymbol{L}\boldsymbol{x}_j + \boldsymbol{L}\boldsymbol{x}_l}{2})^T(\boldsymbol{L}\boldsymbol{x}_l - \boldsymbol{L}\boldsymbol{x}_j) \\
&\frac{\delta}{\delta \boldsymbol{x}'_i} : 2\boldsymbol{A}\boldsymbol{A}^T(\boldsymbol{x}'_i - \boldsymbol{x}_i) - \lambda \boldsymbol{L}^T\boldsymbol{L}(\boldsymbol{x}_l - \boldsymbol{x}_j) = 0 \\
&\qquad \boldsymbol{x}'_i = \boldsymbol{x}_i + \frac{\lambda}{2}\boldsymbol{A}_0^{-1}\boldsymbol{L}^T\boldsymbol{L}(\boldsymbol{x}_l - \boldsymbol{x}_j) \\
&\left(\boldsymbol{L}\boldsymbol{x}_i + \frac{\lambda}{2}\boldsymbol{L}\boldsymbol{A}_0^{-1}\boldsymbol{L}^T\boldsymbol{L}(\boldsymbol{x}_l - \boldsymbol{x}_j) - \frac{\boldsymbol{L}\boldsymbol{x}_j + \boldsymbol{L}\boldsymbol{x}_l}{2}\right)^T(\boldsymbol{L}\boldsymbol{x}_l - \boldsymbol{L}\boldsymbol{x}_j) = 0 \\
&\qquad \frac{\lambda}{2} = \frac{(\frac{\boldsymbol{x}_j + \boldsymbol{x}_l}{2} - \boldsymbol{x}_i)^T\boldsymbol{L}^T\boldsymbol{L}(\boldsymbol{x}_l - \boldsymbol{x}_j)}{(\boldsymbol{x}_l - \boldsymbol{x}_j)^T\boldsymbol{L}^T\boldsymbol{L}\boldsymbol{A}_0^{-1}\boldsymbol{L}^T\boldsymbol{L}(\boldsymbol{x}_l - \boldsymbol{x}_j)} \\
&\qquad \boldsymbol{x}_{i,\min} = \boldsymbol{x}_i + \frac{(\frac{\boldsymbol{x}_j + \boldsymbol{x}_l}{2} - \boldsymbol{x}_i)^T\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j)}{(\boldsymbol{x}_l - \boldsymbol{x}_j)^T\boldsymbol{M}\boldsymbol{A}_0^{-1}\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j)}\boldsymbol{A}_0^{-1}\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j).
\end{aligned}$$

The Hessian matrix equals $2\boldsymbol{A}_0$, which is positive definite, and hence $\boldsymbol{x}_{i,\text{min}}$ is the minimum point. Replacing $\boldsymbol{A}_0 = \boldsymbol{I}$ (identity matrix) gives Eq. 3.2.

The squared adversarial margin is calculated by first simplifying $\boldsymbol{x}_{i,\text{min}}$ and then computing r^2 as follows:

$$
\begin{aligned}
&(\frac{\boldsymbol{x}_j + \boldsymbol{x}_l}{2} - \boldsymbol{x}_i)^T \boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j) \\
=&\frac{1}{2}\big((\boldsymbol{x}_j - \boldsymbol{x}_i) + (\boldsymbol{x}_l - \boldsymbol{x}_i)\big)^T \boldsymbol{M}\big((\boldsymbol{x}_l - \boldsymbol{x}_i) - (\boldsymbol{x}_j - \boldsymbol{x}_i)\big) \\
=&\frac{1}{2}\big(d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)\big)
\end{aligned}
$$

$$
\begin{aligned}
r^2 =&(\boldsymbol{x}_i - \boldsymbol{x}_{i,\text{min}})^T \boldsymbol{A}_0 (\boldsymbol{x}_i - \boldsymbol{x}_{i,\text{min}}) \\
=&\left(\frac{d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)}{2(\boldsymbol{x}_l - \boldsymbol{x}_j)^T \boldsymbol{M}\boldsymbol{A}_0^{-1}\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j)} \boldsymbol{A}_0^{-1}\boldsymbol{L}^T\boldsymbol{L}(\boldsymbol{x}_l - \boldsymbol{x}_j)\right)^T \\
&\boldsymbol{A}_0 \left(\frac{d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)}{2(\boldsymbol{x}_l - \boldsymbol{x}_j)^T \boldsymbol{M}\boldsymbol{A}_0^{-1}\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j)} \boldsymbol{A}_0^{-1}\boldsymbol{L}^T\boldsymbol{L}(\boldsymbol{x}_l - \boldsymbol{x}_j)\right) \\
=&\frac{\big(d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)\big)^2}{4\left((\boldsymbol{x}_l - \boldsymbol{x}_j)^T \boldsymbol{M}\boldsymbol{A}_0^{-1}\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j)\right)^2} (\boldsymbol{x}_l - \boldsymbol{x}_j)^T \boldsymbol{L}^T\boldsymbol{L}\boldsymbol{A}_0^{-1}\boldsymbol{A}_0\boldsymbol{A}_0^{-1}\boldsymbol{L}^T\boldsymbol{L}(\boldsymbol{x}_l - \boldsymbol{x}_j) \\
=&\frac{\big(d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)\big)^2}{4\left((\boldsymbol{x}_l - \boldsymbol{x}_j)^T \boldsymbol{M}\boldsymbol{A}_0^{-1}\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j)\right)}.
\end{aligned}
$$

Substituting $\boldsymbol{A}_0 = \boldsymbol{I}$ gives Eq. 3.3.

Now, we derive the gradient of J_P with respect to $\boldsymbol{M}$. When $d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) > d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $r \geq \tau$ (i.e. $\tilde{d}_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\text{min}}) \geq \tau$ in the hyperspherical case), or $d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) \leq d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$, the gradient equals zero. When $d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) > d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $r < \tau$, the gradient of J_P equals the gradient of $-r^2$ (i.e. $-\tilde{d}^2_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\text{min}})$ in the hyperspherical case), which can be calculated by using the quotient rule and the derivative of trace [61]:

$$
\begin{aligned}
&\frac{\partial}{\partial \boldsymbol{M}}\left(d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)\right)^2 \\
&\quad=2\left(d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)\right)(\boldsymbol{X}_{il} - \boldsymbol{X}_{ij}) \\
&\frac{\partial}{\partial \boldsymbol{M}}(\boldsymbol{x}_l - \boldsymbol{x}_j)^T \boldsymbol{M}\boldsymbol{A}_0^{-1}\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j) \\
&\quad=\frac{\partial}{\partial \boldsymbol{M}}\operatorname{tr}(\boldsymbol{X}_{jl}\boldsymbol{M}\boldsymbol{A}_0^{-1}\boldsymbol{M}) \\
&\quad=\boldsymbol{X}_{jl}\boldsymbol{M}\boldsymbol{A}_0^{-1} + \boldsymbol{A}_0^{-1}\boldsymbol{M}\boldsymbol{X}_{jl}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial}{\partial \boldsymbol{M}} & \frac{\left(d^2_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_l) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j)\right)^2}{4\left((\boldsymbol{x}_l-\boldsymbol{x}_j)^T\boldsymbol{M}\boldsymbol{A}_0^{-1}\boldsymbol{M}(\boldsymbol{x}_l-\boldsymbol{x}_j)+\epsilon\right)} \\
=& \frac{2\left(d^2_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_l) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j)\right)(\boldsymbol{X}_{il}-\boldsymbol{X}_{ij})}{4\left((\boldsymbol{x}_l-\boldsymbol{x}_j)^T\boldsymbol{M}\boldsymbol{A}_0^{-1}\boldsymbol{M}(\boldsymbol{x}_l-\boldsymbol{x}_j)+\epsilon\right)} \\
&- \frac{\left(d^2_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_l) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j)\right)^2(\boldsymbol{X}_{jl}\boldsymbol{M}\boldsymbol{A}_0^{-1}+\boldsymbol{A}_0^{-1}\boldsymbol{M}\boldsymbol{X}_{jl})}{4\left((\boldsymbol{x}_l-\boldsymbol{x}_j)^T\boldsymbol{M}\boldsymbol{A}_0^{-1}\boldsymbol{M}(\boldsymbol{x}_l-\boldsymbol{x}_j)+\epsilon\right)^2},
\end{aligned}
$$

where $\mathrm{tr}(\cdot)$ denotes the trace operator. $\boldsymbol{X}_{ij} = (\boldsymbol{x}_i - \boldsymbol{x}_j)(\boldsymbol{x}_i - \boldsymbol{x}_j)^T$ and $\boldsymbol{X}_{il}, \boldsymbol{X}_{jl}$ are defined similarly. Substituting $\boldsymbol{A}_0 = \boldsymbol{I}$ gives Eq. 3.6.

3.5.2 Derivation of support point, adversarial margin, and gradient of perturbation loss in the high-dimensional case

Support point, adversarial margin and gradient of the perturbation loss with dimensionality reduction are derived by following the same principle as in Appendix 3.5.1.

The method of Lagrangian multiplier is applied to derive a closed-form solution to the support point:

$$
\begin{aligned}
&\min_{\boldsymbol{x}'_i} \left(\boldsymbol{A}^T(\boldsymbol{x}'_i-\boldsymbol{x}_i)\right)^T\left(\boldsymbol{A}^T(\boldsymbol{x}'_i-\boldsymbol{x}_i)\right) - \lambda(\boldsymbol{x}'_i - \frac{\boldsymbol{x}_j+\boldsymbol{x}_l}{2})^T\boldsymbol{D}^T\boldsymbol{L}^T\boldsymbol{L}\boldsymbol{D}(\boldsymbol{x}_l-\boldsymbol{x}_j) \\
&\frac{\delta}{\delta \boldsymbol{x}'_i} : \boldsymbol{x}'_i = \boldsymbol{x}_i + \frac{\lambda}{2}\boldsymbol{A}_0^{-1}\boldsymbol{D}^T\boldsymbol{L}^T\boldsymbol{L}\boldsymbol{D}(\boldsymbol{x}_l-\boldsymbol{x}_j) \\
&\qquad \frac{\lambda}{2} = \frac{(\frac{\boldsymbol{x}_j+\boldsymbol{x}_l}{2}-\boldsymbol{x}_i)^T\boldsymbol{D}^T\boldsymbol{L}^T\boldsymbol{L}\boldsymbol{D}(\boldsymbol{x}_l-\boldsymbol{x}_j)}{(\boldsymbol{x}_l-\boldsymbol{x}_j)^T\boldsymbol{D}^T\boldsymbol{L}^T\boldsymbol{L}\boldsymbol{D}\boldsymbol{A}_0^{-1}\boldsymbol{D}^T\boldsymbol{L}^T\boldsymbol{L}\boldsymbol{D}(\boldsymbol{x}_l-\boldsymbol{x}_j)} \\
&\quad \boldsymbol{x}^{\text{PCA}}_{i,\text{min}} = \boldsymbol{x}_i + \frac{(\frac{\tilde{\boldsymbol{x}}_j+\tilde{\boldsymbol{x}}_l}{2}-\tilde{\boldsymbol{x}}_i)^T\boldsymbol{M}(\tilde{\boldsymbol{x}}_l-\tilde{\boldsymbol{x}}_j)}{(\tilde{\boldsymbol{x}}_l-\tilde{\boldsymbol{x}}_j)^T\boldsymbol{M}\boldsymbol{D}\boldsymbol{A}_0^{-1}\boldsymbol{D}^T\boldsymbol{M}(\tilde{\boldsymbol{x}}_l-\tilde{\boldsymbol{x}}_j)}\boldsymbol{A}_0^{-1}\boldsymbol{D}^T\boldsymbol{M}(\tilde{\boldsymbol{x}}_l-\tilde{\boldsymbol{x}}_j),
\end{aligned}
$$

where $\tilde{\boldsymbol{x}}$ denotes $\boldsymbol{D}\boldsymbol{x}$.

The squared adversarial margin is calculated from the definition of the hyper-ellipsoid:

$$
\begin{aligned}
r^2 =& (\boldsymbol{x}_i - \boldsymbol{x}^{\text{PCA}}_{\text{min}})^T\boldsymbol{A}_0(\boldsymbol{x}_i - \boldsymbol{x}^{\text{PCA}}_{\text{min}}) \\
=& \frac{\left(d^2_{\boldsymbol{M}}(\tilde{\boldsymbol{x}}_i,\tilde{\boldsymbol{x}}_l) - d^2_{\boldsymbol{M}}(\tilde{\boldsymbol{x}}_i,\tilde{\boldsymbol{x}}_j)\right)^2}{4\left((\tilde{\boldsymbol{x}}_l-\tilde{\boldsymbol{x}}_j)^T\boldsymbol{L}^T\boldsymbol{L}\boldsymbol{D}\boldsymbol{A}_0^{-1}\boldsymbol{D}^T\boldsymbol{L}^T\boldsymbol{L}(\tilde{\boldsymbol{x}}_l-\tilde{\boldsymbol{x}}_j)\right)^2} \\
&\cdot (\tilde{\boldsymbol{x}}_l-\tilde{\boldsymbol{x}}_j)^T\boldsymbol{L}^T\boldsymbol{L}\boldsymbol{D}\boldsymbol{A}_0^{-1}\boldsymbol{A}_0\boldsymbol{A}_0^{-1}\boldsymbol{D}^T\boldsymbol{L}^T\boldsymbol{L}(\tilde{\boldsymbol{x}}_l-\tilde{\boldsymbol{x}}_j)
\end{aligned}
$$

$$=\frac{\big(d^2_{\boldsymbol{M}}(\tilde{\boldsymbol{x}}_i,\tilde{\boldsymbol{x}}_l)-d^2_{\boldsymbol{M}}(\tilde{\boldsymbol{x}}_i,\tilde{\boldsymbol{x}}_j)\big)^2}{4(\tilde{\boldsymbol{x}}_l-\tilde{\boldsymbol{x}}_j)^T\boldsymbol{M}\boldsymbol{D}\boldsymbol{A}_0^{-1}\boldsymbol{D}^T\boldsymbol{M}(\tilde{\boldsymbol{x}}_l-\tilde{\boldsymbol{x}}_j)}$$

The perturbation loss is defined similarly to Eq. 3.4 as follows:

$$J_{\text{P}}^{\text{PCA}}=\frac{1}{|\mathcal{R}|}\sum_{\mathcal{R}}\Big\{[\tau^2-\tilde{d}^2_{\boldsymbol{E}}(\boldsymbol{x}_i,\boldsymbol{x}^{\text{PCA}}_{i,\text{min}})]_+\mathbb{1}_{\{d^2_{\boldsymbol{M}}(\tilde{\boldsymbol{x}}_i,\tilde{\boldsymbol{x}}_l)>d^2_{\boldsymbol{M}}(\tilde{\boldsymbol{x}}_i,\tilde{\boldsymbol{x}}_j)\}}+\tau^2\mathbb{1}_{\{d^2_{\boldsymbol{M}}(\tilde{\boldsymbol{x}}_i,\tilde{\boldsymbol{x}}_l)\le d^2_{\boldsymbol{M}}(\tilde{\boldsymbol{x}}_i,\tilde{\boldsymbol{x}}_j)\}}\Big\},$$

where $\tilde{d}^2_{\boldsymbol{E}}(\boldsymbol{x}_i,\boldsymbol{x}^{\text{PCA}}_{i,\text{min}})=\frac{\big(d^2_{\boldsymbol{M}}(\tilde{\boldsymbol{x}}_i,\tilde{\boldsymbol{x}}_l)-d^2_{\boldsymbol{M}}(\tilde{\boldsymbol{x}}_i,\tilde{\boldsymbol{x}}_j)\big)^2}{4((\tilde{\boldsymbol{x}}_l-\tilde{\boldsymbol{x}}_j)^T\boldsymbol{M}\boldsymbol{D}\boldsymbol{D}^T\boldsymbol{M}(\tilde{\boldsymbol{x}}_l-\tilde{\boldsymbol{x}}_j)+\epsilon)}$.

The gradient of $J_{\text{P}}^{\text{PCA}}$ is given as:

$$\begin{aligned}\frac{\partial J_{\text{P}}^{\text{PCA}}}{\partial\boldsymbol{M}}=&\frac{1}{|\mathcal{R}|}\sum_{\mathcal{R}}\alpha_{ijl}\Bigg\{\frac{\big(d^2_{\boldsymbol{M}}(\tilde{\boldsymbol{x}}_i,\tilde{\boldsymbol{x}}_l)-d^2_{\boldsymbol{M}}(\tilde{\boldsymbol{x}}_i,\tilde{\boldsymbol{x}}_j)\big)(\tilde{\boldsymbol{X}}_{ij}-\tilde{\boldsymbol{X}}_{il})}{2\big((\tilde{\boldsymbol{x}}_l-\tilde{\boldsymbol{x}}_j)^T\boldsymbol{M}\boldsymbol{D}\boldsymbol{A}_0^{-1}\boldsymbol{D}^T\boldsymbol{M}(\tilde{\boldsymbol{x}}_l-\tilde{\boldsymbol{x}}_j)+\epsilon\big)}\\&+\frac{\big(d^2_{\boldsymbol{M}}(\tilde{\boldsymbol{x}}_i,\tilde{\boldsymbol{x}}_l)-d^2_{\boldsymbol{M}}(\tilde{\boldsymbol{x}}_i,\tilde{\boldsymbol{x}}_j)\big)^2}{4\big((\tilde{\boldsymbol{x}}_l-\tilde{\boldsymbol{x}}_j)^T\boldsymbol{M}\boldsymbol{D}\boldsymbol{A}_0^{-1}\boldsymbol{D}^T\boldsymbol{M}(\tilde{\boldsymbol{x}}_l-\tilde{\boldsymbol{x}}_j)+\epsilon\big)^2}\\&\cdot(\tilde{\boldsymbol{X}}_{jl}\boldsymbol{M}\boldsymbol{D}\boldsymbol{A}_0^{-1}\boldsymbol{D}^T+\boldsymbol{D}\boldsymbol{A}_0^{-1}\boldsymbol{D}^T\boldsymbol{M}\tilde{\boldsymbol{X}}_{jl})\Bigg\}.\end{aligned}$$

3.5.3 Sparse compositional metric learning towards certified robustness

We start by briefly revisiting the sparse compositional metric learning (SCML) method [39]. The core idea is to represent the Mahalanobis distance as a non-negative combination of K basis elements; that is,

$$\boldsymbol{M}=\sum_{k=1}^{K}w_k\boldsymbol{b}_k\boldsymbol{b}_k^T,\quad \boldsymbol{w}\ge 0,$$

where the basis set $\{\boldsymbol{b}_k\}_{k=1}^K$ is generated by using the Fisher discriminative analysis at several local regions. To learn a discriminative metric with good generalisation ability, the learning objective comprises a margin-based hinge loss function and an L_1-norm regularisation term as follows:

$$\min_{\boldsymbol{w}}J_{\text{SCML}}=\frac{1}{|\mathcal{R}|}\sum_{\mathcal{R}}\big[1+d^2_{\boldsymbol{w}}(\boldsymbol{x}_i,\boldsymbol{x}_j)-d^2_{\boldsymbol{w}}(\boldsymbol{x}_i,\boldsymbol{x}_l)\big]_++\eta\|\boldsymbol{w}\|_1,$$

where $\eta \geq 0$ controls the degree of sparsity.

Similar to LMNN-CR, we propose SCML towards certified robustness (SCML-CR) by adding the perturbation loss J_{P} (Eq. 3.4) to the original objective function J_{SCML}:

$$\min_{\boldsymbol{M} \in \mathbb{S}_{+}^{p}} J = J_{\text{SCML}} + \lambda J_{\text{P}}. \tag{3.8}$$

The adversarial margin of Eq. 3.3 is now a function of $\boldsymbol{w}$:

$$\begin{aligned}
d_{\boldsymbol{E}}^2(\boldsymbol{x}_i, \boldsymbol{x}_{i,\min}) &= \frac{\left(d_{\boldsymbol{w}}^2(\boldsymbol{x}_i, \boldsymbol{x}_l) - d_{\boldsymbol{w}}^2(\boldsymbol{x}_i, \boldsymbol{x}_j)\right)^2}{4 d_{\boldsymbol{w}^2}^2(\boldsymbol{x}_j, \boldsymbol{x}_l)} \\
d_{\boldsymbol{w}}^2(\boldsymbol{x}_i, \boldsymbol{x}_j) &= (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \Big(\sum_{k=1}^{K} w_k \boldsymbol{b}_k \boldsymbol{b}_k^T \Big) (\boldsymbol{x}_i - \boldsymbol{x}_j) \\
d_{\boldsymbol{w}^2}^2(\boldsymbol{x}_j, \boldsymbol{x}_l) &= (\boldsymbol{x}_j - \boldsymbol{x}_l)^T \Big(\sum_{k_1=1}^{K} \sum_{k_2=1}^{K} w_{k_1} w_{k_2} \boldsymbol{b}_{k_1} \boldsymbol{b}_{k_1}^T \boldsymbol{b}_{k_2} \boldsymbol{b}_{k_2}^T \Big) (\boldsymbol{x}_j - \boldsymbol{x}_l).
\end{aligned}$$

The optimisation problem (Eq. 3.8) is solved via the accelerated proximal gradient descent algorithm. The gradient of J_{P} with respect to $\boldsymbol{w}$ is as follows:

$$\begin{aligned}
\frac{\partial J_{\text{P}}}{\partial w_k} =& \frac{1}{|\mathcal{R}|} \sum_{\mathcal{R}} \alpha_{ijl} \Bigg\{ \frac{d_{\boldsymbol{M}}^2(\boldsymbol{x}_i, \boldsymbol{x}_l) - d_{\boldsymbol{M}}^2(\boldsymbol{x}_i, \boldsymbol{x}_j)}{2\left(d_{\boldsymbol{M}^2}^2(\boldsymbol{x}_j, \boldsymbol{x}_l) + \epsilon\right)} \operatorname{tr}\left(\boldsymbol{b}_k \boldsymbol{b}_k^T (\boldsymbol{X}_{ij} - \boldsymbol{X}_{il}) \right) \\
&+ \frac{[d_{\boldsymbol{M}}^2(\boldsymbol{x}_i, \boldsymbol{x}_l) - d_{\boldsymbol{M}}^2(\boldsymbol{x}_i, \boldsymbol{x}_j)]^2}{4\left(d_{\boldsymbol{M}^2}^2(\boldsymbol{x}_j, \boldsymbol{x}_l) + \epsilon\right)^2} \operatorname{tr}\left((\boldsymbol{b}_k \boldsymbol{b}_k^T \boldsymbol{M} + \boldsymbol{M} \boldsymbol{b}_k \boldsymbol{b}_k^T) \boldsymbol{X}_{jl} \right) \Bigg\},
\end{aligned}$$

where ϵ is a small constant added to the denominator of $d_{\boldsymbol{E}}^2(\boldsymbol{x}_i, \boldsymbol{x}_{i,\min})$.

3.5.4 Preliminaries and theorem on generalisation bound

The generalisation bound is proved based on pseudo-robustness of an algorithm. The definition of pseudo-robustness and existing theorem are provided, followed by the proof of Theorem 3.

3.5.4.1 Preliminaries

The following notations will be used in this section. Let $\boldsymbol{z} = (\boldsymbol{x}, y) \in \mathcal{Z}$ denote an instance-label pair, $\mathcal{D}$ denote the underlying distribution of $\boldsymbol{z}$, and ℓ denote a loss function upper bounded by c. $t_{\boldsymbol{z}^n} = \{(\boldsymbol{z}_i, \boldsymbol{z}_j, \boldsymbol{z}_l) : y_i = y_j \neq y_l\}$. The generalisation gap is the difference between the expected risk R and the empirical risk R_n.

Definition 7. [41] An algorithm $\mathcal{A}$ is $(K, \epsilon(\cdot), \hat{n}(\cdot))$ *pseudo-robust* for $K \in \mathbb{N}$, $\epsilon(\cdot) : (\mathcal{Z} \times \mathcal{Z} \times \mathcal{Z})^n \to \mathbb{R}$ and $\hat{n}(\cdot) : (\mathcal{Z} \times \mathcal{Z} \times \mathcal{Z})^n \to \{1, \dots, n^3\}$ if $\mathcal{Z}$ can be partitioned into K disjoint sets, denoted by $\{C_k\}_{k=1}^K$, such that for all samples $\boldsymbol{z}^n \in \mathcal{Z}^n$, there exists a subset of training triplets $\hat{t}_{\boldsymbol{z}^n} \subseteq t_{\boldsymbol{z}^n}$ with $|\hat{t}_{\boldsymbol{z}^n}| = \hat{n}(t_{\boldsymbol{z}^n})$ that the following holds:
$\forall(\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) \in \hat{t}_{\boldsymbol{z}^n}, \forall \boldsymbol{z}_1', \boldsymbol{z}_2', \boldsymbol{z}_3' \in \mathcal{Z}, \forall i, j, l = 1, \dots, K$, if $\boldsymbol{z}_1, \boldsymbol{z}_1' \in C_i$, $\boldsymbol{z}_2, \boldsymbol{z}_2' \in C_j$ and $\boldsymbol{z}_3, \boldsymbol{z}_3' \in C_l$, then

$$|\ell(\mathcal{A}_{t_{\boldsymbol{z}^n}}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3) - \ell(\mathcal{A}_{t_{\boldsymbol{z}^n}}, \boldsymbol{z}_1', \boldsymbol{z}_2', \boldsymbol{z}_3')| \leq \epsilon(t_{\boldsymbol{z}^n}).$$

Theorem 4. [41] If $\mathcal{A}$ is $(K, \epsilon(\cdot), \hat{n}(\cdot))$ pseudo-robust and the training triplets $t_{\boldsymbol{z}^n}$ come from from a sample set $\boldsymbol{z}^n$ that is generated by n i.i.d. draws from $\mathcal{D}$, then with probability at least $1 - \delta$ we have:

$$|R(\mathcal{A}_{t_{\boldsymbol{z}^n}}) - R_n(\mathcal{A}_{t_{\boldsymbol{z}^n}})| \leq \frac{\hat{n}(t_{\boldsymbol{z}^n})}{n^3}\epsilon(t_{\boldsymbol{z}^n}) + c\Big(\frac{n^3 - \hat{n}(t_{\boldsymbol{z}^n})}{n^3} + 3\sqrt{\frac{2K\ln 2 + 2\ln 1/\delta}{n}}\Big). \tag{3.9}$$

Definition 8. [62] A *δ-cover* of a set Θ with respect to a metric ρ is a set $\{\theta^1, \dots, \theta^N\} \subset \Theta$ such that for each $\theta \in \Theta$, there exists some $i \in \{1, \dots, N\}$ such that $\rho(\theta, \theta^i) \leq \delta$. The *$\delta$-covering* number $N(\delta, \Theta, \rho)$ is the cardinality of the smallest δ-cover.

3.5.4.2 Theorem and proof

Theorem 5. Let $\boldsymbol{M}^*$ be the optimal solution to Eq. 3.5. Then with probability at least $1 - \delta$ we have:

$$|R(\boldsymbol{M}^*) - R_n(\boldsymbol{M}^*)| \leq \frac{\hat{n}(t_{\boldsymbol{z}^n})}{n^3} + c\Big(\frac{n^3 - \hat{n}(t_{\boldsymbol{z}^n})}{n^3} + 3\sqrt{\frac{2K\ln 2 + 2\ln 1/\delta}{n}}\Big),$$

where $\hat{n}(t_{\boldsymbol{z}^n})$ denotes the number of triplets whose adversarial margins are larger than τ, c is a constant denoting the upper bound of the loss function (i.e. Eq. 3.4), and $K = |\mathcal{Y}|(1+\frac{2}{\tau})^p$.

Proof. After embedding the perturbation loss, learning algorithms that minimise the classical triplet loss, i.e. $[1 + d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l)]_+ \cdot \mathbb{1}_{\{y_i = y_j \neq y_l\}}$, are $(|\mathcal{Y}|(1+\frac{2}{\tau})^p, 1, \hat{n}(\cdot;\tau))$ pseudo-robust. $\epsilon = 1$ since, by definition of certified neighbourhood, any $\boldsymbol{x}$ that falls into the Euclidean ball with centre $\boldsymbol{x}_i$ and a radius of the desired margin τ will satisfy $d^2_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}_l) > d^2_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}_j)$, and therefore any change in loss is bounded by 1. The value of K can be determined via the covering number [41]. The instance space $\mathcal{X}$ can be partitioned by using the covering number $N(\tau, \mathcal{X}, \|\cdot\|_2)$. By normalising all instances to have unit L_2-norm, we obtain a finite covering number as $N \leq (1+\frac{2}{\tau})^p$ [62]. The label space $\mathcal{Y}$ can be partitioned into $|\mathcal{Y}|$ sets. Therefore, the number of disjoint sets, i.e. K, is always smaller than $|\mathcal{Y}|(1+\frac{2}{\tau})^p$. □

3.5.5 Experimental setup and additional results

3.5.5.1 Experimental setting

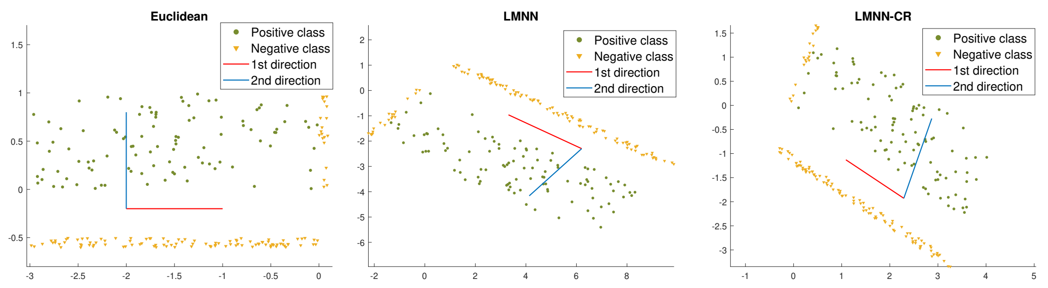
Hyperparameter tuning of compared methods LMNN, SCML, DRIFT and AML are implemented by using the official codes provided by the authors; all parameters are set as default apart from the trade-off parameter. Trade-off parameters are tuned via five-fold cross-validation on the training data. For LMNN, the trade-off parameter μ is chosen from $\{0.1, 0.2, \dots, 0.9\}$. For SCML, the weight of the regularisation term η is chosen from $\{10^{-5}, 10^{-4}, \dots, 10^3\}$, and the number of bases is set as 200, 400 and 1000 for UCI datasets whose sample size is smaller than 500, larger than 500, and high-dimensional datasets, respectively. For LDD, the regulariser weight is chosen from $\{10^{-6}, 10^{-5}, \dots, 10^2\}$. For CAP, the regulariser weight is chosen from $\{10^{-3}, 10^{-2}, \dots, 10\}$, and the rank of $\boldsymbol{M}$ is chosen from 10 values equally spaced between 1 and p. For DRIFT and AML, we search the grid suggested by the authors.

Experimental setting of LMNN-CR $\boldsymbol{M}$ is initialised as the identity matrix. The learning rate γ is initialised to 1. Following [10]'s work, γ is increased by 1% if the loss function decreases and decreased by 50% otherwise. The training stops if the relative change in the objective function is smaller than the threshold of 10^{-7} or reaches the maximum number of iterations of 1000.

Experimental setting of SCML-CR SCML-CR is tuned in the same manner as LMNN-CR via random search; the range of η and the number of bases are same as SCML, and the ranges of τ and λ are same as LMNN-CR. The method is optimised via the accelerated proximal gradient descent algorithm with a backtracking stepsize rule [63]. The initial learning rate is set as 1 and the shrinkage factor is set as 0.8. $\boldsymbol{w}$ is initialised as the unit vector.

3.5.5.2 Comparisons between LMNN and LMNN-CR

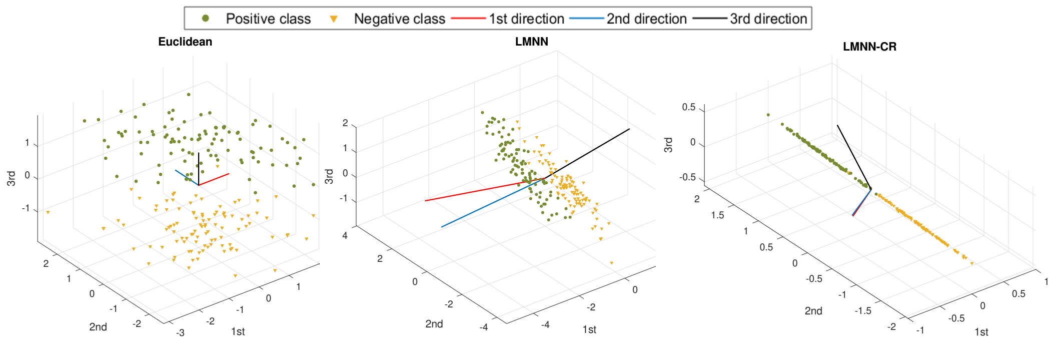
In this section, two toy examples are presented to illustrate the difference in the learning mechanisms of LMNN and LMNN-CR.



(a) $\boldsymbol{M} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \bar{d}_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\min}) = 0.17$ (b) $\boldsymbol{M} = \begin{bmatrix} 10.2 & 3.5 \\ 3.5 & 7.8 \end{bmatrix}, \bar{d}_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\min}) = 0.15$ (c) $\boldsymbol{M} = \begin{bmatrix} 2.1 & 0.6 \\ 0.6 & 3.1 \end{bmatrix}, \bar{d}_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\min}) = 0.16$

Figure 3.6: Comparison of learning mechanisms of LMNN and LMNN-CR when features exhibit different separability.

In the first example, we simulate a two-dimensional binary classification dataset, as shown in Figure 3.6a. The positive class includes 100 instances drawn uniformly from $[-3, 0]$ in the horizontal (abbr. 1st) direction and $[0, 1]$ in the vertical (abbr. 2nd) direction. The negative class consists of two clusters, where the first cluster includes 100 instances drawn from $U(-3, 0)$ and $U(-0.6, -0.5)$ in the 1st and 2nd directions respectively, and the second cluster includes 20 instances drawn from $U(0, 0.1)$ and $U(0, 1)$ in the two directions respectively. By design, instances of positive and negative classes can be separated in both directions, while the separability in the 1st direction is much smaller than the 2nd direction. Figures 3.6b and 3.6c show the instances in the projected feature space with metrics learned from LMNN and LMNN-CR, respectively; the projection direction is indicated by the unit vector of red and blue lines; and the metric and the average of adversarial margins ($\bar{d}_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\min})$) are given in the caption. The objective of LMNN is to satisfy the distance margin. Thus, it expands the distance in both directions. Moreover, since the 1st direction has a small separability in the original instance space, this direction is assigned with a larger weight. In contrast, LMNN-CR controls the scale of $\boldsymbol{M}$. Moreover, a notable difference is that the 2nd direction is assigned with a larger weight than the 1st direction, which is again caused by the small separability in the 1st direction. As any perturbation in the 1st direction is highly likely to result in a misclassification, the proposed method diverts more attention to robust features, i.e. the 2nd direction.



(a) $\boldsymbol{M} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $\bar{d}_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\min})$=0.83

(b) $\boldsymbol{M} = \begin{bmatrix} 22.2 & 23.3 & -16.6 \\ 23.3 & 27.6 & -18.5 \\ -16.6 & -18.5 & 13.7 \end{bmatrix}$, $\bar{d}_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\min})$=0.05

(c) $\boldsymbol{M} = \begin{bmatrix} 0.449 & 0.451 & -0.077 \\ 0.451 & 0.454 & -0.063 \\ -0.077 & -0.063 & 0.853 \end{bmatrix}$, $\bar{d}_{\boldsymbol{E}}(\boldsymbol{x}_i, \boldsymbol{x}_{i,\min})$=0.70

Figure 3.7: Comparison of learning mechanisms of LMNN and LMNN-CR when confronting the problem of multicollinearity.

In the second example, we simulate a three-dimensional binary classification dataset, as shown in Figure 3.7a. The positive and negative classes each includes 100 instances. The first two dimensions are drawn from multivariate Gaussian distributions with $\boldsymbol{\mu}_p = [0.45, 0.45]$, $\boldsymbol{\mu}_n = [-0.45, -0.45]$, $\boldsymbol{\Sigma}_p = \boldsymbol{\Sigma}_n = \left[\begin{smallmatrix} 1 & -0.9 \\ -0.9 & 1 \end{smallmatrix}\right]$. The third dimension equals the sum of the first two dimensions, plus white Gaussian noise with standard deviation of 0.01. By design, the dataset exhibits the problem of strong multicollinearity. This issue has little influence on LMNN as the data is nearly separable in all directions. However, it will affect the certified neighbourhood. Specifically, if the metric assigns equal weights to all dimensions, then the perturbation should be small in all directions so as to guarantee that the perturbed instance stays on the correct side of the decision boundary. In contrast, if the metric assigns weights only to the third dimension, then the perturbation in the first two dimensions will not cause any change in the learned feature space and hence a larger magnitude of perturbation in the third dimension could be allowed. This expectation is supported by the empirical result in Figure 3.7c, where the distance in the third dimension is more important than the first two dimensions.

In summary, our method learns a discriminative metric, and meanwhile, imposes a data-dependent regularisation on the metric. It also achieves larger adversarial margins than LMNN.

3.5.5.3 Additional experimental results

Table 3.7 is a supplement to Section 3.3.2.3, which reports the robustness of DML methods under spherical Gaussian noise with the SNR of 5 dB. Table 3.8 is a supplement to Table 3.5, which reports the performances of AML, LDD and DRIFT on high-dimensional datasets.

Table 3.7: Classification accuracy of DML methods on noise-contaminated datasets. Spherical Gaussian noise with an SNR of 5 dB is added to test data.

		LMNN-based					SCML-based	
Dataset	AML	LMNN	LDD	CAP	DRIFT	LMNN-CR	SCML	SCML-CR
Australian	82.67±1.59	82.46±1.58	83.02±1.58	82.36±1.56	82.58±1.45	**83.50±1.56**	82.93±1.65	**83.42±1.68**
Breast cancer	96.74±1.01	96.25±1.09	96.69±1.09	96.35±1.02	96.66±1.00	**96.71±1.08**	96.40±1.05	**96.65±1.06**
Fourclass	68.91±1.16	67.62±1.23	68.77±1.14	67.63±1.12	**69.03±1.13**	69.01±1.17	68.07±1.16	**68.86±1.06**
Haberman	69.81±1.85	69.84±1.79	**69.92±1.87**	69.23±2.00	69.09±2.49	69.89±1.90	69.65±1.63	**69.88±1.83**
Iris	78.92±3.25	78.61±2.97	78.87±3.16	77.79±3.27	78.43±3.09	**79.04±3.09**	78.16±3.58	**79.01±3.12**
Segment	87.92±0.69	81.02±3.55	86.15±1.26	85.34±2.47	**86.63±1.09**	84.72±2.62	60.18±9.73	**61.33±9.05**
Sonar	83.46±3.39	83.56±4.27	**86.18±2.95**	85.41±2.82	84.65±3.30	85.00±3.15	77.01±4.23	**79.49±3.80**
Voting	94.10±1.07	94.00±1.00	94.25±1.14	94.37±1.17	93.95±1.12	**94.64±1.21**	93.99±1.15	**94.64±1.09**
WDBC	96.49±1.12	91.71±1.90	**96.30±0.94**	96.16±1.08	96.04±0.86	96.11±0.88	95.74±1.30	**96.21±1.16**
Wine	95.12±1.12	93.33±1.63	94.03±1.39	93.97±1.47	**94.66±1.15**	94.51±1.20	94.01±1.56	**94.61±1.32**
#Outperform	-	10	6	7	7	-	10	-

Table 3.8: Generalisation and robustness of DML methods on high-dimensional datasets (additional results).

Isolet						
Method	Clean	SG,SNR=20	SG,SNR=5	G,SNR=20	G,SNR=5	Margin
AML	86.75±3.16	86.59±3.49	85.97±3.69	86.69±3.59	86.24±3.82	0.1261
LDD	90.91±3.90	90.81±4.12	87.97±3.83	90.75±4.12	89.13±4.05	0.1333

MNIST						
Method	Clean	SG,SNR=20	SG,SNR=5	G,SNR=20	G,SNR=5	Margin
AML	89.25	88.70	88.85	89.30	89.20	0.2142
LDD	90.85	90.85	87.90	90.95	90.30	0.2232
DRIFT	90.85	90.75	87.45	90.65	89.45	0.2054

DRIFT is unable to learn a metric on Isolet and hence is not reported.

Chapter 4

Metric Learning for Categorical and Ambiguous Features

4.1 Introduction

The k-nearest neighbours (kNN) algorithm is a classical and widely used classification method by virtue of the nonparametric nature, interpretability, and flexibility in defining the distance between instances [64]. As a discriminative distance function can boost kNN's performance, the idea of learning a task-specific metric from the data was pioneered in [1], which formulates the task of learning a generalised Mahalanobis distance as a convex optimisation problem. Thereafter, many global [10], local [65], kernelised [66] and deep [67] metric learning methods have been proposed to further improve the discriminability. While these methods are effective, they have rarely been applied to data with ordinal and nominal features.

Ordinal and nominal variables (i.e. features) are subsumed under the data type of categorical variables that have measurement scales consisting of a set of categories [68]. Categorical variables with ordered scales are called ordinal variables and the ones with unordered scales are called nominal variables. For example, when collecting a film survey, the audience review (poor, fair, good, excellent) is an ordinal variable; the genre of favourite films (action, comedy, drama, horror) is a nominal variable. Both types of variables occur frequently in social and health sciences and also arise in education and marketing.

Classifying ordinal and nominal variables faces at least the following three challenges. First, a simple way of representing these variables is to encode them as integers and then treat them as real-valued numerical variables. However, for an ordinal variable, the difference between two integers does not necessarily reflect the real distance between the two ordinal levels, and for a nominal variable, the difference between two integers is meaningless for two nominal levels. Another way of representing ordinal and nominal variables is to encode each categorical variable into a set of binary variables, such as through dummy coding. This conversion avoids the above problems, and allows for the modelling of interactions between different levels of the variable. However, it inevitably increases the feature dimension, and the effect is dramatic when each variable has a large number of levels. The second challenge is the ambiguity in ordinal variables. For example, in the example of audience review, the boundaries between levels such as 'good' and 'excellent' are not sharply defined, thereby causing ambiguity. This issue is less common in nominal variables, but it still appears when some categories have overlapping characteristics. Third, for economic and ethical reasons, the categorical data collected in social and health sciences often have a small sample size. This places a restriction on model complexity since a complex model may overfit and generalise poorly to unseen data.

This chapter focuses on adapting metric learning methods for ordinal and nominal features that could work on both types of encoded data, i.e. as integer variables and as dummy variables, and address the feature ambiguity and small-sized problems. Firstly, to mitigate the impact of feature ambiguity, we propose to consider the worst-case perturbation of each instance within a deliberately designed constraint set and learn the Mahalanobis distance through adversarial training. The constraint set takes into account the discrete nature of nominal variables and the ordering nature of ordinal variables. Secondly, we provide a geometric interpretation of the proposed formulation, which suggests that our method dynamically divides the instance space into three regions, namely support region, adversarially vulnerable region, and adversarially robust region. Compared with classical metric learning methods which only uses information on the support region, our method additionally uses information on instances from the adversarially vulnerable region, thereby coping better with the small sample size problem. Thirdly, we prove the generalisation bound for a general form of adversarial training. It suggests that the sample complexity rate of adversarial training remains at the same order as that of standard training in classical methods only if the Mahalanobis distance is regularised with the elementwise 1-norm. Finally, the method is tested on datasets with all ordinal variables and with a mixture of ordinal and nominal variables. It surpasses state-of-the-art methods in cases of high feature ambiguity and small sample size.

The rest of this chapter is organised as follows. Section 4.2 reviews distance metrics used for categorical data and metric learning methods that are robust to feature uncertainty. Section 4.3 proposes the new metric learning algorithm, and discusses the method from both geometric and theoretical perspectives. Section 4.4 verifies the method on simulated and real-world categorical data. Section 4.5 draws a conclusion and suggests future work. Proof, detailed experimental settings, and some additional results are deferred to Appendix 4.6.

4.2 Related work

This section briefly reviews distance metrics for categorical data and metric learning methods that consider feature uncertainty.

Distance metrics for categorical data Various distance or similarity measures are proposed for categorical data, mostly for nominal data, in an unsupervised setting. The most common measure is *overlap*, which defines the similarity between two instances $\boldsymbol{x}_1, \boldsymbol{x}_2$ on the ith feature to be 1 if their values are equal and 0 otherwise. Summing up the similarities over all features defines the distance between $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. Based on overlap, many probabilistic or frequency-based measures have been proposed to assign different weights on matches or mismatches, as well as taking into account the occurrence of other feature values [69]. Another class of measures are based on entropy, where the distance contribution of each categorical level depends on the amount of information it holds. Entropy-based measures have been extended in [70] to quantify the order relation of ordinal variables.

In a supervised setting, non-learning approaches use the label information to determine the discriminative power of each feature and adjust the feature weights in distance calculation accordingly [71]. Learning approaches learn the distance between each pair of categorical levels or a mapping function from each level to a real value by minimising the classification error [72; 73]. More recently, large margin-based metric learning methods have been adapted for ordinal and nominal variables [74; 75]. Building on the assumption that an ordinal variable represents a continuous latent variable that falls into an interval of values, [74] jointly learns the Mahalanobis distance, thresholds of intervals, and parameters of the latent variable distribution. As the number of thresholds is determined by the number of variables and levels within them, the method may involve a large number of parameters and suffer from overfitting. [75] represents the categorical data by computing the interaction between levels, between variables, and between variables and classes, followed by learning the Mahalanobis distance in a kernel space. However, it ignores the natural ordering of ordinal variables.

Metric learning with uncertainty In most metric learning methods, a Mahalanobis distance is optimised such that similar instances become closer with respect to the new metric and dissimilar instances become farther away. As the optimisation process is guided by the side information, its effectiveness degrades in the presence of label noise, influential points, and feature uncertainty. Compared with influential points which account for a small proportion of instances but severely influence the model, feature uncertainty, possibly ensued from ambiguity in the definition of set boundaries, measurement and quantisation errors, and data processing of repeated measurements, normally appears as small perturbations but potentially pollutes a large number of instances [76]. Many robust metric learning methods have been proposed to tackle the above problems [77; 78; 79], and here, we only discuss those on feature uncertainty.

One way to handle feature uncertainty is to build an explicit model of perturbation [49; 50]. [49] assumes a perturbation distribution of each instance, replaces the Mahalanobis distance by its expected value, and iteratively learns the distribution and distance metric by minimising the number of violations of triplet constraints. The method essentially adjusts the constraint on distance margin for each triplet according to its reliableness. Another approach is to learn a distance metric that is less sensitive to feature uncertainty via adversarial training [51; 52]. The method involves two stages. The confusion stage generates adversarial pairs that incur large losses, and the discrimination stage optimises the distance metric based on these augmented pairs. Originating from robust optimisation [17; 80], adversarial training has received considerable attention in recent years as an effective approach to achieving robustness to adversarial examples [5]. In addition, adversarial training is shown to improve the classification accuracy when there is only a limited number of instances available to train the model [81]. While our method shares a similar principle, it differs from existing adversarial metric learning methods in two respects. Firstly, we take the subsequent classification mechanism into consideration when searching for the worst-case perturbation. Derived from triplet constraints, the perturbation is capable of altering the decision of NN classifier. Secondly and more importantly, the loss function in our proposal is designed specifically for ordinal and nominal features with an explicit consideration of their discrete and ordering nature.

4.3 Methodology

In this section, we propose to model feature ambiguity as a perturbation to the instance and learn the Mahalanobis distance through adversarial training. After introducing notations, we will present the method and its optimisation algorithm, followed by a geometric interpretation and a generalisation analysis.

4.3.1 Preliminaries

Let $\boldsymbol{z}^n = \{\boldsymbol{z}_i = (\boldsymbol{x}_i, y_i)\}_{i=1}^n$ denote the training set, where $\boldsymbol{x}_i \in \mathcal{X}$ is the ith training instance associated with label $y_i \in \mathcal{Y} = \{1, \ldots, C\}$; $\boldsymbol{z}_i \in \mathcal{Z}$ is independently and identically distributed (i.i.d.) according to an unknown distribution $\mathcal{D}$. Suppose each instance includes p features, p^{ord} of which are ordinal variables and $p^{\text{nom}} = p - p^{\text{ord}}$ are nominal variables. Ordinal variables can be encoded as consecutive integers or as a set of binary values. In the integer case, a variable with p_r levels takes values from $\{1, 2, \ldots, p_r\}$, and the mapping should follow the order relation. In other words, for ordinal levels $O_1 \prec O_2 \prec \cdots \prec O_{p_r}$ with an order relation $\prec$, there is a mapping function $\mathcal{O}$ such that $\mathcal{O}(O_q) = q, q = 1, \ldots, p_r$. In the binary-valued case, ordinal variables are encoded via the *OrderedPartitions* method [82; 74]. For example, an ordinal variable with 3 levels will be encoded as [1,0,0], [1,1,0] and [1,1,1]. Nominal variables are encoded via the *1-of-K* encoding scheme. For example, a nominal variable with 3 levels will be encoded as [1,0,0], [0,1,0] and [0,0,1]. Let P denote the feature dimension after encoding, which equals $p^{\text{ord}} + \sum_{r=1}^{p^{\text{nom}}} p_r$ if ordinal variables are encoded as integers and equals $\sum_{r=1}^{p} p_r$ if they are encoded as a set of binary values.

In this work, we focus on learning the Mahalanobis distance from triplet-based side information. For any two instances $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathbb{R}^P$, the generalised (squared) Mahalanobis distance is defined as

$$d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{M} (\boldsymbol{x}_i - \boldsymbol{x}_j),$$

where $\boldsymbol{M} \in \mathbb{S}^P_+$ is a $P \times P$ real-valued positive semi-definite (PSD) matrix. A classical triplet-based metric learning method is the large margin nearest neighbors (LMNN) algorithm [10]. It pulls k nearest same-class instances closer and pushes away different-class instances by a fixed margin through optimising the following objective function:

$$\min_{\boldsymbol{M} \in \mathbb{S}^P_+} (1-\mu) \sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{S}} d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) + \mu \sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_l) \in \mathcal{R}} \left[1 + d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l)\right]_+, \tag{4.1}$$

where $[a]_+ = \max(a, 0)$ for $a \in \mathbb{R}$; μ is the trade-off parameter. The constraints $\mathcal{S}$ and $\mathcal{R}$ are defined as follows:

$$\mathcal{S} = \big\{(\boldsymbol{x}_i, \boldsymbol{x}_j) : j \in \underset{a=1,\cdots,n}{\arg\min}_k\{d(\boldsymbol{x}_i, \boldsymbol{x}_a) : y_i = y_a\}\big\},$$

$$\mathcal{R} = \big\{(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_l) : (\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{S}, y_i \neq y_l\big\};$$

$\arg\min_k$ denotes the k minimal elements of a set and d denotes the Euclidean distance. $\boldsymbol{x}_j$ is termed the target neighbour of $\boldsymbol{x}_i$ and $\boldsymbol{x}_l$ is termed the impostor.

4.3.2 Metric learning with adversarial training (MLadv)

The objective function of LMNN (Eq. 4.1) can be interpreted as minimising a linear combination between the *empirical risk* $\frac{1}{n}\sum_{i=1}^{n}\ell(\boldsymbol{x}_i, y_i; \boldsymbol{M})$ and the regulariser on $\boldsymbol{M}$; the hinge loss ℓ of $[1 + d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l)]_+$ separates target neighbours and impostors by a unit margin, and the regulariser is chosen as $\sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. To address the issue of feature ambiguity faced by ordinal and nominal variables, we propose to model the unknown ambiguity as a perturbation of $\boldsymbol{x}_i$. Instead of the hinge loss, we consider the worst-case loss within a certain perturbation range and minimise the *empirical adversarial risk*:

$$\min_{\boldsymbol{M}} \frac{1}{n}\sum_{i=1}^{n} \max_{\boldsymbol{\delta}_i\in\Delta} \ell(\boldsymbol{x}_i + \boldsymbol{\delta}_i, y_i; \boldsymbol{M}). \tag{4.2}$$

$\boldsymbol{\delta}$ denotes the perturbation and its form is specified by set Δ. The optimal solution to the inner maximisation problem is termed the worst-case perturbation and denoted by $\boldsymbol{\delta}^\star$.

To fit ordinal and nominal variables, we need to incorporate their properties when defining the perturbation set Δ. A typical choice of Δ is the set of L_p-bounded perturbation, i.e. $||\boldsymbol{\delta}||_p \leq \varepsilon$, with $p = 1, 2, \infty$. However, a non-integer real-valued ε is not suitable for ordinal and nominal variables as it ignores the discrete nature of nominal variables and the ordering nature of ordinal variables. Therefore, we restrict $\boldsymbol{\delta}$ via the following two conditions: i) $||\boldsymbol{\delta}||_\infty = 1$; and ii) $||\boldsymbol{\delta}||_1 \leq \varepsilon, \varepsilon \in \mathbb{N}$. Since the loss function is linear in $\boldsymbol{\delta}$ as shown in Eq. 4.4, the first condition guarantees that the perturbed instance remains as an integer or a binary value. More crucially, the magnitude of one aligns with the source of feature ambiguity, which arises from non-rigorously defined set boundaries. In the example of film survey, the perturbation from ‘good’ to ‘fair’ matches the real-world decision-making process whereas replacing ‘good’ by ‘bad’ dramatically changes the original information. The second condition controls the level of perturbation. Integrating these two conditions, the perturbation $\boldsymbol{\delta}$ can change at most ε features of each instance.

To train the Mahalanobis distance, we form triplet constraints from both original and perturbed instances [83], and apply different loss functions to these triplets. For the original triplets, we adopt the loss function of LMNN and change the unit distance margin to an adjustable quantity τ. As we shall discuss in Section 4.3.4, τ determines how the instance space is divided into the support region and the adversarially vulnerable region. If the distance margin is satisfied by the triplet $(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_l)$, we will proceed to add perturbation to the instance $\boldsymbol{x}_i$. For the perturbed triplets, we adopt the perceptron loss [84]. Although the perceptron loss is rarely used in metric learning due to the lack of distance margin, it is sensible in our setting since the perturbation itself can serve as a margin in the instance space.

Integrating the above design of perturbation set and loss functions, we propose the following objective function for metric learning with adversarial training (MLadv):

$$\begin{aligned}\min_{\boldsymbol{M}\in\mathbb{S}_+^P} \lambda\|\boldsymbol{M}\|_1 &+ \frac{\mu}{|\mathcal{R}|}\sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j,\boldsymbol{x}_l)\in\mathcal{R}} \Big[\tau + d^2_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_l)\Big]_+ \\ &+ \frac{1-\mu}{|\mathcal{R}|}\sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j,\boldsymbol{x}_l)\in\mathcal{R}} \mathbb{1}[d^2_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_l) > d^2_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j)+\tau] \\ &\qquad \cdot \Big[\max_{\boldsymbol{\delta}_i:\|\boldsymbol{\delta}_i\|_\infty=1,\|\boldsymbol{\delta}_i\|_1\le\varepsilon} \{d^2_{\boldsymbol{M}}(\boldsymbol{x}_i+\boldsymbol{\delta}_i,\boldsymbol{x}_j) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i+\boldsymbol{\delta}_i,\boldsymbol{x}_l)\}\Big]_+,\end{aligned} \tag{4.3}$$

where $|\mathcal{R}|$ denote the numbers of triplets in the set $\mathcal{R}$; $\mathbb{1}[\cdot]$ is the indicator function which equals 1 if the condition is satisfied and 0 otherwise. The element-wise 1-norm (hereinafter abbreviated to L_1-norm), i.e. $\|\boldsymbol{M}\|_1 = \|\text{vec}(\boldsymbol{M})\|_1 = \sum_{m,n=1}^P \boldsymbol{M}_{mn}$, is used to regularise the complexity of the distance matrix. As proved in Section 4.3.5, this choice of regulariser is essential to guarantee that the number of samples required for the adversarially trained metric to generalise has the same order as that for the standard metric. $\lambda > 0$ is a trade-off parameter between the regularisation term and the loss function, and $\mu \in [0, 1]$ balances between the influence from original instances and perturbed instances. The triplet set $\mathcal{R}$ is constructed in the same way as LMNN, i.e. according to Eq. 4.3.1.

4.3.3 Optimisation algorithm

According to the Danskin's theorem [85], the gradient of the maximum of a differentiable function is given by the gradient of the function evaluated at the maximum point, i.e.

$$\nabla_{\boldsymbol{M}} \max_{\boldsymbol{\delta} \in \Delta} \ell(\boldsymbol{x} + \boldsymbol{\delta}, y; \boldsymbol{M}) = \nabla_{\boldsymbol{M}} \ell(\boldsymbol{x} + \boldsymbol{\delta}^\star, y; \boldsymbol{M}),$$

where $\boldsymbol{\delta}^\star = \arg\max_{\boldsymbol{\delta} \in \Delta} \ell(\boldsymbol{x} + \boldsymbol{\delta}, y; \boldsymbol{M})$; ∇ denotes the gradient. Therefore, the optimisation problem (Eq. 4.3) can be solved by first deriving a closed-form solution to the inner maximisation problem and then updating $\boldsymbol{M}$ via the proximal gradient descent algorithm.

4.3.3.1 Inner maximisation

The solution to the worst-case perturbation $\boldsymbol{\delta}_i^\star$ is derived by simplifying the objective function as follows: let $\boldsymbol{\delta}_i^\star = \arg\max_{\boldsymbol{\delta}_i : \|\boldsymbol{\delta}_i\|_\infty = 1, \|\boldsymbol{\delta}_i\|_1 \leq \varepsilon} \{d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \boldsymbol{\delta}_i, \boldsymbol{x}_j) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \boldsymbol{\delta}_i, \boldsymbol{x}_l)\}$, then

$$\begin{aligned} &\arg\max_{\boldsymbol{\delta}_i} \{d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \boldsymbol{\delta}_i, \boldsymbol{x}_j) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \boldsymbol{\delta}_i, \boldsymbol{x}_l)\} \\ \Leftrightarrow &\arg\max_{\boldsymbol{\delta}_i} \{d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) + 2\boldsymbol{\delta}_i^T \boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j)\} \\ \Leftrightarrow &\arg\max_{\boldsymbol{\delta}_i} \boldsymbol{\delta}_i^T \boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j) \end{aligned} \tag{4.4}$$

Under the constraint $\|\boldsymbol{\delta}_i\|_\infty = 1$, we have $\boldsymbol{\delta}_i^\star = \mathrm{sign}(\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j))$. With the additional constraint $\|\boldsymbol{\delta}_i\|_1 \leq \varepsilon, \varepsilon \in \mathbb{N}$, we have

$$\boldsymbol{\delta}^\star_{i,[k]} = \begin{cases} \mathrm{sign}(\boldsymbol{M}_{k\cdot}(\boldsymbol{x}_l - \boldsymbol{x}_j)) & \text{if } k \in \underset{a=1,\cdots,P}{\arg\max_\varepsilon} |\boldsymbol{M}_{a\cdot}(\boldsymbol{x}_l - \boldsymbol{x}_j)| \\ 0 & \text{otherwise} \end{cases}, \tag{4.5}$$

where $\boldsymbol{\delta}^{\star}_{i,[k]}$ denotes the kth element of the vector $\boldsymbol{\delta}^{\star}_{i}$; $\boldsymbol{M}_{k\cdot}$ denotes the kth row of $\boldsymbol{M}$; $\arg\max_{\varepsilon}$ denotes the ε maximal elements of a vector; $\operatorname{sign}(\boldsymbol{v})$ applies the sign function to each element of the vector $\boldsymbol{v}$ and $|\boldsymbol{v}|$ calculates elementwise absolute values. The solution of $\boldsymbol{\delta}^{\star}_{i,[k]}$ may not be unique. After sorting elements of $|\boldsymbol{M}_{a\cdot}(\boldsymbol{x}_l - \boldsymbol{x}_j)|$ in descending order, if there are multiple elements having the same value at the ϵth largest position, then one of them will be selected randomly and contributes to $\boldsymbol{\delta}^{\star}_{i,[k]}$.

4.3.3.2 Outer minimisation

Since the L_1-norm regularisation leads to a non-smooth objective function, the proximal gradient descent algorithm is adopted to optimise $\boldsymbol{M}$ in three steps. In the gradient descent step, $\boldsymbol{M}$ is updated as

$$\begin{aligned} \boldsymbol{M}^{t+\frac{1}{3}} &= \boldsymbol{M}^{t} - \eta^{t}\nabla\boldsymbol{M}|_{\boldsymbol{M}^{t}} \\ \nabla\boldsymbol{M} &= \frac{\mu}{|\mathcal{R}|}\sum_{\mathcal{R}}\alpha_{ijl}(\boldsymbol{X}_{ij} - \boldsymbol{X}_{il}) + \frac{1-\mu}{|\mathcal{R}|}\sum_{\mathcal{R}}(1-\alpha_{ijl})\alpha^{\star}_{ijl}(\boldsymbol{X}^{\star}_{ij} - \boldsymbol{X}^{\star}_{il}) \end{aligned} \tag{4.6}$$

where $\sum_{\mathcal{R}}$ is an abbreviation for $\sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j,\boldsymbol{x}_l)\in\mathcal{R}}$; $\alpha_{ijl} = \mathbb{1}[\tau + d^2_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_j) \geq d^2_{\boldsymbol{M}}(\boldsymbol{x}_i,\boldsymbol{x}_l)]$, $\alpha^{\star}_{ijl} = \mathbb{1}[d^2_{\boldsymbol{M}}(\boldsymbol{x}^{\star}_i,\boldsymbol{x}_j) \geq d^2_{\boldsymbol{M}}(\boldsymbol{x}^{\star}_i,\boldsymbol{x}_l)]$; $\boldsymbol{x}^{\star}_i = \boldsymbol{x}_i + \boldsymbol{\delta}^{\star}_i$; $\boldsymbol{X}_{ij} = (\boldsymbol{x}_i - \boldsymbol{x}_j)(\boldsymbol{x}_i - \boldsymbol{x}_j)^T, \boldsymbol{X}^{\star}_{ij} = (\boldsymbol{x}^{\star}_i - \boldsymbol{x}_j)(\boldsymbol{x}^{\star}_i - \boldsymbol{x}_j)^T$, and $\boldsymbol{X}_{il}, \boldsymbol{X}^{\star}_{il}$ are defined similarly. The learning rate η^t decays during training according to the exponential function $\exp(-0.99(1+0.01t))$. Next, we compute the proximal mapping for the L_1-norm regularisation, which is equivalent to applying the soft-thresholding operator to $\boldsymbol{M}^{t+\frac{1}{3}}$:

$$\boldsymbol{M}^{t+\frac{2}{3}}_{mn} = \operatorname{sign}(\boldsymbol{M}^{t+\frac{1}{3}}_{mn})[|\boldsymbol{M}^{t+\frac{1}{3}}_{mn}| - \lambda\eta^t]_{+}. \tag{4.7}$$

Finally, $\boldsymbol{M}$ is projected onto the cone of PSD matrices via eigendecomposition:

$$\begin{aligned} \boldsymbol{M}^{t+\frac{2}{3}} &= \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^T \\ \boldsymbol{M}^{t+1} &= \boldsymbol{V}\max(\boldsymbol{\Lambda},0)\boldsymbol{V}^T. \end{aligned} \tag{4.8}$$

The optimisation algorithm of the proposed method is summarised in Algorithm 1.

Algorithm 1: Metric learning with adversarial training

```
Input: triplet constraints $\mathcal{R}$, parameters $\lambda, \mu, \tau, \varepsilon$, maximum number of
       iterations $T$
Output: $\boldsymbol{M}^T$
Initialization: $\boldsymbol{M}^0 = \boldsymbol{I}$;
for $t = 1$ to $T$ do
    Compute worst-case perturbation $\boldsymbol{\delta}_i^\star$ according to Eq. 4.5;
    Perform gradient descent on $\boldsymbol{M}$ according to Eq. 4.6;
    Perform proximal mapping on $\boldsymbol{M}$ according to Eq. 4.7;
    Project $\boldsymbol{M}$ onto the PSD cone according to Eq. 4.8.
```

4.3.3.3 Computational complexity

We now analyse the computational complexity of the proposed method. MLadv has the same computational complexity as LMNN in calculating the distance of each triplet, performing gradient descent, and projecting onto the PSD cone; their total complexity equals $O(P^3 + nP^2 + |\mathcal{R}| \cdot P)$, where P is the feature dimension after variable encoding, n is the number of training instances, and $|\mathcal{R}|$ is the number of triplet constraints. The extra cost results from the sorting operation used to find the worst-case perturbation and the soft-thresholding operation used to perform the L_1-norm regularisation. The time complexity of the sorting step is $O(|\mathcal{R}| \cdot P^2 \log P)$ and that of the soft-thresholding step is $O(P^2)$. Overall, the time complexity of MLadv per iteration is $O(P^3 + |\mathcal{R}| \cdot P^2 \log P + nP^2 + |\mathcal{R}| \cdot P)$.

4.3.4 Geometric interpretation

We now provide a geometric interpretation for better understanding the effect of perturbation.

To start with, we rewrite the gradient of Eq. 4.6 by plugging in the worst-case perturbation derived in Eq. 4.5:

$$\frac{\mu}{|\mathcal{R}|}\sum_{\mathcal{R}} \mathbb{1}[d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) \leq \tau + d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)](\boldsymbol{X}_{ij} - \boldsymbol{X}_{il})$$
$$+\frac{1-\mu}{|\mathcal{R}|}\sum_{\mathcal{R}} \mathbb{1}[d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) + \tau < d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) \leq d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) + 2\|\boldsymbol{M}\boldsymbol{x}_{lj}\|_{1,[\varepsilon]}](\boldsymbol{X}^\star_{ij} - \boldsymbol{X}^\star_{il}), \tag{4.9}$$

where $\|\boldsymbol{M}\boldsymbol{x}_{lj}\|_{1,[\varepsilon]} = \sum \max_\varepsilon |\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j)|$ is the sum of ε maximal values in the vector $[|\boldsymbol{M}_{1\cdot}(\boldsymbol{x}_l - \boldsymbol{x}_j)|, \dots, |\boldsymbol{M}_{P\cdot}(\boldsymbol{x}_l - \boldsymbol{x}_j)|]$.

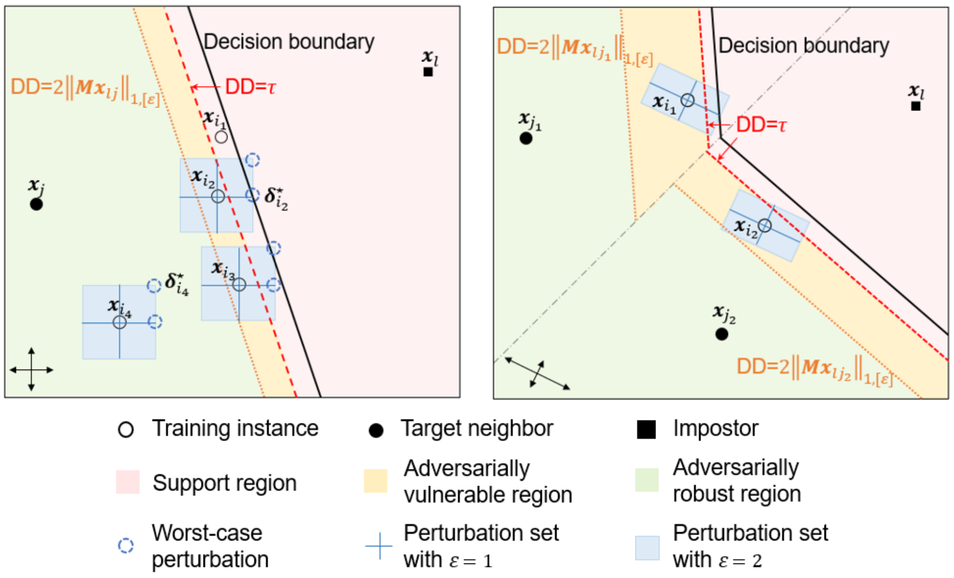


Figure 4.1: Illustration of MLadv. Instances are shown in the linearly mapped feature space induced by an isotropic $\boldsymbol{M}$ (*left*) and an anisotropic $\boldsymbol{M}$ (*right*). Left: LMNN learns $\boldsymbol{M}$ based on instances from the support region where the difference in squared distances (DD) $d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is not greater than τ. MLadv learns on additional instances from the adversarially vulnerable region where the instance may be misclassified after adding the worst-case perturbation. The regions are divided by two hyperplanes that are parallel to the decision boundary with DD of τ and $2\|\boldsymbol{M}\boldsymbol{x}_{lj}\|_{1,[\varepsilon]}$, respectively. Right: Instances lying above the grey dash-dotted line select $\boldsymbol{x}_{j_1}$ as NN and should be separated farther away from the decision boundary due to the high correlation between $\boldsymbol{M}$ and $\boldsymbol{x}_l - \boldsymbol{x}_{j_1}$. The effect of the metric is indicated by the arrow at the bottom-left corner.

Eq. 4.9 shows that, while LMNN and its variants learn the metric only on triplets where the impostor lies insufficiently far away from the instance, i.e. the difference in squared distances (DD) $d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is less than or equal to the required margin τ, the proposed method not only uses these information but also selectively exploits triplets that satisfy the margin constraint. In particular, the new selection criterion considers the correlation between the distance metric and $(\boldsymbol{x}_l - \boldsymbol{x}_j)$: if the correlation is high, i.e. the value of $\|\boldsymbol{M}\boldsymbol{x}_{lj}\|_{1,[\varepsilon]}$ is large, it is more likely this triplet will incur a loss and hence contribute to the gradient.

Figure 4.1 illustrates the above discussion with two figures. In both figures, we show all instances in the linearly mapped feature space induced by the Mahalanobis distance, and consider different positions of $\boldsymbol{x}_i$ with respect to fixed target neighbour $\boldsymbol{x}_j$ and impostor $\boldsymbol{x}_l$. The left figure illustrates which triplets are used in

LMNN and MLadv for calculating the gradient; for simplicity, the learned $\boldsymbol{M}$ is assumed to be isotropic, i.e. a scaled Euclidean distance. For $\boldsymbol{x}_{i_1}$, both methods use the triplet $(\boldsymbol{x}_{i_1}, \boldsymbol{x}_j, \boldsymbol{x}_l)$ since DD is less than τ. For $\boldsymbol{x}_{i_2}$ and $\boldsymbol{x}_{i_3}$, the methods differ. $(\boldsymbol{x}_{i_2}, \boldsymbol{x}_j, \boldsymbol{x}_l)$ and $(\boldsymbol{x}_{i_3}, \boldsymbol{x}_j, \boldsymbol{x}_l)$ satisfy the margin constraint and hence are not used in LMNN. However, they are used in the proposed MLadv as $\boldsymbol{x}_{i_2}$ and $\boldsymbol{x}_{i_3}$ may be misclassified in the presence of perturbation; the perturbation sets with $\varepsilon = 1$ and $\varepsilon = 2$ are indicated by the blue line and blue square, respectively. When $\varepsilon = 1$, $\boldsymbol{x}_{i_2}$ may be misclassified as the worst-case perturbation $\boldsymbol{\delta}^{\star}_{i_2}$ can drag the instance across the decision boundary; when $\varepsilon = 2$, both $\boldsymbol{x}_{i_2}$ and $\boldsymbol{x}_{i_3}$ may be misclassified. For $\boldsymbol{x}_{i_4}$, both methods ignore the triplet $(\boldsymbol{x}_{i_4}, \boldsymbol{x}_j, \boldsymbol{x}_l)$ since $\boldsymbol{x}_{i_4}$ remains far away from the decision boundary even after adding $\boldsymbol{\delta}^{\star}_{i_4}$.

The right figure presents the general case with an anisotropic $\boldsymbol{M}$ and multiple target neighbours, and illustrates the interaction between DD, $\boldsymbol{x}_l - \boldsymbol{x}_j$, and $\boldsymbol{M}$. Even though the DDs of $(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{j_1}, \boldsymbol{x}_l)$ and $(\boldsymbol{x}_{i_2}, \boldsymbol{x}_{j_2}, \boldsymbol{x}_l)$ are the same, $\boldsymbol{x}_{i_1}$ is not robust against the worst-case perturbation whereas $\boldsymbol{x}_{i_2}$ is. The reason is that $\boldsymbol{M}$ expands the horizontal distance, as indicated by the arrows at the bottom-left corner, and has a higher correlation with $\boldsymbol{x}_l - \boldsymbol{x}_{j_1}$ compared to $\boldsymbol{x}_l - \boldsymbol{x}_{j_2}$. This suggests that, for an instance to be invariant to the worst-case perturbation, the requirement of DD is determined locally with respect to $\boldsymbol{x}_l - \boldsymbol{x}_j$ and dynamically with respect to $\boldsymbol{M}$.

In summary, as points with the same DD form a separating hyperplane that is orthogonal to the line joining $\boldsymbol{x}_j$ and $\boldsymbol{x}_l$, the proposed method essentially divides the instance space into three regions according to the hyperplanes with DD of τ and $2\|\boldsymbol{M}\boldsymbol{x}_{lj}\|_{1,[\varepsilon]}$. It then makes use of instances from the support region and adversarially vulnerable region for learning the metric. The additional information from the latter region is particularly important for datasets with a small sample size.

4.3.5 Theoretical analysis

In this section, we provide the generalisation bound for metric learning trained in the adversarial setting. In essence, with the same form of loss function, adversarial training incurs a larger loss than standard training due to the addition of perturbation. Therefore, it is expected that the sample complexity would be higher in order to achieve the same generalisation performance.

We start by defining some notations. The adversarial loss is defined as

$$\tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_j, \boldsymbol{z}_l) = \mathbb{1}[y_i = y_j \neq y_l]\Big[\tau + \max_{\boldsymbol{\delta}_i : \|\boldsymbol{\delta}_i\|_\infty \leq \varepsilon} \{d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \boldsymbol{\delta}_i, \boldsymbol{x}_j) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \boldsymbol{\delta}_i, \boldsymbol{x}_l)\}\Big]_+ . \tag{4.10}$$

The generalisation bound studies the gap between the expected adversarial risk $\tilde{R}(\boldsymbol{M}) = \mathbb{E}_{(\boldsymbol{z}_i, \boldsymbol{z}_j, \boldsymbol{z}_l) \sim \mathcal{D}}[\tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_j, \boldsymbol{z}_l)]$ and the empirical adversarial risk $\tilde{R}_n(\boldsymbol{M}) = \frac{1}{n(n-1)(n-2)} \sum_{i \neq j \neq l} \tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_j, \boldsymbol{z}_l)$. Let $\boldsymbol{M}_{\boldsymbol{z}}$ denote the optimal solution to the learning problem:

$$\min_{\boldsymbol{M} \in \mathbb{S}^+_P} \tilde{R}_n(\boldsymbol{M}) + \lambda \|\boldsymbol{M}\|_1. \tag{4.11}$$

The generalisation bound of $\boldsymbol{M}_{\boldsymbol{z}}$ is given by the following theorem.

Theorem 6. Let $\boldsymbol{M}_{\boldsymbol{z}}$ be the solution to the problem (4.11). Then, with probability at least $1 - \delta$ we have that

$$\begin{aligned} \tilde{R}(\boldsymbol{M}_{\boldsymbol{z}}) - \tilde{R}_n(\boldsymbol{M}_{\boldsymbol{z}}) \leq & \frac{32\tau(x^2_{\max} + \varepsilon x_{\max})\sqrt{e \log P}}{\lambda\sqrt{n}} \\ & + \tau\Big[1 + \frac{x^2_{\max} + 2\varepsilon x_{\max}}{\lambda}\Big]\sqrt{\frac{2\ln(1/\delta)}{n}} + \frac{4\tau}{\sqrt{n}}, \end{aligned} \tag{4.12}$$

where $x_{\max} = \sup_{\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}} \|\boldsymbol{x} - \boldsymbol{x}'\|_\infty$.

Theorem 6 is established based on the Rademacher complexity [16] and U-statistics [86]; proof is given in Appendix 4.6.1.

We make three remarks here. First, by definition, the perturbation size is relatively small compared to $x_{\max}$, and therefore, $\varepsilon x_{\max} < x^2_{\max}$. This suggests that adversarial training does not largely increase the sample complexity. Second, as shown in the proof, if $\boldsymbol{M}$ is regularised via the common choice of the Frobenius norm, the sample complexity required by adversarial training will be higher than the standard training at a rate of $O(\sqrt{P})$. To avoid this sublinear dependence of sample complexity on feature dimension, L_1-norm is used as the regulariser. Third, Theorem 6 provides a general guarantee on the generalisation performance of triplet-based metric learning trained in the adversarial setting. The adversarial loss defined in Eq. 4.10 with $\varepsilon = 1$ unifies the two loss functions defined in our learning objective (Eq. 4.3). In other words, the metric learned through the proposed algorithm has a theoretical guarantee in terms of bounded generalisation gap.

4.4 Experiments

In this section, we first conduct experiments on a discretised dataset to evaluate the proposed method when facing the problems of small sample size and feature ambiguity. Then, we compare it with state-of-the-art methods on datasets with all ordinal variables and mixed ordinal-and-nominal variables.

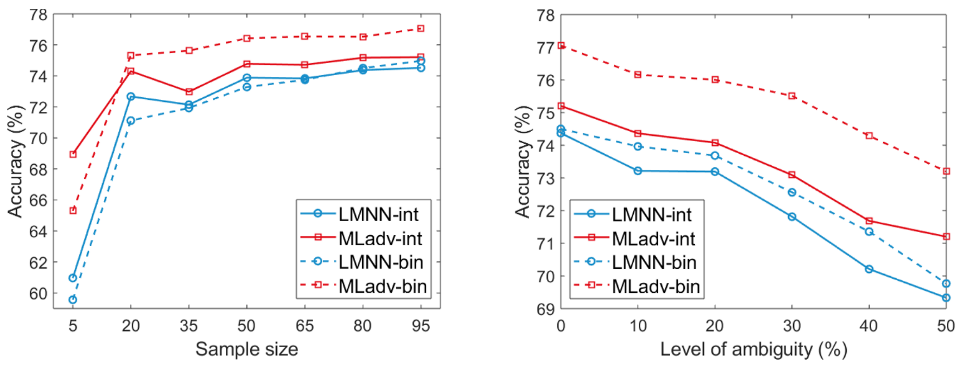
4.4.1 Parameter settings

The proposed method includes four hyperparameters, namely weight of original instances μ, regularisation parameter λ, distance margin τ, and perturbation size ε. Their values are identified via the random search strategy [59]. We sample 100 sets of values and select the one that gives the highest accuracy on the validation set. The range of each hyperparameter is as follows: $\mu \in [0, 1]$, $\lambda \in \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$, $\tau \in [0, \max \|\boldsymbol{x}_{lj}\|_1]$, $\varepsilon \in \{0, 1, \ldots, p^{\text{ord}} + 2p^{\text{nom}}\}$. The upper bound of τ is inspired by Eq. 4.9 with $\boldsymbol{M}$ initialised as the Euclidean distance. The upper bound of ε is chosen based on the fact that perturbing one ordinal level to its adjacent level or a nominal level to another level causes at most $p^{\text{ord}} + 2p^{\text{nom}}$ changes in encoded features. In addition, the initial learning rate is tuned for each dataset before optimising the hyperparameters. We search its value from $\{10^{-2}, 10^{-1}, \ldots, 10^{2}\}$ while holding $\mu, \tau = 1$ (i.e. replicating LMNN). The MATLAB code for our method is available at `http://github.com/xyang6/MLadv`.

Triplet constraints are constructed from 3 target neighbours and 10 nearest impostors calculated under the Euclidean distance. 3NN is used as the classifier.

4.4.2 Experiments with discretised features

The goal of this experiment is to understand the potential of the proposed method for data with a small training set and ambiguous features. Our experiment is based on the UCI dataset Magic, which has 10 real-valued features, 19020 instances, and



(a) effect of training sample size **(b)** effect of feature ambiguity

Figure 4.2: Evaluation of LMNN and MLadv on the discretised dataset Magic. Ordinal variables are encoded as integers ('int') or a set of binary values ('bin').

2 classes. All features are first discretised into ordinal features with five equal-frequency levels, and then encoded as integers (denoted as 'int') or as a set of binary values (denoted as 'bin'). We compare LMNN and the proposed method on both types of data.

4.4.2.1 Learning from small training sets

In this study, we build the training set by randomly selecting $5, 20, \ldots, 95$ instances from each class; the validation and test sets each include 9000 instances. The experiment is implemented 20 times and the mean accuracy is shown in Figure 4.2a; quantitative results, including the standard deviation, are provided in Appendix 4.6.2.2.

First, our MLadv outperforms LMNN over the whole range of training sample size, no matter what the encoding scheme is. Second, we see a clear advantage of MLadv over LMNN when the training set is small. Third, we notice that our MLadv performs better with binary encoding than integer encoding, when the sample size is larger than 20.

4.4.2.2 Learning under feature ambiguity

We move on to evaluate the method when encountering feature ambiguity. The experimental setting is same as before; the training sample size is selected as 80. To simulate ambiguity, for each feature, we select $10\%, 20\%, \ldots, 50\%$ instances whose ground-truth real values are closest to the discretisation threshold, and change their ordinal level to the adjacent level.

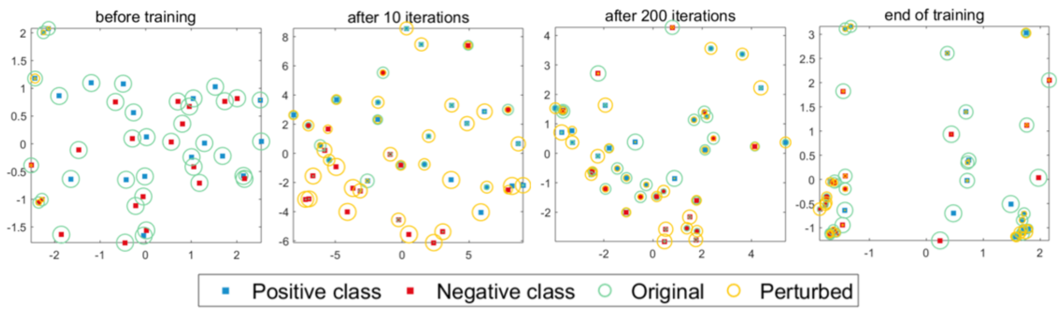


Figure 4.3: Demonstration of the training process of MLadv on Magic with binary encoding. Figures show the 2D embedding of the learned distance via multidimensional scaling. Sizes of green circles and yellow circles are proportional to the number of triplets violating the distance margin constraint and incurring a loss after adding the worst-case perturbation, respectively. As the training progresses, the metric becomes more robust against the perturbations and the difference between the intra-class distance and the inter-class distance becomes more remarkable.

Figure 4.2b shows the classification accuracy of this study. MLadv improves LMNN consistently over a wide range of ambiguity levels, and the performance gain becomes slightly larger as the ambiguity level increases.

4.4.2.3 Visualisation of training process

Our geometric interpretation suggests that MLadv considers additional triplets from the adversarially vulnerable region, which would be particularly valuable in the small-sized problem. In Figure 4.3, we present the training process of MLadv at different iterations. The multidimensional scaling [87] is used to embed the learned distance between 20 instances into two dimensions. Sizes of green circles and yellow circles are proportional to the number of triplets that do not satisfy the distance margin (i.e. second term of Eq. 4.3) and the number of triplets that incur a loss after adding the worst-case perturbation (i.e. third term of Eq. 4.3), respectively.

At the beginning of training, as instances of the same class are not well separated from instances of the different class, almost all triplets violate the distance margin constraints. Therefore, the metric is learned mostly from the original instances (as indicated by most points being in green circles). After 10 iterations, the majority of instances are closer to target neighbours than to impostors, but they are not robust to the worst-case perturbation (as indicated by a large number of yellow circles). Our method will continue using their information for metric learning. After 200 iterations, sizes of yellow circles become smaller, indicating that the learned metric becomes more robust. At the end of training, while some instances still violate the margin constraint, a large number of instances are surrounded by instances of the same class and locate far away from instances of the different class.

4.4.3 Experiments on real datasets

The goal of this experiment is to compare the proposed method with robust metric learning methods and ordinal metric learning methods under the conditions when the training sample size is small or the feature ambiguity is present. As ambiguities in categorical levels occur more frequently in ordinal variables than in nominal variables, our experiments only study datasets with all ordinal variables or with a mixture of ordinal and nominal variables.

4.4.3.1 Datasets and experimental settings

We use 6 datasets from UCI machine learning repository [57] and WEKA workbench [88]. Information on feature type, feature dimension, sample size and class information is listed in Table 4.1. Here, we explain the last column of ambiguity, which is assigned based on our understanding of the data. The degree of ambiguity is inherent in the data and may be inferred from the data source. Lecturer and Social Worker collect subjective ratings and assessments respectively, and hence may include a high level of ambiguity. Hayer-Roth and Lymphography are social data and medical data respectively; ambiguity is also likely to exist in these data. Car and Nursery are derived from a hierarchical decision model; their ambiguity levels are expected to be relatively low as there is an underlying rule behind these data.

Table 4.1: Characteristics of the datasets.

Dataset	Abb.	Feature Type	#Instances	p^{ord}	p^{nom}	#Classes	Ambiguity
Car	CA	ordinal	1728	6	0	4	low
Nursery	NU	ordinal+nominal	12960	6	2	4	low
Hayes-Roth	HR	ordinal+nominal	132	2	2	3	medium
Lymphography	LY	ordinal+nominal	148	3	15	4	medium
Lecturer	LE	ordinal	1000	4	0	5	high
Social Worker	SW	ordinal	1000	10	0	4	high

Each dataset is randomly split into the training, validation, and test sets. To simulate a small-sample environment, we set their proportions as 20%, 40%, 40% for all datasets except for the large dataset Nursery. For Nursery, 100 samples are selected as the training set, and the remaining samples are equally split into the validation and test sets. We repeat the random split 20 times, and report the mean value and standard deviation of classification accuracy.

We compare the proposed method with LMNN and three closely related methods. DRIFT [49] and AML [64] are robust metric learning methods that are designed to handle feature uncertainty for real-valued data. Ord-LMNN [74] adapts LMNN to ordinal variables by assuming a latent variable for each ordinal variable. Training procedures of these methods are specified in Appendix 4.6.2.1.

4.4.3.2 Results and discussions

Table 4.2 reports the classification accuracy of 3NN with the Mahalanobis distance learned from different methods. First, we see that the proposed method outperforms the baseline method LMNN, regardless of the encoding scheme. Second, we compare MLadv with the existing ordinal metric learning method Ord-LMNN. Ord-LMNN considers the order relation of ordinal variables and is effective on datasets Balance Scale and Car. However, as the method estimates the distributional parameters for each feature, its effectiveness highly depends on the data quality. When the ambiguity level is high, the accuracy of Ord-LMNN becomes even worse than the baseline whereas our method remains competitive. Third, the robust metric learning method DRIFT achieves a high accuracy when the feature ambiguity is high. However, as the method ignores the properties of ordinal and nominal variables, its performance is inferior to our method. Overall, our method achieves the best or second-best performance on each dataset.

We make a final remark on the encoding scheme and practicability of the proposed method. On most datasets, MLadv-bin is superior to MLadv-int. We hypothesise that, as binary encoding represents the data by using more features, the expressive power of the metric increases and hence may improve the discriminability. While the two encoding schemes are evaluated separately in our experiment,

Table 4.2: Classification accuracy (mean value±standard deviation) of 3NN with different metric learning methods. The best methods are shown in bold and the second best ones are underlined.

	LMNN-int	LMNN-bin	DRIFT	AML	Ord-LMNN	**MLadv-int**	**MLadv-bin**
				low level of ambiguity			
CA	89.94±1.31	92.24±0.89	90.24±1.17	88.84±1.13	**93.94±1.43**	90.13±1.32	92.90±1.13
NU	85.73±1.68	86.11±1.78	86.01±2.02	79.83±3.11	**87.54±1.45**	86.65±2.12	86.67±1.48
				medium level of ambiguity			
HR	71.83±10.72	76.42±6.80	71.34±10.37	65.98±7.85	75.12±9.55	74.51±10.06	**78.58±5.94**
LY	78.51±7.15	79.91±6.65	83.16±6.40	68.25±17.57	74.56±9.17	82.37±3.58	**83.33±4.85**
				high level of ambiguity			
LE	55.08±2.55	54.83±2.53	55.64±3.00	55.61±2.28	53.03±3.27	**55.90±2.70**	55.61±3.00
SW	50.00±2.61	50.58±2.30	50.73±2.93	50.50±2.07	48.68±2.82	51.10±2.15	**51.91±3.09**

they could be determined at the step of choosing the initial learning rate in practical applications. In other words, there is no need to tune hyperparameters twice. Except for Lymphography, this early decision can always find the optimal method between MLadv-int and MLadv-bin.

4.5 Conclusions and future work

In this chapter, we propose that adversarial training with a deliberately designed perturbation set can enhance triplet-based metric learning methods in mitigating the problems of high feature ambiguity and small sample size faced by ordinal and mixed ordinal-and-nominal data. Experiments on real datasets verify the efficacy of our method. Moreover, we discuss the effect of adversarial training from a geometric perspective. Compared with previous methods using a fixed margin constraint, the proposed method provides a flexible way of adjusting the margin, which is data-dependent and dynamic. Furthermore, we guarantee the theoretical performance of adversarially trained metric. It shows that the adversarial training does not increase the order of sample complexity required for the metric to generalise, provided the suitable regulariser of L_1-norm is applied.

This study focuses only on categorical features. Generalising the method to a mix of categorical and numerical features would be practically valuable. In addition, metric learning comprises a loss function and a regulariser, and we only tailor the loss function to incorporate the properties of categorical features. In the future, we will consider designing specific regularisers for this feature type.

4.6 Appendix

4.6.1 Proof of generalisation bound

The generalisation bound is established based on the Rademacher complexity [16]. With the same form of loss function, adversarial training incurs a larger loss than standard training due to the addition of perturbation. Therefore, we need to first show that the adversarial loss and the Rademacher complexity of the adversarial loss function class are both bounded. After that, we prove the bound following the works of [89; 90; 86].

For completeness, we list all notations used in the proof as follows.

Inner product, vector norm, and matrix norm: $\langle \boldsymbol{X}, \boldsymbol{Y} \rangle = \text{trace}(\boldsymbol{X}^T \boldsymbol{Y})$ denotes the Frobenius inner product of matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$. $\|\boldsymbol{v}\|_1$ and $\|\boldsymbol{v}\|_2$ denote the L_1-norm and L_2-norm of a vector $\boldsymbol{v}$, respectively; $\|\boldsymbol{M}\|_1 = \sum_{mn} \boldsymbol{M}_{mn}$ and $\|\boldsymbol{M}\|_F$ denote the (elementwise) L_1-norm and the Frobenius norm of a matrix $\boldsymbol{M}$, respectively. Given any matrix norm $\|\cdot\|$, its dual norm is defined as $\|\boldsymbol{M}\|_* = \sup\{\langle \boldsymbol{M}, \boldsymbol{X} \rangle : \|\boldsymbol{X}\| \leq 1\}$.

Adversarial loss of triplet-based metric learning problems with L_∞-bounded perturbations:

$$\tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_j, \boldsymbol{z}_l) = \mathbb{1}[y_i = y_j \neq y_l][\tau + \max_{\boldsymbol{\delta}_i : \|\boldsymbol{\delta}_i\|_\infty \leq \varepsilon} \{d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \boldsymbol{\delta}_i, \boldsymbol{x}_j) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \boldsymbol{\delta}_i, \boldsymbol{x}_l)\}]_+ \tag{4.13}$$

Expected adversarial risk:

$$\tilde{R}(\boldsymbol{M}) = \mathbb{E}_{(\boldsymbol{z}_i, \boldsymbol{z}_j, \boldsymbol{z}_l) \sim \mathcal{D}}[\tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_j, \boldsymbol{z}_l)]$$

Empirical adversarial risk:

$$\tilde{R}_n(\boldsymbol{M}) = \frac{1}{n(n-1)(n-2)} \sum_{i \neq j \neq l} \tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_j, \boldsymbol{z}_l)$$

Rademacher complexity [91]: For any function class $\mathcal{F}$, given a sample set $\boldsymbol{z}^n$ of size n, the empirical Rademacher complexity of $\mathcal{F}$ with respect to $\boldsymbol{z}^n$ is defined as:

$$\hat{\mathcal{R}}_n(\mathcal{F}) = \frac{1}{n} \mathbb{E}_{\boldsymbol{\sigma}} \Big[\sup_{f \in \mathcal{F}} \sum_{i=1}^{n} \sigma_i f(\boldsymbol{z}_i) \Big],$$

where $\sigma_1, \ldots, \sigma_n$ are Rademacher variables, i.i.d. according to $\mathbb{P}(\sigma_i = 1) = \mathbb{P}(\sigma_i = -1) = \frac{1}{2}$. The Rademacher complexity is the expectation of the empirical Rademacher complexity over all samples of size n drawn according to $\mathcal{D}$: $\mathcal{R}_n(\mathcal{F}) = \mathbb{E}_{\boldsymbol{z}^n \sim \mathcal{D}}[\hat{\mathcal{R}}_n(\mathcal{F})]$.

We propose to learn the distance metric by optimising the following objective function:

$$\min_{\boldsymbol{M} \in \mathbb{S}_P^+} \tilde{R}_n(\boldsymbol{M}) + \lambda \|\boldsymbol{M}\|_1. \tag{4.14}$$

The optimal solution to Eq. 4.14 is denoted as $\boldsymbol{M}_{\boldsymbol{z}}$. Since $\tilde{R}_n(\boldsymbol{M}_{\boldsymbol{z}}) + \lambda \|\boldsymbol{M}_{\boldsymbol{z}}\|_1 \leq \tilde{R}_n(\mathbf{0}) + \lambda \|\mathbf{0}\|_1 \leq \tau$, where $\mathbf{0}$ denotes the zero matrix, we can restrict the parameter space of $\boldsymbol{M}$ as:

$$\mathcal{H} = \Big\{ \boldsymbol{M} : \boldsymbol{M} \in \mathbb{S}_P^+, \|\boldsymbol{M}\|_1 \leq \frac{\tau}{\lambda} \Big\}.$$

The following lemma shows that the adversarial loss is bounded.

Lemma 1. The adversarial loss of Eq. 4.13 is upper bounded:

$$\sup_{\boldsymbol{z}_i, \boldsymbol{z}_j, \boldsymbol{z}_l \in \mathcal{Z}} \sup_{\boldsymbol{M} \in \mathcal{H}} \tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_j, \boldsymbol{z}_l) \leq \tau \Big[1 + \frac{2\varepsilon x_{\text{max}}}{\lambda} + \frac{x_{\text{max}}^2}{\lambda} \Big], \tag{4.15}$$

where $x_{\text{max}} = \sup_{\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}} \|\boldsymbol{x} - \boldsymbol{x}'\|_\infty$.

Proof.

$$
\begin{aligned}
&\tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_j, \boldsymbol{z}_l)\\
\leq&[\tau + \max_{\boldsymbol{\delta}_i:\|\boldsymbol{\delta}_i\|_\infty \leq \varepsilon} \{d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \boldsymbol{\delta}_i, \boldsymbol{x}_j) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \boldsymbol{\delta}_i, \boldsymbol{x}_l)\}]_+\\
=&[\tau + d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \varepsilon\,\mathrm{sign}(\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j)), \boldsymbol{x}_j) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \varepsilon\,\mathrm{sign}(\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j)), \boldsymbol{x}_l)]_+\\
=&[\tau + d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_l) + 2\varepsilon\ \mathrm{sign}(\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j))^T\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j)]_+\\
\leq&\tau + \langle \boldsymbol{M}, \boldsymbol{X}_{ij}\rangle + 2\varepsilon\langle \boldsymbol{M}, (\boldsymbol{x}_l - \boldsymbol{x}_j)\,\mathrm{sign}(\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j))^T\rangle\\
\overset{(a)}{\leq}&\tau + \|\boldsymbol{M}\|_1\|\boldsymbol{X}_{ij}\|_\infty + 2\varepsilon\|\boldsymbol{M}\|_1\|(\boldsymbol{x}_l - \boldsymbol{x}_j)\,\mathrm{sign}(\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j))^T\|_\infty\\
\Rightarrow& \sup_{\boldsymbol{z}_i,\boldsymbol{z}_j,\boldsymbol{z}_l \in \mathcal{Z}} \sup_{\boldsymbol{M}\in\mathcal{H}} \tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_j, \boldsymbol{z}_l) \overset{(b)}{\leq} \tau + \frac{\tau}{\lambda}x^2_{\max} + 2\varepsilon\frac{\tau}{\lambda}x_{\max}
\end{aligned}
$$

□

Remark: Step (a) of the above proof makes use of the dual norm of $\|\boldsymbol{M}\|_1$, and step (b) bounds $\|(\boldsymbol{x}_l - \boldsymbol{x}_j)\,\mathrm{sign}(\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j))^T\|_\infty$ by $x_{\max}$. If we regularise $\boldsymbol{M}$ via the Frobenius norm, the dual norm will be the Frobenius norm, and $\|(\boldsymbol{x}_l - \boldsymbol{x}_j)\,\mathrm{sign}(\boldsymbol{M}(\boldsymbol{x}_l - \boldsymbol{x}_j))^T\|_F \leq \sqrt{P}\sup_{\boldsymbol{x},\boldsymbol{x}'\in\mathcal{X}} \|\boldsymbol{x}_l - \boldsymbol{x}_j\|_2$. This sublinear dependence of the loss function on the feature dimension is unavoidable, even after normalising all instances to have a unit length with respect to the L_2-norm.

The following lemma shows that the Rademacher complexity of the adversarial loss function class is bounded.

Lemma 2. Let $\mathcal{R}_n = \frac{1}{n}\mathbb{E}_{\boldsymbol{z}^n,\boldsymbol{\sigma}}[\sup_{\boldsymbol{M}\in\mathcal{H}} \sum_{i=1}^n \sigma_i \tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}'_i, \boldsymbol{z}''_i)]$, where $\boldsymbol{z}'_i, \boldsymbol{z}''_i$ are independent of $\boldsymbol{z}_i$. Then,

$$\mathcal{R}_n \leq \frac{8\tau(x^2_{\max} + \varepsilon x_{\max})\sqrt{e\log P}}{\lambda\sqrt{n}} + \frac{\tau}{\sqrt{n}}. \tag{4.16}$$

Proof. The proof builds on the contraction lemma of the Rademacher complexity [92] and the Khinchin–Kahane inequality (Lemma 9 of [89]).

$$
\begin{aligned}
&\mathbb{E}_{\boldsymbol{z}^n,\boldsymbol{\sigma}} \sup_{\boldsymbol{M}\in\mathcal{H}} \sum_{i=1}^n \sigma_i \tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_i', \boldsymbol{z}_i'') \\
=&\mathbb{E}_{\boldsymbol{z}^n,\boldsymbol{\sigma}} \sup_{\boldsymbol{M}\in\mathcal{H}} \sum_{i=1}^n \sigma_i \mathbb{1}[y = y' \neq y''][\tau + d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \boldsymbol{\delta}_i^\star, \boldsymbol{x}_i') - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \boldsymbol{\delta}_i^\star, \boldsymbol{x}_i'')]_+ \\
\leq&\mathbb{E}_{\boldsymbol{z}^n,\boldsymbol{\sigma}} \sup_{\boldsymbol{M}\in\mathcal{H}} \Big| \sum_{i=1}^n \sigma_i \big([\tau + d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \boldsymbol{\delta}_i^\star, \boldsymbol{x}_i') - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \boldsymbol{\delta}_i^\star, \boldsymbol{x}_i'')]_+ - \tau\big) \Big| \\
&+ \mathbb{E}_{\boldsymbol{\sigma}} \Big| \sum_{i=1}^n \sigma_i \tau \Big| \\
\overset{(a)}{\leq}&2\mathbb{E}_{\boldsymbol{z}^n,\boldsymbol{\sigma}} \sup_{\boldsymbol{M}\in\mathcal{H}} \Big| \sum_{i=1}^n \sigma_i \big[d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \boldsymbol{\delta}_i^\star, \boldsymbol{x}_i') - d^2_{\boldsymbol{M}}(\boldsymbol{x}_i + \boldsymbol{\delta}_i^\star, \boldsymbol{x}_i'')\big] \Big| + \mathbb{E}_{\boldsymbol{\sigma}} \Big| \sum_{i=1}^n \sigma_i \tau \Big| \\
\leq&4\mathbb{E}_{\boldsymbol{z}^n,\boldsymbol{\sigma}} \sup_{\boldsymbol{M}\in\mathcal{H}} \Big| \sum_{i=1}^n \sigma_i d^2_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_i') \Big| \\
&+ 4\varepsilon\mathbb{E}_{\boldsymbol{z}^n,\boldsymbol{\sigma}} \sup_{\boldsymbol{M}\in\mathcal{H}} \Big| \sum_{i=1}^n \sigma_i \operatorname{sign}(\boldsymbol{M}(\boldsymbol{x}_i'' - \boldsymbol{x}_i'))^T \boldsymbol{M}(\boldsymbol{x}_i'' - \boldsymbol{x}_i') \Big| + \mathbb{E}_{\boldsymbol{\sigma}} \Big| \sum_{i=1}^n \sigma_i \tau \Big| \\
\leq&\frac{4\tau}{\lambda}\mathbb{E}_{\boldsymbol{z}^n,\boldsymbol{\sigma}} \Big\| \sum_{i=1}^n \sigma_i (\boldsymbol{x}_i - \boldsymbol{x}_i')(\boldsymbol{x}_i - \boldsymbol{x}_i')^T \Big\|_\infty \\
&+ \frac{4\tau\varepsilon}{\lambda}\mathbb{E}_{\boldsymbol{z}^n,\boldsymbol{\sigma}} \Big\| \sum_{i=1}^n \sigma_i (\boldsymbol{x}_i'' - \boldsymbol{x}_i') \operatorname{sign}(\boldsymbol{M}(\boldsymbol{x}_i'' - \boldsymbol{x}_i'))^T \Big\|_\infty + \tau\mathbb{E}_{\boldsymbol{\sigma}} \Big| \sum_{i=1}^n \sigma_i \Big| \qquad (4.17)
\end{aligned}
$$

Step (a) is obtained by applying the Talagrand's contraction lemma.

Each term in the last line of inequality (4.17) can be bounded by applying the Khinchin–Kahane inequality. Here, we show the bound of the second term; bounds of the first and third terms are derived in [89] and results are listed below for completeness.

$$
\begin{aligned}
&\mathbb{E}_{\boldsymbol{z}^n,\boldsymbol{\sigma}} \Big\| \sum_{i=1}^n \sigma_i (\boldsymbol{x}_i'' - \boldsymbol{x}_i') \operatorname{sign}(\boldsymbol{M}(\boldsymbol{x}_i'' - \boldsymbol{x}_i'))^T \Big\|_\infty \\
\leq&\mathbb{E}_{\boldsymbol{z}^n,\boldsymbol{\sigma}} \Big\| \sum_{i=1}^n \sigma_i (\boldsymbol{x}_i'' - \boldsymbol{x}_i') \operatorname{sign}(\boldsymbol{M}(\boldsymbol{x}_i'' - \boldsymbol{x}_i'))^T \Big\|_q \quad \text{for any } 1 < q < \infty \\
=&\mathbb{E}_{\boldsymbol{z}^n,\boldsymbol{\sigma}} \Big[\sum_{k_1,k_2=1}^P \Big| \sum_{i=1}^n \sigma_i (\boldsymbol{x}_{i,[k_1]}'' - \boldsymbol{x}_{i,[k_1]}') \operatorname{sign}(\boldsymbol{M}_{k_2\cdot}(\boldsymbol{x}_i'' - \boldsymbol{x}_i')) \Big|^q \Big]^{\frac{1}{q}}
\end{aligned}
$$

$$
\begin{aligned}
&\leq \mathbb{E}_{\boldsymbol{z}^n}\Big[\sum_{k_1,k_2=1}^{P}\mathbb{E}_{\boldsymbol{\sigma}}\big|\sum_{i=1}^{n}\sigma_i(\boldsymbol{x}''_{i,[k_1]}-\boldsymbol{x}'_{i,[k_1]})\operatorname{sign}(\boldsymbol{M}_{k_2\cdot}(\boldsymbol{x}''_i-\boldsymbol{x}'_i))\big|^q\Big]^{\frac{1}{q}}\\
&\overset{(b)}{\leq} \mathbb{E}_{\boldsymbol{z}^n}\Big[\sum_{k_1,k_2=1}^{P}(q-1)^{\frac{q}{2}}\Big(\mathbb{E}_{\boldsymbol{\sigma}}\big|\sum_{i=1}^{n}\sigma_i(\boldsymbol{x}''_{i,[k_1]}-\boldsymbol{x}'_{i,[k_1]})\operatorname{sign}(\boldsymbol{M}_{k_2\cdot}(\boldsymbol{x}''_i-\boldsymbol{x}'_i))\big|^2\Big)^{\frac{q}{2}}\Big]^{\frac{1}{q}}\\
&= \mathbb{E}_{\boldsymbol{z}^n}\Big[\sum_{k_1,k_2=1}^{P}(q-1)^{\frac{q}{2}}\Big(\sum_{i=1}^{n}\big(\boldsymbol{x}''_{i,[k_1]}-\boldsymbol{x}'_{i,[k_1]}\big)^2\big(\operatorname{sign}(\boldsymbol{M}_{k_2\cdot}(\boldsymbol{x}''_i-\boldsymbol{x}'_i))\big)^2\Big)^{\frac{q}{2}}\Big]^{\frac{1}{q}}\\
&\leq q^{\frac{1}{2}}\mathbb{E}_{\boldsymbol{z}^n}\Big[\sum_{k_1,k_2=1}^{P}\Big(\sum_{i=1}^{n}\sup_{\boldsymbol{x},\boldsymbol{x}'\in\mathcal{X}}\|\boldsymbol{x}-\boldsymbol{x}'\|_\infty^2\Big)^{\frac{q}{2}}\Big]^{\frac{1}{q}}\\
&= q^{\frac{1}{2}}P^{\frac{2}{q}}\sup_{\boldsymbol{x},\boldsymbol{x}'\in\mathcal{X}}\|\boldsymbol{x}-\boldsymbol{x}'\|_\infty\sqrt{n}\\
&\overset{(c)}{=} 2\sup_{\boldsymbol{x},\boldsymbol{x}'\in\mathcal{X}}\|\boldsymbol{x}-\boldsymbol{x}'\|_\infty\sqrt{en\log P}
\end{aligned}
$$

$$
\mathbb{E}_{\boldsymbol{z}^n,\boldsymbol{\sigma}}\Big\|\sum_{i=1}^{n}\sigma_i(\boldsymbol{x}_i-\boldsymbol{x}'_i)(\boldsymbol{x}_i-\boldsymbol{x}'_i)^T\Big\|_\infty \leq 2\sup_{\boldsymbol{x},\boldsymbol{x}'\in\mathcal{X}}\|\boldsymbol{x}-\boldsymbol{x}'\|_\infty^2\sqrt{en\log P}
$$

$$
\mathbb{E}_{\boldsymbol{\sigma}}\Big|\sum_{i=1}^{n}\sigma_i\Big| \leq \sqrt{n} \tag{4.18}
$$

In step (b), we apply the Khinchin–Kahane inequality with $2 < q < \infty$. In step (c), we set $q = 4\log P$. Putting results of (4.18) into the inequality (4.17) gives the bound of (4.16). □

We now prove the generalisation bound of $\boldsymbol{M}_{\boldsymbol{z}}$.

Theorem 7. Let $\boldsymbol{M}_{\boldsymbol{z}}$ be the solution to the problem (4.14). Then, for any $0 < \delta < 1$, with probability $1-\delta$ we have that

$$
\begin{aligned}
\tilde{R}(\boldsymbol{M}_{\boldsymbol{z}}) - \tilde{R}_n(\boldsymbol{M}_{\boldsymbol{z}}) \leq{}& \frac{32\tau(x_{\max}^2+\varepsilon x_{\max})\sqrt{e\log P}}{\lambda\sqrt{n}}\\
&+\tau\Big[1+\frac{x_{\max}^2+2\varepsilon x_{\max}}{\lambda}\Big]\sqrt{\frac{2\ln(1/\delta)}{n}}+\frac{4\tau}{\sqrt{n}},
\end{aligned}
\tag{4.19}
$$

where $x_{\max} = \sup_{\boldsymbol{x},\boldsymbol{x}'\in\mathcal{X}}\|\boldsymbol{x}-\boldsymbol{x}'\|_\infty$.

Proof.

Step 1: We bound the difference between $\tilde{R}(\boldsymbol{M}_{\boldsymbol{z}}) - \tilde{R}_n(\boldsymbol{M}_{\boldsymbol{z}})$ and $\mathbb{E}_{\boldsymbol{z}^n}\sup_{\boldsymbol{M}\in\mathcal{H}}[\tilde{R}(\boldsymbol{M}) - \tilde{R}_n(\boldsymbol{M})]$ via the McDiarmid's inequality [93].

First, we observe that $\tilde{R}(\boldsymbol{M_z}) - \tilde{R}_n(\boldsymbol{M_z}) \leq \sup_{\boldsymbol{M}\in\mathcal{H}}[\tilde{R}(\boldsymbol{M}) - \tilde{R}_n(\boldsymbol{M})]$. Next, we apply the McDiarmid's inequality to bound the difference between $\sup_{\boldsymbol{M}\in\mathcal{H}}[\tilde{R}(\boldsymbol{M}) - \tilde{R}_n(\boldsymbol{M})]$ and $\mathbb{E}_{\boldsymbol{z}^n}\sup_{\boldsymbol{M}\in\mathcal{H}}[\tilde{R}(\boldsymbol{M}) - \tilde{R}_n(\boldsymbol{M})]$, where $\mathbb{E}_{\boldsymbol{z}^n}$ denotes the expectation with respect to the training sample set $\boldsymbol{z}^n$. An essential condition of the McDiarmid's inequality is that the function $\sup_{\boldsymbol{M}\in\mathcal{H}}[\tilde{R}(\boldsymbol{M}) - \tilde{R}_n(\boldsymbol{M})]$ has bounded differences, which is shown as follows. Let $\boldsymbol{z}^n = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_{k-1}, \boldsymbol{z}_k, \boldsymbol{z}_{k+1}, \ldots, \boldsymbol{z}_n)$ and $\boldsymbol{z}^{n\prime} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_{k-1}, \boldsymbol{z}'_k, \boldsymbol{z}_{k+1}, \ldots, \boldsymbol{z}_n)$ be two training sample sets that differ in one sample. Combining the result of Lemma 1, we have the following inequality:

$$\begin{aligned}
&\Big|\sup_{\boldsymbol{M}\in\mathcal{H}}[\tilde{R}(\boldsymbol{M};\boldsymbol{z}^n) - \tilde{R}_n(\boldsymbol{M};\boldsymbol{z}^n)] - \sup_{\boldsymbol{M}\in\mathcal{H}}[\tilde{R}(\boldsymbol{M};\boldsymbol{z}^{n\prime}) - \tilde{R}_n(\boldsymbol{M};\boldsymbol{z}^{n\prime})]\Big| \\
\leq&\Big|\sup_{\boldsymbol{M}\in\mathcal{H}}\tilde{R}_n(\boldsymbol{M};\boldsymbol{z}^n) - \sup_{\boldsymbol{M}\in\mathcal{H}}\tilde{R}_n(\boldsymbol{M};\boldsymbol{z}^{n\prime})\Big| \\
=&\frac{1}{n(n-1)(n-2)}\sup_{\boldsymbol{M}\in\mathcal{H}}\sum_{k\neq j\neq l}|\tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_k,\boldsymbol{z}_j,\boldsymbol{z}_l) - \tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}'_k,\boldsymbol{z}_j,\boldsymbol{z}_l)| \\
\leq&\frac{1}{n(n-1)(n-2)}\sup_{\boldsymbol{M}\in\mathcal{H}}\sum_{k\neq j\neq l}(|\tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_k,\boldsymbol{z}_j,\boldsymbol{z}_l)| + |\tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}'_k,\boldsymbol{z}_j,\boldsymbol{z}_l)|) \\
\leq&\frac{2}{n}\sup_{\boldsymbol{M}\in\mathcal{H}}\tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_k,\boldsymbol{z}_j,\boldsymbol{z}_l) \\
\leq&\frac{2\tau}{n}\Big[1 + \frac{2\varepsilon x_{\max}}{\lambda} + \frac{x_{\max}^2}{\lambda}\Big].
\end{aligned}$$

Applying the McDiarmid's inequality to the term $\sup_{\boldsymbol{M}\in\mathcal{H}}[\tilde{R}(\boldsymbol{M}) - \tilde{R}_n(\boldsymbol{M})]$, with probability $1-\delta$ there holds

$$\begin{aligned}
&\sup_{\boldsymbol{M}\in\mathcal{H}}[\tilde{R}(\boldsymbol{M}) - \tilde{R}_n(\boldsymbol{M})] \\
\leq&\mathbb{E}_{\boldsymbol{z}^n}\sup_{\boldsymbol{M}\in\mathcal{H}}[\tilde{R}(\boldsymbol{M}) - \tilde{R}_n(\boldsymbol{M})] + \tau\Big[1 + \frac{2\varepsilon x_{\max}}{\lambda} + \frac{x_{\max}^2}{\lambda}\Big]\sqrt{\frac{2\ln(1/\delta)}{n}}.
\end{aligned} \tag{4.20}$$

Step 2: We bound the expectation term $\mathbb{E}_{\boldsymbol{z}^n}\sup_{\boldsymbol{M}\in\mathcal{H}}[\tilde{R}(\boldsymbol{M}) - \tilde{R}_n(\boldsymbol{M})]$ by reducing the analysis of non-i.i.d. triplets to i.i.d. random variables via the U-statistic [94] and symmetrising with the introduction of Rademacher variables [16].

First, based on Lemma 4 of [86], we can derive the following inequality:

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{z}^n} \sup_{\boldsymbol{M}\in\mathcal{H}} [\tilde{R}(\boldsymbol{M}) - \tilde{R}_n(\boldsymbol{M})] \\
=&\mathbb{E}_{\boldsymbol{z}^n} \sup_{\boldsymbol{M}\in\mathcal{H}} \Big[\tilde{R}(\boldsymbol{M}) - \frac{1}{n(n-1)(n-2)} \sum_{i\neq j\neq l} \tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_j, \boldsymbol{z}_l)\Big] \\
\leq&\mathbb{E}_{\boldsymbol{z}^n} \sup_{\boldsymbol{M}\in\mathcal{H}} \Big[\tilde{R}(\boldsymbol{M}) - \frac{1}{\lfloor\frac{n}{3}\rfloor} \sum_{i=1}^{\lfloor\frac{n}{3}\rfloor} \tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_{\lfloor\frac{n}{3}\rfloor+i}, \boldsymbol{z}_{\lfloor\frac{n}{3}\rfloor+i})\Big],
\end{aligned} \tag{4.21}$$

where $\lfloor\cdot\rfloor$ denotes the floor function. For simplicity, define $\bar{R}_n(\boldsymbol{M}) = \frac{1}{\lfloor\frac{n}{3}\rfloor} \sum_{i=1}^{\lfloor\frac{n}{3}\rfloor} \tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_{\lfloor\frac{n}{3}\rfloor+i}, \boldsymbol{z}_{\lfloor\frac{n}{3}\rfloor+i})$.

Next, we symmetrise by replacing $\tilde{R}(\boldsymbol{M})$ with $\mathbb{E}_{\bar{\boldsymbol{z}}^n}[\bar{R}_n(\boldsymbol{M})]$. Let $\bar{\boldsymbol{z}}^n = (\bar{\boldsymbol{z}}_1, \dots, \bar{\boldsymbol{z}}_n)$ denote another training set, where $\bar{\boldsymbol{z}}_i$'s are independent of each other and independent of $\boldsymbol{z}_i$'s. Then,

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{z}^n} \sup_{\boldsymbol{M}\in\mathcal{H}} [\tilde{R}(\boldsymbol{M}) - \bar{R}_n(\boldsymbol{M})] =&\mathbb{E}_{\boldsymbol{z}^n} \sup_{\boldsymbol{M}\in\mathcal{H}} \big[\mathbb{E}_{\bar{\boldsymbol{z}}^n}[\bar{R}_n(\boldsymbol{M}; \bar{\boldsymbol{z}}^n)] - \bar{R}_n(\boldsymbol{M}; \boldsymbol{z}^n)\big] \\
\leq&\mathbb{E}_{\boldsymbol{z}^n, \bar{\boldsymbol{z}}^n} \sup_{\boldsymbol{M}\in\mathcal{H}} [\bar{R}_n(\boldsymbol{M}; \bar{\boldsymbol{z}}^n) - \bar{R}_n(\boldsymbol{M}; \boldsymbol{z}^n)].
\end{aligned} \tag{4.22}$$

Finally, we symmetrise again by introducing the Rademacher variables.

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{z}^n, \bar{\boldsymbol{z}}^n} \sup_{\boldsymbol{M}\in\mathcal{H}} [\bar{R}_n(\boldsymbol{M}; \bar{\boldsymbol{z}}^n) - \bar{R}_n(\boldsymbol{M}; \boldsymbol{z}^n)] \\
=&\mathbb{E}_{\boldsymbol{z}^n, \bar{\boldsymbol{z}}^n} \frac{1}{\lfloor\frac{n}{3}\rfloor} \sup_{\boldsymbol{M}\in\mathcal{H}} \sum_{i=1}^{\lfloor\frac{n}{3}\rfloor} [\tilde{\ell}_{\boldsymbol{M}}(\bar{\boldsymbol{z}}_i, \bar{\boldsymbol{z}}_{\lfloor\frac{n}{3}\rfloor+i}, \bar{\boldsymbol{z}}_{\lfloor\frac{n}{3}\rfloor+i}) - \tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_{\lfloor\frac{n}{3}\rfloor+i}, \boldsymbol{z}_{\lfloor\frac{n}{3}\rfloor+i})] \\
=&\mathbb{E}_{\boldsymbol{z}^n, \bar{\boldsymbol{z}}^n, \boldsymbol{\sigma}} \frac{1}{\lfloor\frac{n}{3}\rfloor} \sup_{\boldsymbol{M}\in\mathcal{H}} \sum_{i=1}^{\lfloor\frac{n}{3}\rfloor} \sigma_i [\tilde{\ell}_{\boldsymbol{M}}(\bar{\boldsymbol{z}}_i, \bar{\boldsymbol{z}}_{\lfloor\frac{n}{3}\rfloor+i}, \bar{\boldsymbol{z}}_{\lfloor\frac{n}{3}\rfloor+i}) - \tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_{\lfloor\frac{n}{3}\rfloor+i}, \boldsymbol{z}_{\lfloor\frac{n}{3}\rfloor+i})] \\
\leq&\mathbb{E}_{\boldsymbol{z}^n, \bar{\boldsymbol{z}}^n, \boldsymbol{\sigma}} \frac{1}{\lfloor\frac{n}{3}\rfloor} \sup_{\boldsymbol{M}\in\mathcal{H}} \sum_{i=1}^{\lfloor\frac{n}{3}\rfloor} \sigma_i \tilde{\ell}_{\boldsymbol{M}}(\bar{\boldsymbol{z}}_i, \bar{\boldsymbol{z}}_{\lfloor\frac{n}{3}\rfloor+i}, \bar{\boldsymbol{z}}_{\lfloor\frac{n}{3}\rfloor+i}) \\
&- \mathbb{E}_{\boldsymbol{z}^n, \bar{\boldsymbol{z}}^n, \boldsymbol{\sigma}} \frac{1}{\lfloor\frac{n}{3}\rfloor} \sup_{\boldsymbol{M}\in\mathcal{H}} \sum_{i=1}^{\lfloor\frac{n}{3}\rfloor} \sigma_i \tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_{\lfloor\frac{n}{3}\rfloor+i}, \boldsymbol{z}_{\lfloor\frac{n}{3}\rfloor+i}) \\
=&2\mathbb{E}_{\boldsymbol{z}^n, \boldsymbol{\sigma}} \frac{1}{\lfloor\frac{n}{3}\rfloor} \sup_{\boldsymbol{M}\in\mathcal{H}} \sum_{i=1}^{\lfloor\frac{n}{3}\rfloor} \sigma_i \tilde{\ell}_{\boldsymbol{M}}(\boldsymbol{z}_i, \boldsymbol{z}_{\lfloor\frac{n}{3}\rfloor+i}, \boldsymbol{z}_{\lfloor\frac{n}{3}\rfloor+i})
\end{aligned} \tag{4.23}$$

Step 3: Substituting the result of Lemma 2 into the inequality (4.23) and combining with inequalities (4.20),(4.21),(4.22) prove the theorem. □

4.6.2 Implementation details and additional experimental results

This section includes experimental settings of MLadv and comparison methods, quantitative results of experiments with discretised features, and convergence curves of MLadv.

4.6.2.1 Experimental settings of MLadv and comparison methods

MLadv M is initialised as the identity matrix. The learning rate decays during training according to the exponential function $\exp(-0.99(1+0.01t))$. The training stops if the relative change in the objective function is smaller than the threshold of 10^{-7} or reaches the maximum number of iterations of 5000.

Comparison Methods The comparison methods are implemented by using the official codes provided by the authors. Trade-off parameters are selected based on the validation performance. For LMNN, μ is chosen from $\{0.1, 0.2, \ldots, 0.9\}$. For DRIFT and AML, a grid search is performed with the grid suggested by the authors. For Ord-LMNN, the uniform prior is tested in the experiment; λ is chosen from $\{0.4, 1, \ldots, 4\}$ and τ is chosen from $\{0.5, 1, \ldots, 3.5\}$. All other parameters are set as default.

4.6.2.2 Quantitative Results of Experiments with discretised Features

Tables 4.3 and 4.4 are supplements to Figure 4.2, which list the mean value and standard deviation of classification accuracy on the discretised dataset Magic.

4.6.2.3 Convergence curve

As shown in Figure 4.4, MLadv converges before reaching the maximum iteration number.

Table 4.3: Effect of training sample size on the classification accuracy (mean value ±standard deviation) of LMNN and MLadv.

Sample size	LMNN-int	MLadv-int	LMNN-bin	MLadv-bin
5	60.97±9.91	68.96±5.32	59.56±6.52	65.29±5.08
20	72.67±4.11	74.30±3.08	71.12±4.28	75.32±3.03
35	72.14±2.26	72.98±2.18	71.93±2.07	75.62±2.51
50	73.88±1.80	74.77±2.16	73.28±2.07	76.42±1.86
65	73.83±2.21	74.72±1.91	73.75±1.94	76.54±1.62
80	74.37±1.72	75.17±1.50	74.50±1.87	76.53±1.76
95	74.52±1.27	75.20±1.22	74.97±1.40	77.05±1.64

Table 4.4: Effect of ambiguity level on the classification accuracy of LMNN and MLadv.

Ambiguity level	LMNN-int	MLadv-int	LMNN-bin	MLadv-bin
0%	74.37±1.72	75.20±1.22	74.50±1.87	77.05±1.64
10%	73.22±1.59	74.36±1.34	73.96±1.58	76.16±1.35
20%	73.19±1.65	74.08±1.78	73.69±2.10	76.00±1.40
30%	71.81±2.07	73.09±1.85	72.56±2.37	75.51±1.70
40%	70.21±2.63	71.69±2.34	71.36±2.63	74.29±2.54
50%	69.33±3.06	71.20±2.53	69.77±2.84	73.20±2.53

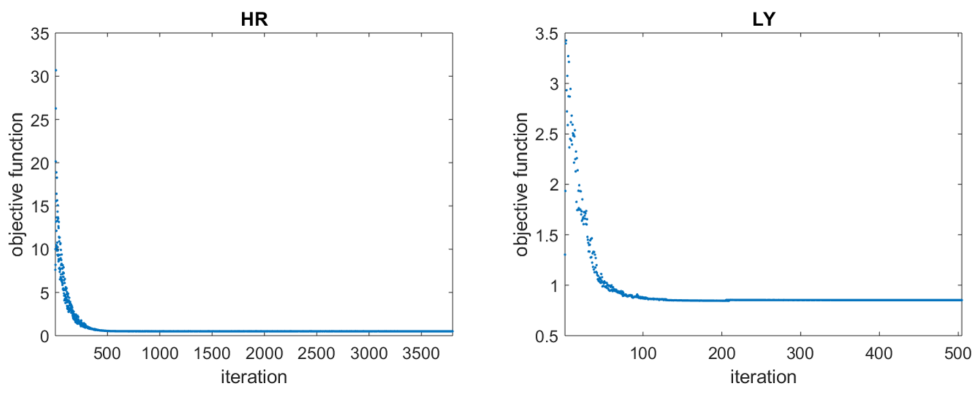


Figure 4.4: Convergence curves of MLadv on datasets HR and LY.

Chapter 5

Metric Learning for Probabilistic Labels

5.1 Introduction

Metric learning [3; 6] is shown to be effective in improving the classification performance of distance-based classifiers, such as k-nearest neighbour (NN). It aims to learn a distance measure such that semantically similar instances are close under the new measure and dissimilar instances are far apart. In most supervised metric learning algorithms, the metric, most commonly the generalised Mahalanobis distance, is learned by formulating an optimisation problem [1]. The class information of training instances is used to construct pairwise or triplet constraints, which are then encoded into the loss function and guide the learning process; instances of the same class are supposed to be similar and instances of different classes should be dissimilar.

The current formulation builds on the assumption that the class label is deterministic, meaning that each instance belongs to one of the classes with complete certainty. However, in some practical applications, it is more natural that the instance belongs to a class with some probability and thus the labels are probabilistic. For instance, a physician may encounter uncertainty when diagnosing a patient as having a disease and hence could only provide a probability to indicate the confidence in the decision. Another application is crowdsourcing. Multiple annotators are asked to classify the same instance and they may not agree on the class label. The proportion of annotators who assign the same label can be viewed as the probability that the instance belongs to this class.

The key challenge of learning the metric from probabilistic labels is how to construct informative constraints. A brute-force approach is to convert probabilistic labels into deterministic labels, such as by selecting the class whose probability exceeds a pre-defined threshold in binary classification tasks or the one with the highest probability in multi-class classification tasks, and then proceed as usual. However, this results in a loss of information about uncertainty and raises a new issue of choosing an appropriate threshold [95]. [96] proposes a solution to construct triplet constraints from probabilistic labels for binary classification problems. It ranks all instances according to their probabilities of belonging to the positive class and, based on the ranks, four sets of constraints are derived. For example, instances with closer ranks are considered to be more similar. While it takes account of probabilistic labels, the method is limited to binary classification. A simple transformation of a multi-class problem into a binary problem via the one-vs-all scheme again discards rich information about individual classes.

In this chapter, we propose a new metric learning algorithm for probabilistic labels based on neighbourhood components analysis (NCA) [11]. NCA is a classical method which learns a linear transformation matrix to maximise the expected accuracy of the stochastic nearest neighbour classifier. Benefiting from its probabilistic formulation, we can sidestep the issue of constructing pairwise or triplet constraints and learn the metric directly from probabilistic labels. More specifically, we firstly revise the existing way of estimating class membership probabilities by replacing deterministic labels with probabilistic labels. Secondly, to fully exploit the probability information on all classes and ensure training stability, we propose to learn the transformation matrix by minimising the Jensen-Shannon divergence (JSD) [97] between the observed probability distribution and the predicted probability distribution.

In addition to the above methodological contribution, we propose new evaluation criteria to assess the effectiveness of metric learning algorithms in the setting of probabilistic labels. The conventional performance measure is the classification accuracy; we suggest to also measure the distance between the observed probability distribution and the predicted probability distribution. The reason is that the predicted probability would be valuable to inform real-world decision making, such as in the medical diagnosis application. Therefore, a good prediction should not only be correct but also associated with a reasonable probability of being correct. To this end, three statistical distances are considered. We test the proposed algorithm on 7 real datasets with simulated and real probabilistic labels, and the method achieves good performance under both accuracy-based and distance-based measures.

The rest of this chapter is organised as follows. Section 5.2 provides preliminaries and reviews NCA. Section 5.3 proposes the new metric learning algorithm. Section 5.4 presents the empirical study and introduces the new evaluation criteria. Section 5.5 draws a conclusion and discusses future work.

5.2 Neighbourhood components analysis (NCA)

In a conventional supervised setting, i.e. labels are deterministic, we are given a set of n training examples $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, where instances $\boldsymbol{x}_i \in \mathbb{R}^p$ and labels $y_i \in \{1, \dots, C\}$. The generalised Mahalanobis distance is represented by a $p \times p$ real-valued positive semi-definite matrix $\boldsymbol{M}$ and, as $\boldsymbol{M}$ can be decomposed into $\boldsymbol{L}^T\boldsymbol{L}$, it is equivalent to computing the Euclidean distance in the linearly transformed space after mapping the data by $\boldsymbol{L}$:

$$\begin{aligned} d_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{M} (\boldsymbol{x}_i - \boldsymbol{x}_j)} \\ &= \|\boldsymbol{L}\boldsymbol{x}_i - \boldsymbol{L}\boldsymbol{x}_j\|_2. \end{aligned}$$

The goal of neighbourhood components analysis (NCA) is to learn the linear transformation matrix $\boldsymbol{L}$ such that the test accuracy of NN classifier is maximised. As the test data is unavailable, the method instead maximises the leave-one-out (LOO) classification accuracy on the training data. Moreover, to avoid the discontinuity of the LOO accuracy as a function of $\boldsymbol{L}$, a stochastic selection rule is used to approximate the selection of NN, leading to a probabilistic formulation of metric learning.

The stochastic selection rule defines the probability that $\boldsymbol{x}_i$ selects $\boldsymbol{x}_j$ as its neighbour by applying a softmax function over the Mahalanobis distance:

$$p_{ij} = \frac{\exp(-\|\boldsymbol{L}\boldsymbol{x}_i - \boldsymbol{L}\boldsymbol{x}_j\|_2^2)}{\sum_{k \neq i} \exp(-\|\boldsymbol{L}\boldsymbol{x}_i - \boldsymbol{L}\boldsymbol{x}_k\|_2^2)}, \quad p_{ii} = 0. \tag{5.1}$$

Let z_i denote the predicted label of $\boldsymbol{x}_i$. Then $\boldsymbol{x}_i$ will be predicted as class c with the following probability:

$$P(z_i = c) = \sum_{j=1}^{n} p_{ij} \mathbb{1}[y_j = c], \tag{5.2}$$

where $\mathbb{1}[\cdot]$ denotes the indicator function.

To learn $\boldsymbol{L}$, two objective functions have been proposed. The first one is to maximise the expected number of instances that are correctly classified:

$$\max_{\boldsymbol{L}} \sum_{i=1}^{n} \sum_{j=1}^{n} p_{ij} \mathbb{1}[y_j = y_i].$$

The second objective is to maximise the log-likelihood of all instances selecting same-class points as their neighbours:

$$\max_{\boldsymbol{L}} \sum_{i=1}^{n} \log \Big(\sum_{j=1}^{n} p_{ij} \mathbb{1}[y_j = y_i] \Big). \tag{5.3}$$

Eq. 5.3 can be alternatively interpreted as minimising the sum of Kullback–Leibler (KL) divergences from the predicted probability distribution $P(Z_i) = [P(z_i = 1), \ldots, P(z_i = C)]$ to the observed probability distribution $P(Y_i)$ over all instances.

5.3 Probabilistic neighbourhood components analysis (PNCA)

In this section, we propose a new approach to learning the metric from probabilistic labels; the method is termed probabilistic neighbourhood components analysis (PNCA). The formulation of PNCA is first introduced, followed by its objective function. The optimisation algorithm is finally presented.

5.3.1 Probabilistic formulation of NCA

Compared with deterministic labels where each instance $\boldsymbol{x}_i$ is associated with a label y_i, in the case of probabilistic labels, $\boldsymbol{x}_i$ is associated with a vector of label probabilities $P(Y_i) = [P(y_i = 1), \ldots, P(y_i = C)]$. Learning the metric based on NCA is particularly appropriate for probabilistic labels since it outputs a prediction with probability. Nevertheless, the method requires two revisions on the estimation of predicted probabilities and the objective function for learning $\boldsymbol{L}$.

To estimate the predicted probability that $\boldsymbol{x}_i$ belongs to class c, we replace the deterministic labels in Eq. 5.2 by probabilistic labels as follows:

$$P(z_i = c) = \sum_{j=1}^{n} p_{ij} P(y_j = c), \tag{5.4}$$

where p_{ij} is given by Eq. 5.1 and is a function of the transformation matrix $\boldsymbol{L}$. As presented in the following section, $\boldsymbol{L}$ will be learned by optimising the *divergence loss*.

5.3.2 Divergence loss

To make full use of the probability information on labels, we propose to minimise the Jensen-Shannon divergence (JSD) [97] between the observed probability distribution $P(Y)$ and the predicted probability distribution $P(Z)$: let $P(M_i) = \frac{1}{2}(P(Y_i) + P(Z_i))$, then the divergence loss is defined as

$$\mathcal{L}_{\text{div}} = \frac{1}{n} \sum_{i=1}^{n} D_{\text{JSD}}(P(Y_i) \| P(Z_i))$$

$$=\frac{1}{n}\sum_{i=1}^{n}\Big[\frac{1}{2}D_{\text{KL}}(P(Y_i)\|P(M_i))+\frac{1}{2}D_{\text{KL}}(P(Z_i)\|P(M_i))\Big] \tag{5.5}$$
$$=\frac{1}{2n}\sum_{i=1}^{n}\sum_{c=1}^{C}\left(P(y_i=c)\log\frac{P(y_i=c)}{P(m_i=c)}+P(z_i=c)\log\frac{P(z_i=c)}{P(m_i=c)}\right).$$

JSD is an information-theoretic divergence measure between two probability distributions; a small value of JSD indicates that the distributions are more similar. Therefore, the preceding objective function encourages learning a transformation matrix such that the predicted probability, derived from neighbours in the transformed space, is similar to the observed probability. Moreover, compared with the KL divergence used in NCA, optimising the JSD leads to a more stable training process. The reason is that the JSD is well defined over the probability range of $[0, 1]$ and thus will be less affected by highly confident instances whose label probabilities approach to 1 for one class and 0 for all other classes. This will be further explained after presenting the gradient in Section 5.3.3.

5.3.3 Optimisation problem of PNCA

The optimisation problem of PNCA is formulated by combining the divergence loss with a regularisation term to constrain the complexity of $\boldsymbol{L}$:

$$\min_{\boldsymbol{L}} \mathcal{L}_{\text{div}} + \lambda\|\boldsymbol{L}\|_F^2, \tag{5.6}$$

where $\|\boldsymbol{L}\|_F$ denotes the Frobenius norm of $\boldsymbol{L}$; λ is a hyperparameter which controls the influence of the regularisation.

The optimisation problem of Eq. 5.6 is non-convex and is optimised via the gradient descent algorithm. The gradient of $\mathcal{L}_{\text{div}}$ is given as follows:

$$\frac{\partial\mathcal{L}_{\text{div}}}{\partial\boldsymbol{L}}=\boldsymbol{L}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\sum_{c=1}^{C}\log\Big(\frac{1}{2}+\frac{1}{2}\frac{P(y_i=c)}{P(z_i=c)}\Big)\big(P(y_j=c)-P(z_i=c)\big)\right)p_{ij}\boldsymbol{X}_{ij}, \tag{5.7}$$

where $\boldsymbol{X}_{ij}=(\boldsymbol{x}_i-\boldsymbol{x}_j)(\boldsymbol{x}_i-\boldsymbol{x}_j)^T$. As mentioned before, minimising the JSD leads to more stable training than minimising the KL divergence. This is due to the influence of $P(z_i = c)$ on the gradient. The objective function built on the KL divergence and the associated gradient are given in Eq. 5.8. Compared with it, minimising the JSD introduces an additional logarithm operator to $1/P(z_i = c)$ and hence avoids the gradient to be predominantly determined by a few instances whose $P(z_i = c)$ is close to zero.

$$\begin{aligned} \mathcal{L} &= \frac{1}{n}\sum_{i=1}^{n} D_{\mathrm{KL}}(P(Y_i)\|P(Z_i)); \\ \frac{\partial \mathcal{L}}{\partial \boldsymbol{L}} &= 2\boldsymbol{L}\sum_{i=1}^{n}\sum_{j=1}^{n}\Big(\sum_{c=1}^{C}\frac{P(y_i = c)}{P(z_i = c)}P(y_j = c) - 1\Big)p_{ij}\boldsymbol{X}_{ij}. \end{aligned} \tag{5.8}$$

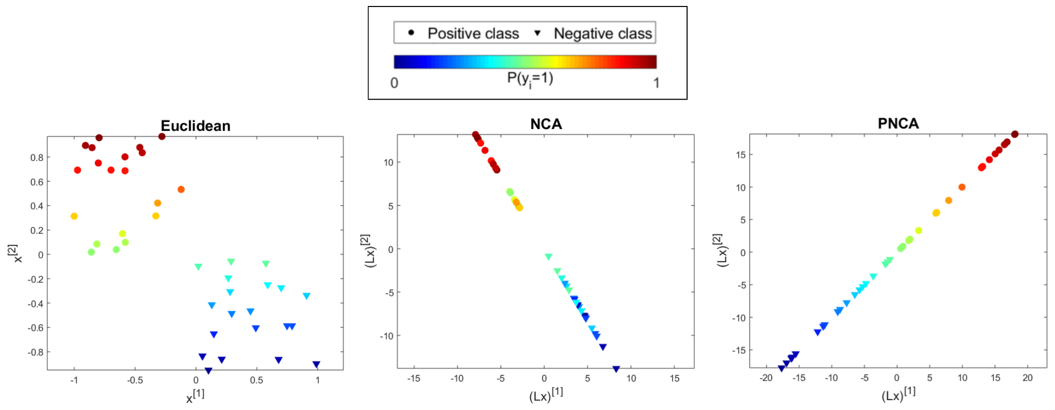
5.4 Experiments

In this section, we firstly explore how probabilistic labels affect metric learning on two toy examples. Secondly, we evaluate the performance of PNCA on 7 datasets with synthetic and real probabilistic labels. Thirdly, we propose new evaluation measures to assess the reliability of predicted probabilities.

5.4.1 Toy examples

The goal of this experiment is to understand the benefit and limitation of considering label probabilities when learning the metric. To this end, we simulate two binary classification datasets, synthesise probabilistic labels, visualise the metric learned from NCA and PNCA, and compare the classification accuracy. NCA is optimised by using the KL divergence as the objective function (i.e. Eq. 5.3) and, for a fair comparison, PNCA is optimised with only the divergence loss (i.e. $\lambda = 0$).

The first dataset shows the benefit of utilising label probabilities. The dataset includes 40 instances of two dimensions. 20 instances are drawn uniformly from $[-1, 0]$ in the first dimension and $[0, 1]$ in the second dimension, and the remaining 20 instances are drawn uniformly from $[0, 1]$ and $[0, -1]$ in the two dimensions respectively. Deterministic labels are generated according to the following classification rule – an instance is of the positive class if its second feature has a positive value. Probabilistic labels are also generated based on the second feature alone – the probability of belonging to the positive class equals to the cumulative probability of the second feature; that is, $P(y_i = 1) = P(U \leq \boldsymbol{x}_i^{[2]})$, where $U \sim \text{unif}(-1, 1)$ and $\boldsymbol{x}_i^{[2]}$ denotes the value of the second feature. Figure 5.1a shows the dataset; deterministic labels are indicated by shapes and probabilistic labels are indicated by the colour scale.

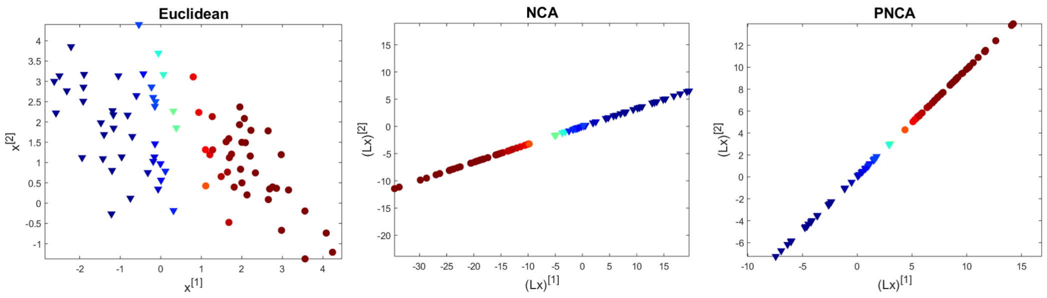


(a) $\boldsymbol{M} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ (b) $\boldsymbol{M} = \begin{bmatrix} 74.0 & -72.9 \\ -72.9 & 71.8 \end{bmatrix}$ (c) $\boldsymbol{M} = \begin{bmatrix} 0.1 & -9.1 \\ -9.1 & 693.9 \end{bmatrix}$

Figure 5.1: Illustration of the benefit of probabilistic labels. Instances are shown in (a) the original space, (b) the transformed space with $\boldsymbol{L}$ learned from NCA, (c) the transformed space with $\boldsymbol{L}$ learned from PNCA. The learned metric is given in the subcaption.

Figures 5.1b and 5.1c show instances in the transformed space with the linear transformation matrix learned from NCA and PNCA, respectively; the corresponding Mahalanobis distances are given in the caption. The objective of NCA encourages all instances are correctly classified by using nearby samples. Consequently, the learned metric reduces the intra-class dispersion and retains the inter-class separation. The objective of PNCA encourages the predicted probabilities of all instances to be similar to the observed probabilities. The learned metric shrinks the distance along the first dimension (indicated by a small value of $\boldsymbol{M}_{1,1}$) and expands the distance along the second dimension (indicated by a large value of $\boldsymbol{M}_{2,2}$). Consequently, instances with similar observed label probabilities locate close to each other. On a separate test data which includes 2000 instances generated from the same distribution, NCA obtains a classification accuracy of 0.981, and PNCA obtains a higher accuracy of 0.996. This example suggests that the proposed method is able to exploit the extra supervision provided by probabilistic than deterministic labels and, by minimising the divergence between distributions, it identifies that the first feature is less useful than the second feature for classification.

The second dataset explains the limitation of PNCA. The dataset includes 80 instances generated from two-dimensional Gaussian distributions. The mean vectors of the positive class and the negative class are $[2, 1]^T$ and $[-1, 2]^T$, respectively; the covariance matrices are set as $\left[\begin{smallmatrix} 1 & -0.5 \\ -0.5 & 1 \end{smallmatrix}\right]$ for both classes. Deterministic labels are generated according to the Bayes rule, and probabilistic labels are generated according to the posterior distribution. Figure 5.2a shows the dataset.



(a) $M = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ (b) $M = \begin{bmatrix} 79.2 & 9.2 \\ 9.2 & 1.1 \end{bmatrix}$ (c) $M = \begin{bmatrix} 25.5 & 6.1 \\ 6.1 & 1.4 \end{bmatrix}$

Figure 5.2: Illustration of the limitation of PNCA.

Figures 5.2b and 5.2c show instances in the transformed space after applying NCA and PNCA, respectively. Compared with NCA which focuses on separating instances of different classes by a large margin, PNCA focuses more on grouping instances with similar probabilities. Consequently, the generalisation performance of PNCA is inferior to NCA. The classification accuracy of PNCA and NCA are 0.830 and 0.861, respectively.

5.4.2 Real datasets

To fully understand the effectiveness of PNCA, we test the method on three types of datasets. The first two data types, regression datasets and multi-class classification datasets, are used to synthesise probabilistic labels for binary classification tasks and multi-class classification tasks, respectively. The third data type, a crowdsourcing dataset, provides real probabilistic labels.

Since existing metric learning methods are built on deterministic labels and cannot directly work with probabilistic labels, we only compare PNCA with the

Table 5.1: Characteristics of the datasets.

Dataset	Type	#Classes	#Features (#Reduced dim.)	#Instances	Class distribution
Concrete	regression	2	8	1030	319/711
Energy	regression	2	8	768	330/438
Housing	regression	2	13	506	110/396
Vehicle	multiclass	4	18	846	
Accent	multiclass	5	12	164	
Segment	multiclass	7	19	2310	
Music	crowdsourcing	10	124 (20)	700	

The class distribution is listed for imbalanced datasets. These datasets are evaluated based on F1 score. All other datasets are evaluated based on classification accuracy.

baseline method NCA and the metric learning method for probabilistic labels InML [96].

The overall setup of experiments and parameter tuning of each method are first explained, followed by data description and experimental results for three data types. Table 5.1 lists main characteristics of the 7 datasets studied in this experiment.

5.4.2.1 Experimental settings

All datasets are pre-processed with mean-centring and standardisation. We use 70%-30% training-test partitions and report the average result over 20 rounds of random split. 1NN is used as the classifier.

For a fair comparison, we include the Frobenius norm regularisation in NCA. Eq. 5.3 is chosen as its objective function. InML is designed for binary classification problems. To extend it to the multi-class classification setting, we adopt the one-vs-all strategy and construct triplet constraints for each class in turn. Hyperparameters of all methods are tuned via 5-fold cross-validation on the training set. For NCA, the weight of the regularisation term is selected from the range of $\{10^{-3}, 10^{-2}, \ldots, 10^{2}\}$. For InML, the regulariser weight α is selected from $\{10^{-4}, 10^{-3}, \ldots, 10^{1}\}$. For the proposed PNCA, λ is selected from $\{10^{-7}, 10^{-6}, \ldots, 10^{-2}\}$. NCA is implemented by using drToolbox [98], and InML is implemented by following the procedure described in [96].

5.4.2.2 Regression datasets

We simulate binary classification tasks from three regression datasets [57], namely Concrete, Energy and Housing. The Energy dataset includes two response variables and the response ‘heating load’ is used in this study. We apply min-max normalisation to the real-valued response and use the normalised value as probabilistic labels. To generate deterministic labels, we assign instances to the positive class if the normalised value is larger than 0.5 and otherwise to the negative class. As the dataset is imbalanced, F1 score is used as the evaluation measure.

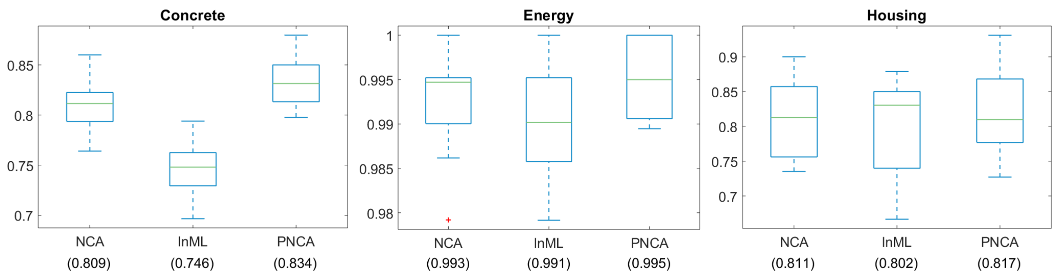


Figure 5.3: Boxplots of F1 scores of 1NN on regression datasets. Mean values of F1 scores are given in the bracket.

Figure 5.3 gives the boxplots and mean values of F1 scores for NCA, InML and PNCA. Compared with NCA, PNCA achieves higher mean value on all datasets. Except on the Housing dataset where it has a slightly lower median, PNCA obtains higher median, lower quartile and upper quartile than NCA. Same finding applies when comparing PNCA with InML.

5.4.2.3 Multi-class classification datasets

For multi-class classification, we adopt datasets Vehicle (4 classes), Accent (5 classes), and Segment (7 classes) [57]. For Accent, we have removed the response ‘US’ since it causes severe class imbalance. Based on the original features and class information, a multinominal logistic regression is applied. The estimated probabilities are used as probabilistic labels, and the class with the maximum probability defines the deterministic labels. As the logistic regression model cannot perfectly fit the data, there is some discrepancy between the original class labels and the derived deterministic labels. The transformation matrix is trained on the derived probabilistic/deterministic labels and tested on the original labels. The classification accuracy

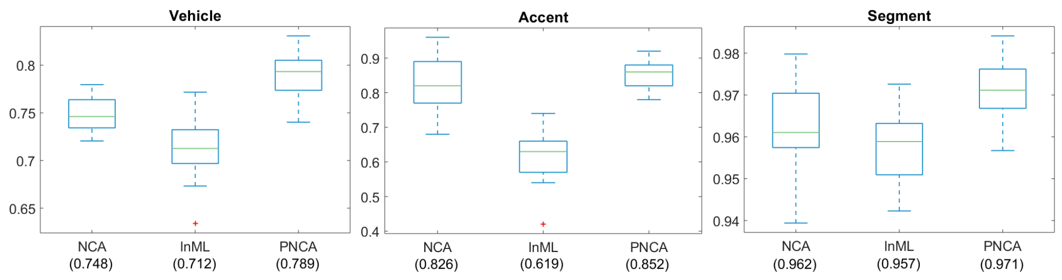


Figure 5.4: Boxplots and mean values of accuracy of 1NN on multi-class classification datasets.

is used as the evaluation measure.

Figure 5.4 shows the experimental results. On all three datasets, PNCA achieves the highest mean and median accuracy. The method is also reliable. Its spread, as indicated by the interquartile range (IQR), is competitive with NCA and InML on the Vehicle dataset and is smaller on the other two datasets.

5.4.2.4 Real-world crowdsourcing dataset

The final experiment adopts a crowdsourcing dataset with real probabilistic labels. The music genre dataset consists of 700 songs, carefully chosen by experts as representative of 10 music genres [99]. It has been published on Amazon Mechanical Turk for annotation, and each song is categorised as one of the genres by multiple annotators. The proportion of annotators who classify an instance as a particular music genre gives the probability that the instance belongs to that class. The class with the highest probability gives the deterministic label. As discussed in [99], only a couple of annotators provide high quality answers. This is also found after our conversion, where the annotation labels agree with the expert labels on only 71% of instances. To avoid overfitting, we add an additional pre-processing step by reducing the feature dimensionality from 124 to 20 via PCA; the leading PCs account for 90% of total variance.

Figure 5.5 shows the classification accuracy of 1NN. The proposed method again shows its advantage over NCA with higher accuracy and smaller spread. InML is originally proposed for binary classification, and when the one-vs-all scheme is applied, triplet constraints constructed from the positive class and the merged minority class may not provide useful supervision, thereby leading to an

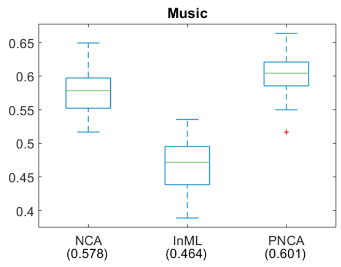


Figure 5.5: Boxplots and mean values of accuracy of 1NN on the crowdsourcing dataset.

unsatisfactory classification performance which is worse than a random guess.

5.4.3 Evaluation criteria for probabilistic labels

As the input supervision is in the form of probability and reflects uncertainty of classifying an instance, associating the predicted outcome with a probability may aid in decision making of downstream applications. Therefore, we propose additional evaluation criteria to measure the distance between the observed probability and the predicted probability. For all methods, the predicted probability is calculated according to Eq. 5.4 with p_{ij} calculated with respect to the learned transformation matrix. We consider three metrics that measures the distance between probability distributions, namely the JS distance (denoted as D_{JS}) [100], the Hellinger distance (D_{H}) [101], and the total variation distance (D_{TV}) [102].

The definitions of the studied distances are given as follows. Let P and Q denote two discrete probability distributions over $\mathcal{X}$, and $P(A) = \sum_{x \in A} P(x)$ for $A \subseteq \mathcal{X}$. Then,

$$\begin{aligned}
D_{\text{JS}}(P,Q) &= \sqrt{D_{\text{JSD}}(P\|Q)} \\
D_{\text{H}}(P,Q) &= \sqrt{\sum\nolimits_x \left(\sqrt{P(x)} - \sqrt{Q(x)}\right)^2} = \left\|\sqrt{P} - \sqrt{Q}\right\|_2 \\
D_{\text{TV}}(P,Q) &= \sup\nolimits_{A \subseteq \mathcal{X}} |P(A) - Q(A)| = \frac{1}{2}\|P - Q\|_1.
\end{aligned}$$

All three distances are proper metrics, i.e. satisfying the axioms of triangle inequality, non-negativity, symmetry and identity of indiscernibles, and are bounded. Moreover, they are well defined for all possible values of $P(x)$ and $Q(x)$, which is particularly important for probabilistic labels as it is very likely that some observed label probabilities are close to 0. Among the three distances, the JS distance is most directly linked with our objective function, since it is the square root of JSD. Hellinger and TV distances both belong to the family of f-divergence and they are related as $\frac{(D_{\mathrm{H}})^2}{2} \leq D_{\mathrm{TV}} \leq D_{\mathrm{H}}$ [102].

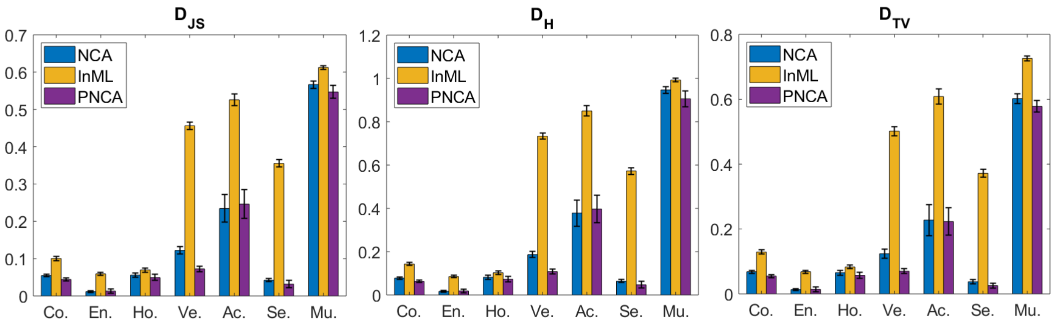


Figure 5.6: Evaluation of metric learning methods based on the Jensen-Shannon distance (D_{JS}), the Hellinger distance (D_{H}), and the total variation distance (D_{TV}). Dataset names are abbreviated to the first two letters.

Figure 5.6 shows the performance of different methods evaluated by using distance measures; a smaller distance indicates that the predicted probability distribution is more consistent with the test probability distribution. In terms of the JS distance, PNCA outperforms NCA on 5 out of 7 datasets. InML is founded on LMNN [10] rather than a probabilistic framework. Therefore, compared with NCA and PNCA, its predicted probability is more different from the observed probability. PNCA retains its benefit when measured by the Hellinger distance and the TV distance; it obtains to the smallest distance on 6 and 7 datasets, respectively.

To further illustrate the benefit of utilising the label probabilities to metric learning and value of the predicted probability, we provide a visualisation of the Vehicle dataset in Figure 5.7. t-SNE [103] is used to embed the linearly transformed test data into two dimensions. Class categories are indicated by different colours and shapes; predicted probabilities are indicated by colour transparency, where a lighter colour corresponds to a smaller probability. We see that, by using PNCA, instances of Class 3 and 4 form their own groups and stay away from instances of other classes. Instances of Class 1 and 2 could not be well separated. In this case, the predicted probability is smaller when an instance is surrounded by more heterogeneous neighbours. This pattern suggests that PNCA can produce a

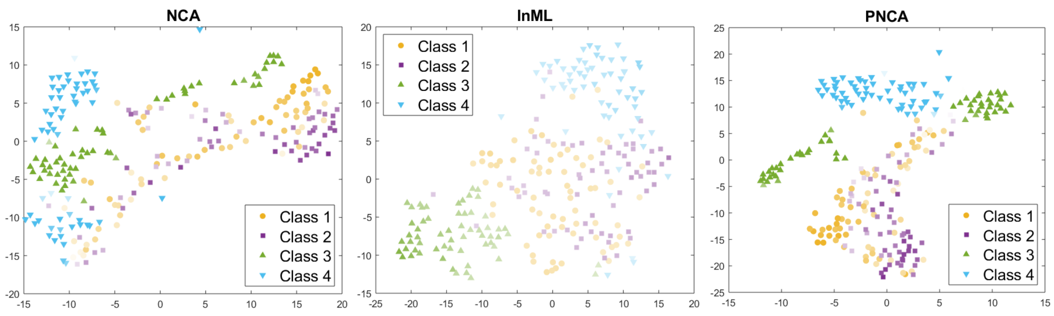


Figure 5.7: t-SNE of the Vehicle dataset. Class categories are represented by different colours; lighter colour indicates a smaller probability that an instance belongs to the corresponding class.

reliable predicted probability, which may serve as an indicator of the confidence in the predicted outcome.

5.5 Conclusions and future work

In this chapter, we proposed PNCA as a simple yet effective way of learning the distance metric from probabilistic labels. The algorithm fully exploits the label information in the estimation of predicted probabilities and in the optimisation of the loss function. Experiments on 7 datasets demonstrate the effectiveness of PNCA in terms of both accuracy and three newly introduced distance criteria.

This work suggests the potential of adopting a probabilistic framework for metric learning from probabilistic labels. In the future, we intend to further improve the framework, such as correcting the label noise as in [78] and generalising it for stochastic selection of k-neighbourhoods with $k > 1$ as in [104].

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this thesis, three distance metric learning algorithms have been proposed to improve the robustness of existing methods against instance perturbation and extend their applicability for categorical features and probabilistic labels.

Chapter 3 addresses the limitation of large margin-based metric learning methods in withstanding a small perturbation of test instance. A loss function that directly enlarges the adversarial margin has been proposed. It prevents the perturbed instance from changing the label of its nearest neighbours and thus guarantees that the prediction of kNN classifier remains the same. Extensive experiments on benchmark datasets show that the proposed loss can be easily leveraged with existing triplet-based methods and validate its effectiveness in improving the generalisation ability and robustness of the learned distance metric.

Chapter 4 presents a new metric learning algorithm that specifically targets at categorical features. The method alleviates the issue of feature ambiguity by learning the metric through adversarial training. It is particularly beneficial for small-sized datasets as more information is utilised when learning the metric. The generalisation ability of the proposed method has been guaranteed theoretically and verified empirically on ordinal and mixed ordinal-and-nominal datasets with high feature ambiguity and small sample size.

Chapter 5 presents a new algorithm which allows the metric to be learned from probabilistic labels. By adapting NCA and minimising the Jensen-Shannon divergence, the probability information on all classes is fully exploited to supervise metric learning. Considering the practical value of associating the predicted outcome with a reasonable probability, evaluation criteria based on statistical distances have been suggested to assess the efficacy of different methods.

6.2 Future work

6.2.1 Certified robustness for nonlinear metric learning methods

While enlarging the adversarial margin is shown to be effective in improving certified robustness for deep learning models, the approach proposed in Chapter 3, though sharing the same idea, cannot certify adversarial robustness. The reason is that, like most metric learning algorithms, our method fixes target neighbours and impostors prior to metric learning for computational efficiency, and thus the support point derived from the fixed set of triplet constraints is not necessarily the nearest adversarial example among all possible triplets. To achieve certified robustness and avoid high computational cost, one possible solution is to incorporate prototype learning [105] into our method. The idea is to learn few prototype examples to represent the training set, and all test instances will be classified according to their distances to these examples. As the number of prototypes is generally very small, the number of triplets is limited and hence the support points can be derived efficiently.

The expressive power of the Mahalanobis distance is limited as it corresponds to learning a global linear transformation. To suit datasets that have multi-modality or nonlinearly separable features, local metric learning methods and kernelised methods have been proposed. It would be interesting to investigate their robustness and extend the idea of adversarial margin to these methods when appropriate.

6.2.2 Metric learning from noisy probabilistic labels and label distributions

In Chapter 5, the metric is learned from probabilistic labels, assuming the label information is correct. On the other hand, the crowdsourcing dataset contains a relatively high level of noise. Therefore, it would be valuable to robustify the model against label noise. One possible solution is to adopt the latent variable approach proposed in [78], which introduces a latent variable to represent the true label and associates the observed label with this latent variable via a transition matrix. The transition matrix is only suitable for deterministic labels and should be replaced by an appropriate non-parametric model to suit probabilistic labels.

A closely related topic to probabilistic labels is label distribution learning [106], where each instance is assigned to a label distribution. Conceptually, the two topics differ in two aspects. Learning from probabilistic labels is a single-label classification task, and the probability reflects the inherent difficulty of classifying an instance. In contrast, learning from label distributions is a multi-label classification task; all labels with non-zero probabilities are correct labels for an instance. The probability reflects the degree to which the label describes an instance. It would be interesting to extend PNCA to this problem setting by taking into account the correlation between labels.

6.2.3 Uncertainty estimation for metric learning

Most metric learning algorithms output a point estimation of the distance metric. On the other hand, it would be desirable if the algorithm could provide an uncertainty estimate to characterise the confidence in this output. This would be particularly valuable when the metric is learned from imperfect data, such as the ambiguous features and probabilistic labels studied in Chapters 4 and 5. One way of estimating the uncertainty is through Bayesian metric learning, which has been discussed in [107]. Based on this, the first future work would be to formulate PNCA in a Bayesian framework. Another way of uncertainty estimation is to adopt bootstrap [108] or ensemble methods [109]. The latter approach has been actively researched in quantifying uncertainty for deep learning models, and it also shows promising performances on detecting out-of-distribution inputs and adversarial examples. Incorporating these ideas into metric learning would be another interesting future research topic.

Appendix A

MSDH: Matched Subspace Detector with Heterogeneous Noise

Appendix B

Data-Augmented Matched Subspace Detector for Hyperspectral Subpixel Target Detection

Bibliography

[1] E. P. Xing, M. I. Jordan, S. J. Russell, and A. Y. Ng, "Distance metric learning with application to clustering with side-information," in *Advances in Neural Information Processing Systems*, pp. 521–528, 2003.

[2] L. Yang and R. Jin, "Distance metric learning: A comprehensive survey," *Michigan State Universiy*, vol. 2, no. 2, p. 4, 2006.

[3] B. Kulis, "Metric learning: A survey," *Foundations and Trends in Machine Learning*, vol. 5, no. 4, pp. 287–364, 2013.

[4] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[5] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *International Conference on Learning Representations*, 2018.

[6] A. Bellet, A. Habrard, and M. Sebban, "Metric learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 9, no. 1, pp. 1–151, 2015.

[7] P. C. Mahalanobis, "On the generalized distance in statistics," National Institute of Science of India, 1936.

[8] R. Short and K. Fukunaga, "The optimal distance measure for nearest neighbor classification," *IEEE Transactions on Information Theory*, vol. 27, no. 5, pp. 622–627, 1981.

[9] D. Li and Y. Tian, "Survey and experimental study on metric learning methods," *Neural Networks*, vol. 105, pp. 447–462, 2018.

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

[11] J. Goldberger, G. E. Hinton, S. T. Roweis, and R. R. Salakhutdinov, "Neighbourhood components analysis," in *Advances in Neural Information Processing Systems*, pp. 513–520, 2005.

[12] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014.

[13] V. Vapnik, *The nature of statistical learning theory*. Springer Science & Business Media, 2013.

[14] O. Bousquet and A. Elisseeff, "Stability and generalization," *Journal of Machine Learning Research*, vol. 2, pp. 499–526, 2002.

[15] H. Xu and S. Mannor, "Robustness and generalization," *Machine Learning*, vol. 86, no. 3, pp. 391–423, 2012.

[16] P. L. Bartlett and S. Mendelson, "Rademacher and gaussian complexities: Risk bounds and structural results," *Journal of Machine Learning Research*, vol. 3, pp. 463–482, 2002.

[17] A. Ben-Tal and A. Nemirovski, "Robust convex optimization," *Mathematics of Operations Research*, vol. 23, no. 4, pp. 769–805, 1998.

[18] E. Wong and Z. Kolter, "Provable defenses against adversarial examples via the convex outer adversarial polytope," in *International Conference on Machine Learning*, pp. 5286–5295, 2018.

[19] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *International Conference on Learning Representations*, 2017.

[20] C. Xiao, J. Y. Zhu, B. Li, W. He, M. Liu, and D. Song, “Spatially transformed adversarial examples,” in *International Conference on Learning Representations*, 2018.

[21] B. McFee and G. R. Lanckriet, “Metric learning to rank,” in *International Conference on Machine Learning*, pp. 775–782, 2010.

[22] J. Yang, D. She, Y.-K. Lai, and M.-H. Yang, “Retrieving and classifying affective images via deep metric learning,” in *AAAI Conference on Artificial Intelligence*, 2018.

[23] M. Koestinger, M. Hirzer, P. Wohlhart, P. M. Roth, and H. Bischof, “Large scale metric learning from equivalence constraints,” in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2288–2295, IEEE, 2012.

[24] J. Hu, J. Lu, and Y.-P. Tan, “Sharable and individual multi-view metric learning,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 9, pp. 2281–2288, 2017.

[25] K. Roth, T. Milbich, S. Sinha, P. Gupta, B. Ommer, and J. P. Cohen, “Revisiting training strategies and generalization performance in deep metric learning,” in *International Conference on Machine Learning*, 2020.

[26] H. Xiong and X.-W. Chen, “Kernel-based distance metric learning for microarray data classification,” *BMC Bioinformatics*, vol. 7, no. 1, pp. 1–11, 2006.

[27] S. Xiang, F. Nie, and C. Zhang, “Learning a Mahalanobis distance metric for data clustering and classification,” *Pattern Recognition*, vol. 41, no. 12, pp. 3600–3612, 2008.

[28] X. Wang, X. Han, W. Huang, D. Dong, and M. R. Scott, “Multi-similarity loss with general pair weighting for deep metric learning,” in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5022–5030, 2019.

[29] M. Dong, Y. Wang, X. Yang, and J.-H. Xue, “Learning local metrics and influential regions for classification,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 6, pp. 1522–1529, 2020.

[30] L. Gautheron, A. Habrard, E. Morvant, and M. Sebban, “Metric learning from imbalanced data with generalization guarantees,” *Pattern Recognition Letters*, vol. 133, pp. 298–304, 2020.

[31] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, “Information-theoretic metric learning,” in *International Conference on Machine Learning*, pp. 209–216, ACM, 2007.

[32] J. T. Kwok and I. W. Tsang, “Learning with idealized kernels,” in *International Conference on Machine Learning*, pp. 400–407, 2003.

[33] K. Song, F. Nie, J. Han, and X. Li, “Parameter free large margin nearest neighbor for distance metric learning,” in *AAAI Conference on Artificial Intelligence*, 2017.

[34] J. Xu, L. Luo, C. Deng, and H. Huang, “Bilevel distance metric learning for robust image recognition,” in *Advances in Neural Information Processing Systems*, pp. 4198–4207, 2018.

[35] H. Liu, Z. Han, Y.-S. Liu, and M. Gu, “Fast low-rank metric learning for large-scale and high-dimensional data,” in *Advances in Neural Information Processing Systems*, pp. 817–827, 2019.

[36] M. T. Law, N. Thome, and M. Cord, “Quadruplet-wise image similarity learning,” in *IEEE International Conference on Computer Vision*, pp. 249–256, 2013.

[37] K. Sohn, “Improved deep metric learning with multi-class n-pair loss objective,” in *Advances in Neural Information Processing Systems*, pp. 1857–1865, 2016.

[38] X. Li, L. Yu, C.-W. Fu, M. Fang, and P.-A. Heng, “Revisiting metric learning for few-shot image classification,” *Neurocomputing*, 2020.

[39] Y. Shi, A. Bellet, and F. Sha, “Sparse compositional metric learning,” in *AAAI Conference on Artificial Intelligence*, 2014.

[40] Z. Huo, F. Nie, and H. Huang, “Robust and effective metric learning using capped trace norm: Metric learning via capped trace norm,” in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1605–1614, ACM, 2016.

[41] A. Bellet and A. Habrard, “Robustness and generalization for metric learning,” *Neurocomputing*, vol. 151, pp. 259–267, 2015.

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

[43] R. Jin, S. Wang, and Y. Zhou, “Regularized distance metric learning: Theory and algorithm,” in *Advances in Neural Information Processing Systems*, pp. 862–870, 2009.

[44] D. Lim, G. Lanckriet, and B. McFee, “Robust structural metric learning,” in *International Conference on Machine Learning*, pp. 615–623, 2013.

[45] H. Wang, F. Nie, and H. Huang, “Robust distance metric learning via simultaneous ℓ_1-norm minimization and maximization,” in *International Conference on Machine Learning*, pp. 1836–1844, 2014.

[46] M. T. Law, N. Thome, and M. Cord, “Fantope regularization in metric learning,” in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1051–1058, 2014.

[47] L. Luo and H. Huang, “Matrix variate Gaussian mixture distribution steered robust metric learning,” in *AAAI Conference on Artificial Intelligence*, 2018.

[48] K. Liu, L. Brand, H. Wang, and F. Nie, "Learning robust distance metric with side information via ratio minimization of orthogonally constrained $\ell_{2,1}$-norm distances," in *International Joint Conference on Artificial Intelligence*, 2019.

[49] H.-J. Ye, D.-C. Zhan, X.-M. Si, and Y. Jiang, "Learning Mahalanobis distance metric: considering instance disturbance helps," in *International Joint Conference on Artificial Intelligence*, pp. 3315–3321, 2017.

[50] Q. Qian, J. Tang, H. Li, S. Zhu, and R. Jin, "Large-scale distance metric learning with uncertainty," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8542–8550, 2018.

[51] S. Chen, C. Gong, J. Yang, X. Li, Y. Wei, and J. Li, "Adversarial metric learning," in *International Joint Conference on Artificial Intelligence*, pp. 2021–2027, 2018.

[52] Y. Duan, W. Zheng, X. Lin, J. Lu, and J. Zhou, "Deep adversarial metric learning," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2780–2789, 2018.

[53] S. An, M. Hayat, S. H. Khan, M. Bennamoun, F. Boussaid, and F. Sohel, "Contractive rectifier networks for nonlinear maximum margin classification," in *IEEE International Conference on Computer Vision*, pp. 2515–2523, 2015.

[54] G. Elsayed, D. Krishnan, H. Mobahi, K. Regan, and S. Bengio, "Large margin deep networks for classification," in *Advances in Neural Information Processing Systems*, pp. 842–852, 2018.

[55] Z. Yan, Y. Guo, and C. Zhang, "Adversarial margin maximization networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[56] G. W. Ding, Y. Sharma, K. Y. C. Lui, and R. Huang, "MMA training: Direct

input space margin maximization through adversarial training," in *International Conference on Learning Representations*, 2020.

[57] D. Dua and C. Graff, "UCI machine learning repository," 2017.

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

[59] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.

[60] D. Cai, X. He, J. Han, and T. S. Huang, "Graph regularized nonnegative matrix factorization for data representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1548–1560, 2010. Data: `http://www.cad.zju.edu.cn/home/dengcai/Data/MLData.html`.

[61] K. B. Petersen and M. S. Pedersen, "The matrix cookbook, nov 2012," *URL http://www2. imm. dtu. dk/pubdb/p. php*, vol. 3274, 2012.

[62] M. J. Wainwright, *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*, ch. Metric entropy and its uses, pp. 121–158. Cambridge Series in Statistical and Probabilistic Mathematics, Cambridge University Press, 2019.

[63] R. Tibshirani, "CMU Machine Learning 10-725 Lecture Slides: Proximal Gradient Descent and Acceleration," 2015. URL: `https://www.stat.cmu.edu/~ryantibs/convexopt-S15/lectures/08-prox-grad.pdf`. Last visited on 18/May/2020.

[64] G. H. Chen and D. Shah, "Explaining the success of nearest neighbor methods in prediction," *Foundations and Trends in Machine Learning*, vol. 10, no. 5-6, pp. 337–588, 2018.

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

[66] L. Torresani and K.-c. Lee, "Large margin component analysis," in *Advances in Neural Information Processing Systems*, pp. 1385–1392, 2007.

[67] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *International Workshop on Similarity-Based Pattern Recognition*, pp. 84–92, Springer, 2015.

[68] A. Agresti, *Categorical data analysis*, vol. 482. John Wiley & Sons, 2003.

[69] M. Alamuri, B. R. Surampudi, and A. Negi, "A survey of distance/similarity measures for categorical data," in *International Joint Conference on Neural Networks*, pp. 1907–1914, 2014.

[70] Y. Zhang, Y.-m. Cheung, and K. C. Tan, "A unified entropy-based distance metric for ordinal-and-nominal-attribute data clustering," *IEEE Transactions on Neural Networks and Learning Systems*, 2019.

[71] S. Cost and S. Salzberg, "A weighted nearest neighbor algorithm for learning with symbolic features," *Machine Learning*, vol. 10, no. 1, pp. 57–78, 1993.

[72] V. Cheng, C.-H. Li, J. T. Kwok, and C.-K. Li, "Dissimilarity learning for nominal data," *Pattern Recognition*, vol. 37, no. 7, pp. 1471–1477, 2004.

[73] J. Xie, B. Szymanski, and M. Zaki, "Learning dissimilarities for categorical symbols," in *Feature Selection in Data Mining*, pp. 97–106, 2010.

[74] Y. Shi, W. Li, and F. Sha, "Metric learning for ordinal data," in *AAAI Conference on Artificial Intelligence*, 2016.

[75] C. Zhu, L. Cao, Q. Liu, J. Yin, and V. Kumar, "Heterogeneous metric learning of categorical data with hierarchical couplings," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 7, pp. 1254–1267, 2018.

[76] S. Tsang, B. Kao, K. Y. Yip, W.-S. Ho, and S. D. Lee, "Decision trees for uncertain data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 1, pp. 64–78, 2009.

[77] H. Wang, F. Nie, and H. Huang, "Robust distance metric learning via simultaneous ℓ_1-norm minimization and maximization," in *International Conference on Machine Learning*, pp. 1836–1844, 2014.

[78] D. Wang and X. Tan, "Robust distance metric learning in the presence of label noise," in *AAAI Conference on Artificial Intelligence*, 2014.

[79] D. Wang and X. Tan, "Robust distance metric learning via bayesian inference," *IEEE Transactions on Image Processing*, vol. 27, no. 3, pp. 1542–1553, 2017.

[80] G. R. Lanckriet, L. E. Ghaoui, C. Bhattacharyya, and M. I. Jordan, "A robust minimax approach to classification," *Journal of Machine Learning Research*, vol. 3, pp. 555–582, 2002.

[81] R. Babbar and B. Schölkopf, "Data scarcity, robustness and extreme multi-label classification," *Machine Learning*, vol. 108, no. 8-9, pp. 1329–1351, 2019.

[82] P. A. Gutierrez, M. Perez-Ortiz, J. Sanchez-Monedero, F. Fernandez-Navarro, and C. Hervas-Martinez, "Ordinal regression methods: survey and experimental study," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 1, pp. 127–146, 2015.

[83] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *International Conference on Learning Representations*, 2015.

[84] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang, "A tutorial on energy-based learning," *Predicting Structured Data*, 2006.

[85] O. L. Mangasarian, *Nonlinear programming*. SIAM, 1994.

[86] L. Luo, J. Xu, C. Deng, and H. Huang, “Robust metric learning on grassmann manifolds with generalization guarantees,” in *AAAI Conference on Artificial Intelligence*, vol. 33, pp. 4480–4487, 2019.

[87] M. A. Cox and T. F. Cox, “Multidimensional scaling,” in *Handbook of Data Visualization*, pp. 315–347, Springer, 2008.

[88] E. Frank, M. A. Hall, and I. H. Witten, “The WEKA workbench. online appendix for “data mining: Practical machine learning tools and techniques”,” 2016.

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

[90] D. Yin, R. Kannan, and P. Bartlett, “Rademacher complexity for adversarially robust generalization,” in *International Conference on Machine Learning*, pp. 7085–7094, 2019.

[91] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning*. MIT press, 2018.

[92] M. Ledoux and M. Talagrand, *Probability in Banach Spaces: isoperimetry and processes*. Springer Science & Business Media, 2013.

[93] C. McDiarmid, “On the method of bounded differences,” *Surveys in Combinatorics*, vol. 141, no. 1, pp. 148–188, 1989.

[94] S. Clémençon, G. Lugosi, and N. Vayatis, “Ranking and empirical minimization of U-statistics,” *The Annals of Statistics*, vol. 36, no. 2, pp. 844–874, 2008.

[95] F. Rodrigues, F. Pereira, and B. Ribeiro, “Gaussian process classification and active learning with multiple annotators,” in *International Conference on Machine Learning*, pp. 433–441, 2014.

[96] M. Huai, C. Miao, Y. Li, Q. Suo, L. Su, and A. Zhang, “Metric learning from probabilistic labels,” in *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1541–1550, 2018.

[97] J. Lin, “Divergence measures based on the shannon entropy,” *IEEE Transactions on Information theory*, vol. 37, no. 1, pp. 145–151, 1991.

[98] L. Van der Maaten, E. O. Postma, and H. J. van den Herik, “Matlab toolbox for dimensionality reduction,” *MICC, Maastricht University*, 2007.

[99] F. Rodrigues, F. Pereira, and B. Ribeiro, “Learning from multiple annotators: distinguishing good from random labelers,” *Pattern Recognition Letters*, vol. 34, no. 12, pp. 1428–1436, 2013.

[100] D. M. Endres and J. E. Schindelin, “A new metric for probability distributions,” *IEEE Transactions on Information Theory*, vol. 49, no. 7, pp. 1858–1860, 2003.

[101] T. Kailath, “The divergence and bhattacharyya distance measures in signal selection,” *IEEE Transactions on Communication Technology*, vol. 15, no. 1, pp. 52–60, 1967.

[102] A. L. Gibbs and F. E. Su, “On choosing and bounding probability metrics,” *International Statistical Review*, vol. 70, no. 3, pp. 419–435, 2002.

[103] L. v. d. Maaten and G. Hinton, “Visualizing data using t-sne,” *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[104] D. Tarlow, K. Swersky, L. Charlin, I. Sutskever, and R. Zemel, “Stochastic k-neighborhood selection for supervised and unsupervised learning,” in *International Conference on Machine Learning*, pp. 199–207, 2013.

[105] C. Gupta, A. S. Suggala, A. Goyal, H. V. Simhadri, B. Paranjape, A. Kumar, S. Goyal, R. Udupa, M. Varma, and P. Jain, “Protonn: Compressed and accurate knn for resource-scarce devices,” in *International Conference on Machine Learning*, pp. 1331–1340, 2017.

[106] X. Geng, “Label distribution learning,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 7, pp. 1734–1748, 2016.

[107] D. Wang and X. Tan, “Bayesian neighborhood component analysis,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 7, pp. 3140–3151, 2017.

[108] M. Lohaus, P. Hennig, and U. von Luxburg, “Uncertainty estimates for ordinal embeddings,” *arXiv preprint arXiv:1906.11655*, 2019.

[109] B. Lakshminarayanan, A. Pritzel, and C. Blundell, “Simple and scalable predictive uncertainty estimation using deep ensembles,” in *Advances in Neural Information Processing Systems*, pp. 6402–6413, 2017.