LARGE-SCALE HIGH DIMENSIONAL DISTANCE METRIC LEARNING AND ITS APPLICATION TO COMPUTER VISION

By

Qi Qian

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science - Doctor of Philosophy

2015

ABSTRACT

LARGE-SCALE HIGH DIMENSIONAL DISTANCE METRIC LEARNING AND ITS APPLICATION TO COMPUTER VISION

By

Qi Qian

Learning an appropriate distance function (i.e., similarity) is one of the key tasks in machine learning, especially for distance based machine learning algorithms, e.g., k-nearest neighbor classifier, k-means clustering, etc. Distance metric learning (DML), the subject to be studied in this dissertation, is designed to learn a metric that pulls the examples from the same class together and pushes the examples from different classes away from each other. Although many DML algorithms have been developed in the past decade, most of them can handle only small data sets with hundreds of features, significantly limiting their applications to real world applications that often involve millions of training examples represented by hundreds of thousands of features. Three main challenges are encountered to learn the metric from these large-scale high dimensional data: (i) To make sure that the learned metric is a Positive Semi-Definitive (PSD) matrix, a projection into the PSD cone is required at every iteration, whose cost is cubic in the dimensionality making it unsuitable for high dimensional data; (ii) The number of variables that needs to be optimized in DML is quadratic in the dimensionality, which results in the slow convergence rate in optimization and high requirement of memory storage; (iii) The number of constraints used by DML is at least quadratic, if not cubic, in the number of examples depending on if pairwise constraints or triplet constraints are used in DML. Besides, features can be redundant due to high dimensional representations (e.g., face features) and DML with feature selection is preferred

for these applications.

The main contribution of this dissertation is to address these challenges both theoretically and empirically. First, for the challenge arising from the PSD projection, we exploit the mini-batch strategy and adaptive sampling with smooth loss function to significantly reduce the number of updates (i.e., projections) while keeping the similar performance. Second, for the challenge arising from high dimensionality, we propose a dual random projection approach, which enjoys the light computation due to the usage of random projection and at the same time, significantly improves the effectiveness of random projection. Third, for the challenge with large-scale constraints, we develop a novel multi-stage metric learning framework. It divides the original optimization problem into multiple stages. It reduces the computation by adaptively sampling a small subset of constraints at each stage. Finally, to handle redundant features with group property, we develop a greedy algorithm that selects feature group and learns the corresponding metric simultaneously at each iteration leading to further improvement of learning efficiency when combined with adaptive mini-batch strategy and incremental sampling. Besides the theoretical and empirical investigation of DML on the benchmark datasets of machine learning, we also apply the proposed methods to several important computer vision applications (i.e., fine-grained visual categorization (FGVC) and face recognition).

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor Dr. Rong Jin. I met him for the first time when I was a master student at Nanjing University. The talk with him opened a window with a different but very insightful and beautiful scenery for me. When I began my PhD study at MSU, he devoted a lot of his time to train me thinking in a mathematical way to solve learning problems, where I found my passion and happiness in research. His strict but friendly guidance is exhaustive and tireless, which makes me more and more confident about my career. His relentless passion for academic research also influences me much beyond the study. I'm very fortunate to have Dr. Jin as my advisor.

I also would like to thank other committee members: Dr. Pang-Ning Tan, Dr. Sara Selin Aviyente and Dr. Xiaoming Liu. I would like to thank my colleagues at MSU. They are: Fengjie Li, Mehrdad Mahdavi, Tianbao Yang, Lijun Zhang, Jinfeng Yi, Zheyun Feng, BeiBei Liu, Lanbo She and Qiaozi Gao. They not only discussed with me about my research, but also helped me to make my life in East Lansing better. I also would like to thank our professional and patient staffs at the department of computer science: Norma Teague, Linda Moore and Katherine Trinklein. I wish Norma and Linda enjoy their retired life.

I would like to thank my colleagues in NEC Laboratories America, they are: Shenghuo Zhu, Yuanqing Lin and Xiaoyu Wang. I had two summer internships there and my mentor Shenghuo Zhu inspired me a lot on both research and engineering. I would like to thank people in Alibaba Seattle office and silicon valley office, where I took my recent summer intern. I would like to thank Luo Si, Jun Wang, Jun Tao, ZhiPan Li and Xu Xie.

I want to thank Dr. Zhi-Hua Zhou, who is my master advisor at Nanjing University. Without his help, I would not have touched machine learning and began such a wonderful

research trip. His devoted attitude towards both work and life benefits me a lot since then.

Finally, I would like to thank my dear family. Without supports from them, I cannot finish my study so smoothly. I thank my wife Juhua Hu, who helps me much no matter research or life. Every publication of mine has her contribution. The sweetest acknowledgement is to my son, Michael Jiabo Qian, who brings me a total different role and life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

Chapter 1

Introduction

Machine learning, as a subfield of artificial intelligence, involves automatically improving from experience and has been successfully applied for many real world applications [83]. Distance functions are essential to many machine learning tasks. For example, k-nearest neighbor (k-NN) classifier [1] assigns the test example with the most frequent label in its k nearest neighbors from training set. K-means clustering algorithm [53] outputs clusters according to the distance from each instance to k centers. However, Euclidean distance with hand crafted features may not be sufficient to capture the differences between different classes or clusters. Fig. 1.1 provides an example where the k-NN classifier with Euclidean distance is used to identify the flower type "English marigold". Using LLC features [113], we found that the example from a different class (i.e., "Common dandelion") is more closer to the target example than the one from the same class. Therefore, learning an appropriate distance function becomes the key for distance based methods.

In this dissertation, we will study distance metric learning (DML), which aims to learn a metric that pulls the examples from the same class together and pushes the examples from the different classes away from each other according to the supervised information. We start the discussion by introducing the form of the distance metric and supervision in DML.

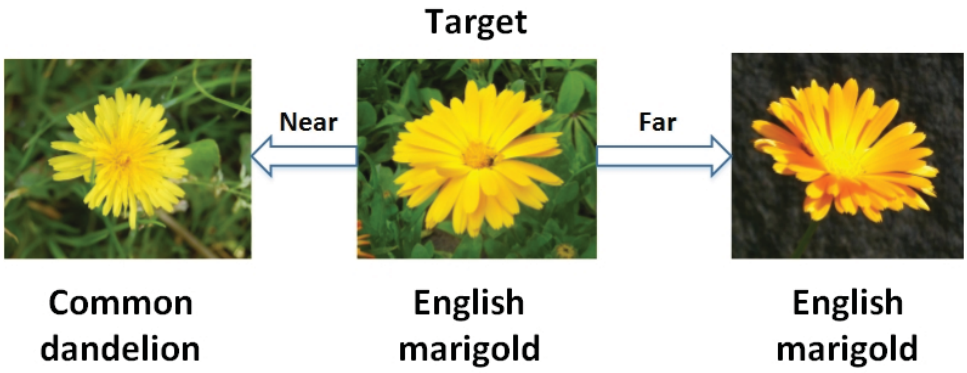


Figure 1.1: Illustrate the problem of Euclidean distance. The left image is more closer than the right one to the target image under Euclidean distance while it is actually from a different class "Common dandelion".

1.1 Distance Metric

Distance function measures the distance between any two examples, which could be denoted as $\text{dist}(\mathbf{x}_i, \mathbf{x}_j)$ for d-dimensional examples $\mathbf{x}_i$ and $\mathbf{x}_j$. Most existing DML methods adopt the form of "Mahalanobis distance" [79]: $\text{dist}_M(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^\top M(\mathbf{x}_i - \mathbf{x}_j)$, where M is called a **distance metric** with the size of $d \times d$. It is easy to verify that the distance function is equivalent to Euclidean distance when M is an identity matrix. The target of DML is to learn a better distance function (i.e., distance metric) than Euclidean distance to evaluate the distances between pairs of examples correctly. As a distance function, it should satisfy certain properties, such as symmetric, non-negative and triangle inequality. Consequently, the learned metric is required to be a positive semi-definite (PSD) matrix, which is often referred as the **PSD constraint**. We will elaborate strategies on how to handle it in Section 2.1.

1.2 Supervision in Distance Metric Learning

To learn a good distance metric, certain supervised information from the data is needed. Different from many supervised [106] or semi-supervised [123] machine learning approaches, the supervision for distance metric learning appears as pairwise or triplet constraints rather than the labels for training examples. Each pairwise constraint is consisted of **two** examples. The metric is learned to guarantee that the distance of pairs from the same class is smaller than a pre-defined threshold while that of pairs from different classes is large [115]. In contrast, **three** examples are included in a triplet constraint [114]. A good metric should make sure that the examples from the same class is separated from the examples of different classes with a large margin. It is obvious that a single triplet constraint could be divided in to three pairwise constraints. Recently, some researchers proposed quadruplet constraints, which contain four examples in a constraint and could be considered as a variant of triplet constraints [76].

These constraints could be derived from labels or provided by the applications directly. For example, pairwise constraints consist of pairs of examples with the same label and pairs of those from different classes. Some applications can also provide the pairwise constraint directly. In face recognition [63], each training example contains two face images and the label is given as an indicator that these two images belong to the same person or not. Given these constraints, various DML methods are developed and representative ones with pairwise or triplet constraints are described in Section 2.2.1 and Section 2.2.2, respectively.

Note that although dimension reduction purpose methods, e.g., principle component analysis (PCA) [95], linear discriminant analysis (LDA) [54] or unsupervised methods, e.g, locally linear embedding (LLE) [94], isometric feature mapping (ISOMAP) [101], sometimes

are also categorized to distance metric learning, we will focus on the discussion of the general purpose DML methods with pairwise or triplet supervision in this study.

1.3 Computational Challenges in Distance Metric Learning

Given these supervised information, the objective of DML is optimizing over pairwise or triplet constraints while keeping the learned metric in the PSD cone. Many DML methods have been developed under this framework, however, little of them try to address the fundamental issues in DML. The computational challenges are mainly from three aspects.

- To make sure the learned metric is a PSD matrix, most DML methods have to project the intermediate solution onto the PSD cone at **every iteration**. The projection step requires the eigen-decomposition operator, which costs $\mathcal{O}(d^3)$ (at least $\mathcal{O}(d^2)$ and d is the dimensionality of data).

- The number of variables that need to be optimized increases from $\mathcal{O}(d)$ in linear model to $\mathcal{O}(d^2)$ in DML. It results in a slower convergence rate in solving the related optimization problem [89]. In addition, d^2 variables become too huge to be stored in the memory when the dimensionality of data is sufficiently large.

- Large number of constraints are needed to avoid overfitting when learning a metric. The number of constraints could be up to $\mathcal{O}(n^3)$ (i.e., triplet constraints), where n is the number of examples. It means that the total cost of DML could be up to $\mathcal{O}(d^3n^3)$.

Besides, to capture the details of images in computer vision (e.g., face recognition), over-completed descriptors are sampled from each image, which results in high dimensional

representations and redundant features.

The occurrence of all these problems makes the large-scale high dimensional DML an extremely difficult task, and we will study it extensively in this work.

1.4 Applications of Distance Metric Learning to Computer Vision

Distance metric learning is an important subject in machine learning, and has found applications in many domains, including information retrieval [56], supervised classification [114], clustering [115, 23] and domain adaptation [93]. Besides machine learning community, other research groups, e.g., computer vision and data mining [110, 111], also realize the importance of DML, and have applied it for many real world applications.

In computer vision, DML is first found to be very helpful for image classification [49, 82, 108, 38], visual object recognition [36, 74] and image retrieval [57, 24, 112]. Image classification and object recognition tasks are often solved as the multi-class classification problem. They first obtain a metric according to the pairwise or triplet information and then apply k-NN classifier for classification. Since images are very sensitive to different poses, light conditions and angles, DML can learn an invariant space to obtain the semantic difference from low level features between different objects. In a standard image retrieval scenario, training set consists of pairwise or triplet constraints and a distance metric is learned to rank these pairs appropriately. When there comes a query image, the system should output the images that are most similar to the query one, which is equivalent to obtaining the images closest to the query under the learned metric.

In addition, DML is also adopted for face verification problem [59, 50, 29, 17, 71]. Face

verification is very practical and has already been used in many real businesses, e.g., terrorist detection, border control and access control system [64]. After obtaining a metric via optimizing lots of pairs of images from the same or different persons, the access control system could identify the person automatically and keys are not necessary for the house. For example, when a stranger comes, the camera in the door can take a photo for him and then compare it to the photos of people who live in the house. If the pairwise comparison under the learned metric returns true, the door will open, otherwise it will keep close. Compared with the traditional access badges, face is naturally unique for each person and is more difficult or even impossible to copy, which makes the house much safer. Note that different from conventional multi-class classification mission, there could be billion people in the dataset and each person has little number of images (e.g., one per person), which makes most of existing classifiers, especially one-vs-all methods, failed for this classification problem with huge number of classes. Besides these applications, DML is also adopted by object tracking [27, 104], video event detection [4, 87], etc. in computer vision.

In this work, we will introduce DML to fine-grained visual categorization (FGVC) [7]. In contrast to classifying an image to a basic class, FGVC requires to categorize the image to a subordinate class, where classes only have subtle difference and the number of classes could be very large. Furthermore, the images in the same class could be very different due to the different poses, examples, etc., which means the intra-class variant is large. The properties of DML, which is independent from the number of classes compared to one-vs-all strategy and flexible to handle large intra-class invariant, make it appropriate for FGVC. To the best of our knowledge, this is the first work to apply DML for the challenging FGVC task.

1.5 Contributions

In this study, we develop several randomized algorithms to alleviate the challenges described in Section 1.3. First, to reduce the number of PSD projections, we combine the mini-batch stochastic gradient descent method with an adaptive sampling strategy. Second, we develop a dual random projection method to handle the large number of variables. Finally, we propose a multi-stage metric learning framework to divide the learning procedure into a series of subproblems, where each one only has an epoch ($\mathcal{O}(n)$) of active constraints, and the total computational cost of the resulting new framework of DML is linear in the dimensionality and the number of examples ($\mathcal{O}(dn)$). Besides these, we investigate the feature selection problem and develop a greedy method to select features and learn metrics simultaneously.

The detailed contributions of the dissertation are described as follows.

- We first exploit the combination of the mini-batch strategy with smooth loss function for DML. Then, we propose an adaptive sampling approach for efficient DML. We verify, both theoretically and empirically, the efficiency and effectiveness of the mini-batch strategy with smooth loss and the adaptive sampling approach for DML, respectively. Finally, we present two hybrid approaches that exploit the combination of the mini-batch strategy with adaptive sampling for DML. To the best of our knowledge, it is the first work that reduce the number of updates in DML while keeping the similar performance.

- We propose a dual random projection approach for high dimensional DML. Our approach, on one hand, enjoys the light computation of random projection, and on the other hand, significantly improves the effectiveness of random projection. We verify the effectiveness of the proposed algorithms both empirically and theoretically.

- We develop a novel multi-stage metric learning framework for high dimensional DML to address high dimensional DML with large number of constraints. We divide the original optimization problem into multiple stages. At each stage, only a small subset of constraints that are difficult to be classified by the currently learned metric will be adaptively sampled and used to improve the learned metric. Then we extend the dual random projection strategy to solve each subproblem. The empirical study with standard FVGC benchmark datasets verifies that our framework is both effective and efficient compared to the state-of-the-art FGVC approaches.

- We propose a greedy algorithm that can select descriptors (i.e., features) and learn the corresponding metrics simultaneously for face recognition with the guaranteed convergence rate. To alleviate the high computational cost of exhausted search that finds only one feature group at each iteration, we investigate the adaptive mini-batch strategy and incremental sampling respectively, and the hybrid algorithm that combines these two strategies is also studied. The empirical study on face recognition confirms the effectiveness and efficiency of the proposed methods.

Chapter 2

Literature Survey

In this chapter, we will introduce the existing DML methods briefly. Section 2.1 describes the general strategies to address the PSD constraint that all DML methods have to deal with. After that, Section 2.2 - 2.4 present the popular DML methods. Finally, Section 2.5 summarizes the theoretical analysis for DML.

2.1 PSD Constraint in Distance Metric Learning

Before introducing the specific DML methods, we first demonstrate the strategies to handle PSD constraint, which keeps the learned metric in the PSD cone, in DML. Both early DML methods that require the gradient from all constraints at each iteration [115, 45], and efficient approaches [66, 98] that deal with only one constraint at each iteration by exploiting the techniques of online learning or stochastic optimization, share one common strategy: in order to ensure that the learned distance metric is PSD, these approaches require, at each iteration, projecting the learned distance metric $\tilde{M}$ onto the PSD cone by solving the problem

$$\min_{M \in PSD} \|M - \tilde{M}\|_F^2$$

The solution is the positive eigenvalues with the corresponding eigenvectors [47] (i.e., $M = \sum_i \lambda_i v_i v_i',\ \lambda_i > 0$ where λ_i and v_i are the i-th eigenvalue and the corresponding eigenvector

of $\tilde{M}$). Therefore, it has to perform the eigen-decomposition for a given matrix, which is computationally expensive (i.e., at least $\mathcal{O}(d^2)$).

Several studies have been proposed to avoid projections in SGD. In [55], the authors developed a projection free SGD algorithm that replaces the projection step with a constrained linear programming problem. In [80], the authors proposed a SGD algorithm with only one projection that is performed at the end of the iterations. Unfortunately, the improvement of the two algorithms in computational efficiency is limited, because they have to compute, *at each iteration*, the minimum eigenvalue and the eigenvector of the updated distance metric, an operation with $O(d^2)$ cost, where d is the dimensionality of the data.

It is noteworthy to mention that in a recent study [24], the authors show empirically that it is possible to learn a good distance metric using online learning without having to perform the projection at each iteration. In fact, only one projection into the PSD cone is performed at the end of online learning to ensure that the resulting matrix is PSD. It is different from the algorithms presented in [55, 80] and no additional mechanisms are needed to prevent the intermediate solutions from being too far away from the PSD cone. We refer this strategy as **one-projection paradigm** in the rest of this dissertation.

2.2 Linear Distance Metric Learning

Most of conventional DML approaches are equivalent to seeking a linear transformation for the input space. In this section, we will describe representative approaches in detail according to the type of constraints used. More examples could be found in two survey papers [117, 73]

2.2.1 Distance Metric Learning with Pairwise Constraints

As mentioned in Section 1.2, DML usually involves two kinds of constraints: pairwise or triplet. Most early works for distance metric learning focus on optimizing pairwise constraints. Xing, et al. [115] proposed one of the earliest methods for the typical DML problem as studied in this dissertation. The objective is maximizing the distance between the pairs of examples from the different classes while keeping the distance between the pairs of examples with the same label less than a pre-defined threshold. Although the problem is solved via a semi-definite programming (SDP) method [12] that is only runnable on small datasets, they show the advantage of DML for distance based methods and simulate more research for this topic. POLA method [96] is then proposed to alleviate the computation cost of DML. It is an online learning method, which only receives a single pairwise constraint at each iteration. If the current metric makes a mistake, it will be updated accordingly, otherwise the metric will be the same. The cost of each iteration is much less since only the gradient from a single constraint is computed. In addition, the PSD projection is only needed when the pair of examples from the same class is misclassified and the cost could be as low as $\mathcal{O}(d^2)$ since the update is a rank one matrix. MCML [45] is the convex relaxed version of NCA [46]. It represents the extreme case of DML that all data points in the same class should be mapped into a single location by the learned metric. For this purpose, it minimizes the KL divergence between the distribution returned by the learned metric and that from the ideal case. It also proposed to keep a low-rank copy after obtaining the metric to achieve dimension reduction. Till now, all of these DML methods may require PSD projection at every iteration, which could be $\mathcal{O}(d^3)$. ITML [32] applied the LogDet regularizer to avoid PSD projections. LogDet divergence is a Bregman matrix divergence [13] between two matrices.

By introducing this regularizer, the metric is updated in the inverse form with a rand one matrix at each iteration, which could be finished without computing the inverse practically via the Sherman-Morrison-Woodbury formula. Meanwhile, the updating rule makes sure that the intermediate learned metric is still in the PSD cone. Although there is no PSD projection for ITML, the additional cost is included and the cost for each iteration is still no less than $\mathcal{O}(d^2)$.

2.2.2 Distance Metric Learning with Triplet Constraints

DML methods with pairwise constraints aim to learn a metric that collapse all examples in each class to a small cluster whose radius is the pre-defined threshold. However, it is difficult when the intra-class variance is large, which is often the case in real world applications. Thus, triplet constraints is proposed to handle this problem more flexibly [114]. Each triplet constraint contains three examples: two examples with the same label where one is among the k-nearest neighbors of the target example and one example from the different class who is also in the nearest neighbors of the target example. LMNN [114] then learns the metric to push away the example of different class with a large margin. Fig. 2.1 illustrates the effectiveness of LMNN. Compared with previous DML approaches, it only requires that the examples in the nearest neighbors could be collapse to a small cluster, which preserves the large intra-class variance and is more appropriate for combining with k-NN classifier. Since it borrows the concept of large margin, it also could be analyzed from the view of support vector machine (SVM) [107], where LMNN is similar to learn a series of local SVM-like models [33].

After the success of LMNN, Chechik, et al. [24] applied the triplet constraints with an online learning fashion for efficiency and achieved a good performance for ranking. Surpris-

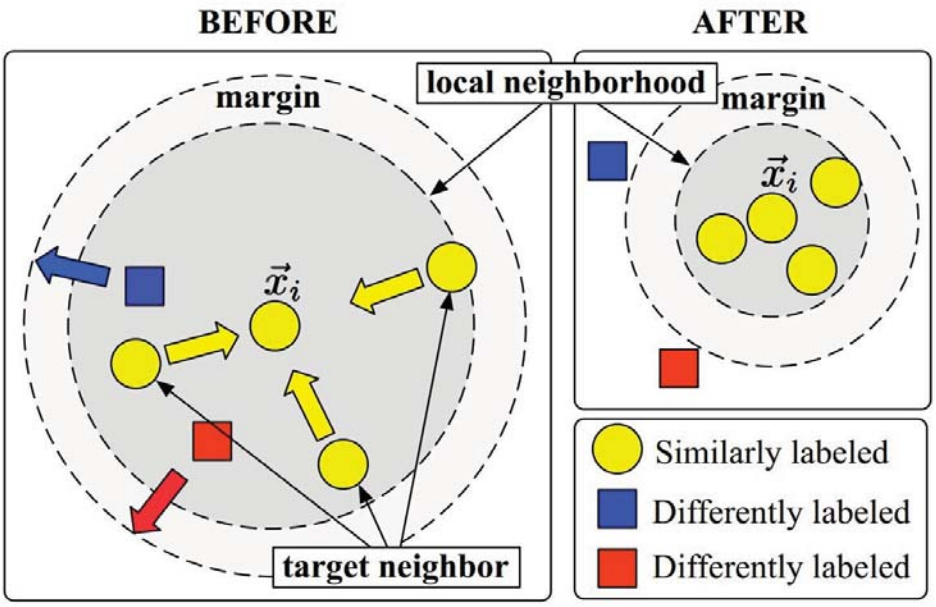


Figure 2.1: The figure is from the work [114] and illustrates the learning procedure of LMNN. The learned metric pushes away the examples that are from different classes but in the nearest neighbors with a large margin.

ingly, they found empirically that one PSD projection at the end of the algorithm will not affect the performance, which implies the potential possibility of getting rid of the expensive PSD projection. Since gradient descent method requires projecting the intermediate solution onto PSD cone, Shat, et al. [98] adopted mini-batch stochastic gradient descent method to balance the cost of computing gradient and that of PSD projection. Furthermore, it also reduces the variance of the stochastic method, which only randomly samples a single triplet at each iteration.

2.3 Nonlinear Distance Metric Learning

Sometimes, linear transformation could not exploit complicated data distribution sufficiently, e.g, face images usually lie on the nonlinear manifold [59]. To handle this nonlinear feature space, **kernel** trick, which has been well studied in linear model [15], is applied to extend the

existing DML methods [45, 32, 114, 102]. Its basic idea is mapping the input examples to a much higher or even infinite dimensional space and then obtaining the linear transformation there. Although it sometimes performs better, the size of learned metric (i.e., dual variables) is $\mathcal{O}(n^2)$ for DML, which is more expensive when $n \gg d$. The alternative way to provide nonlinear learning capacity is **deep learning** (or called deep neural networks), which is very popular these years due to its encouraging performance [72]. Unlike kernel trick, deep learning deals with the input features directly and learns the nonlinear mapping between each layer according to nonlinear activation functions, e.g., tanh, sigmoid, etc. The original structure of deep learning is designed for classification and the pipeline is optimized with the single image as input. By replacing the loss function as pairwise [59] or triplet distance [112] and concatenating the pipelines in parallel, it is convenient to adapt to DML tasks.

Besides these extensions for the single metric learning, there are some studies on learning **multiple metrics** simultaneously. The key part is defining the metric for appropriate components. Some works apply multi-metric learning on data. That is, they learn each metric for each class or local cluster, and sometimes even for each single example [42, 84]. For example, MM-LMNN [114] learns a metric per class and the distance between two examples is defined on the metric corresponding to the label of examples. On the other hand, the multiple metrics also could be learned according to different feature blocks [61, 29]. For example, each face image could be divided into different parts, e.g., nose, eyes, ears, etc., and then PMML [29] learns a single metric on each feature block which represents the different part. Although multiple metrics may improve the performance over a single metric, it intensifies the already expensive optimization problem by increasing the number of metrics. Moreover, some good properties of a single metric are sacrificed, e.g., global transformation, interactions between different components, etc. Since all of these extensions

are based on conventional DML, we will focus on learning a single linear distance metric in this dissertation.

2.4 Improving Efficiency of Distance Metric Learning

Although many DML methods have been developed, the cost of update at each iteration is still ($\mathcal{O}(d^2)$) due to the number of variables. Recently, some works aim to alleviate the high computation cost via sparsity. Different from linear classifier, the sparsity of metric could come from two aspects: low rank and item sparsity.

Since each metric is a PSD matrix, it could be decomposed as $M = LL$, where M is a $d \times d$ matrix, L is a $d \times r$ matrix and r is the rank of M. When $r \ll d$, the total number of variables for optimizing decreases from d^2 to rd, which is only $\mathcal{O}(d)$. Some existing methods apply low rank strategy to accelerate the learning process [114, 31]. The drawback is that the corresponding optimization problem is not convex anymore, hence, it is not guaranteed to obtain a global optimal metric. In addition, these methods require to fix r before applying the DML methods, which easily leads to the suboptimal solution when the true rank is larger than the pre-defined rank. Some studies apply trace norm or fantope regularization [77] to obtain a low rank metric, while the cost for training stage is still at least $\mathcal{O}(d^2)$.

For item sparsity, it could be further categorized into two parts: sparsity on the single item and sparsity on the columns. The former one assumes that only s items have the nonzero value in each column/row, where $s \ll d$ and is corresponding to the L_1 regularizer for each column/row [89]. In the latter scenario, there is only limited number of columns with nonzero items, which is equivalent to penalizing the whole matrix with $L_{2,1}$ regularizer [78]. These methods could output the sparse metric which is efficient for test stage, however, the

learning process still involves $\mathcal{O}(d^2)$ variables.

2.5 Generalization Performance of Distance Metric Learning

Besides developing different DML methods for applications, some studies begin to analyze the generalization performance of DML methods theoretically [68, 16, 6]. Given a specific DML method and the training set that are i.i.d. sampled from an unknown distribution, we are interested in the learned model's prediction error on an arbitrary pairwise constraint from the true distribution, which is known as **generalization error**. However, the only evaluation that we could have is its prediction error on the training set, which is usually referred as **empirical error**. According to the theory of stability, Jin, et al. [68] proposed the first work that uncovered the relationship between them in DML and found that the empirical error converges to the generalization error at the rate of $\mathcal{O}(\sqrt{\frac{1}{n}})$ with high probability. After that, some other studies show the similar result via different techniques [16, 6]. Cao, et al. further researched the influence from the different regularizers, such as Frobenius norm, L_1 norm, $L_{2,1}$ norm and trace norm.

These analyses are based on the classification accuracy for pairwise constraints, while the more interesting question is the performance of k-NN classifier with the learned metric. Guo, et al. [51] proposed the theory that bridges this gap and shows that the generalization error of the linear SVM with the similarity matrix from the learned metric is upper bounded by the generalization error of the corresponding metric learning, which means that the generalization performance of linear classifier is guaranteed by that of metric learning.

Chapter 3

Large-scale Distance Metric Learning by Adaptive Sampling and Mini-Batch Stochastic Gradient Descent (SGD)

In this chapter[1], we will address the challenge from PSD projections ($\mathcal{O}(d^3)$). Unlike previous efforts for handling PSD constraint as mentioned in Section 2.1, we will focus on reducing the number of PSD projections rather than optimizing each projection in SGD, which is the popular strategy for large-scale data. As a result, the key challenge in developing efficient SGD algorithms for DML is how to reduce the number of projections without affecting the performance of DML.

A common approach for reducing the number of updates and projections in DML is to use the non-smooth loss function. A popular choice of the non-smooth loss function is the hinge loss, whose derivative becomes zero when the input value exceeds a certain threshold. Many online learning algorithms for DML [24, 32, 66] take advantage of the non-smooth loss function to reduce the number of updates and projections. In [98], the authors proposed a

[1]This chapter is adapted from the published paper: Q. Qian, R. Jin, J. Yi, L. Zhang and S. Zhu. Efficient Distance Metric Learning by Adaptive Sampling and Mini-Batch Stochastic Gradient Descent (SGD). Machine Learning Journal (MLJ), 99:3, 353-372, 2015.

structure preserving metric learning algorithm (SPML) that combines a mini-batch strategy with the hinge loss to further reduce the number of updates for DML. It groups multiple constraints into a mini-batch and performs only one update of the distance metric for each mini-batch. But, according to our empirical study, although SPML reduces the running time of the standard SGD algorithm, it results in a significantly worse performance for several datasets, due to the deployment of the mini-batch strategy.

In this chapter, we first develop a new mini-batch based SGD algorithm for DML, termed **Mini-SGD**. Unlike SPML that relies on the hinge loss, the proposed Mini-SGD algorithm exploits a *smooth* loss function for DML. By using a smooth loss function, the proposed algorithm is able to effectively take advantage of the reduction in the variance of gradients achieved by the mini-batch, which in return leads to a better regret bound for online learning [28] and consequentially a more accurate prediction for the learned distance metric. We show theoretically that by using a smooth loss function, Mini-SGD is able to achieve similar convergence rate as the standard SGD algorithm but with significantly less number of updates. The second contribution of this work is to develop a new strategy, termed **adaptive sampling**, for reducing the number of projections in DML. The key idea of adaptive sampling is to first measure the "difficulty" in classifying a constraint using the learned distance metric, and then perform stochastic updating based on the difficulty measure. Finally, we develop two **hybrid approaches** that combine adaptive sampling with mini-batch to further improve the computational efficiency of SGD for DML. We conduct an extensive empirical study to verify the effectiveness and efficiency of the proposed algorithms for DML. We summarize the main contribution of this work as follows:

- To the best of our knowledge, this is the first work that exploits the combination of the mini-batch strategy with smooth loss function for DML. We verify, both theoretically

and empirically, the efficiency and effectiveness of the mini-batch strategy with smooth loss for DML.

- We propose an adaptive sampling approach for efficient DML. We verify, both theoretically and empirically, the efficiency and effectiveness of the adaptive sampling approach for DML.

- We present two hybrid approaches that exploit the combination of the mini-batch strategy with adaptive sampling for DML. Our extensive empirical study verifies that the hybrid approaches are significantly more efficient than both the mini-batch strategy and the adaptive sampling approach.

The rest of this chapter is organized as follows: Section 3.1 describes the proposed SGD algorithms for DML based on mini-batch and adaptive sampling. Two hybrid approaches are presented that combine mini-batch and adaptive sampling for DML. The theoretical guarantees for both mini-batch based and adaptive sampling based SGD are also presented in Section 3.1. Section 3.2 summarizes the results of the empirical study, and Section 3.3 concludes this chapter.

3.1 Improved SGD for DML by Mini-batch and Adaptive Sampling

We first review the basic framework of DML with triplet constraints. We then present two strategies to improve the computational efficiency of SGD for DML, one by mini-batch and the other by adaptive sampling. We present the theoretical guarantees for both strategies,

and defer more detailed analysis to the appendix. At the end of this section, we present two hybrid approaches that combine mini-batch with adaptive sampling for more efficient DML.

Let $\mathcal{X} \subset \mathbb{R}^d$ be the domain for input patterns, where d is the dimensionality. For the convenience of analysis, we assume all the input patterns with bounded norm, i.e. $\forall \mathbf{x} \in \mathcal{X}, |\mathbf{x}|_2 \leq r$. Given a distance metric $M \in \mathbb{R}^{d \times d}$, the distance square between $\mathbf{x}_a$ and $\mathbf{x}_b$, denoted by $\text{dist}_M(\mathbf{x}_a, \mathbf{x}_b)$, is measured by

$$\text{dist}_M(\mathbf{x}_a, \mathbf{x}_b) = (\mathbf{x}_a - \mathbf{x}_b)^\top M (\mathbf{x}_a - \mathbf{x}_b)$$

Let $\Omega = \{M : M \succeq 0, \|M\|_F \leq R\}$ be the domain for distance metric M, where R specifies the domain size. Let $\mathcal{D} = \{(\mathbf{x}_i^1, \mathbf{x}_j^1, \mathbf{x}_k^1), \ldots, (\mathbf{x}_i^N, \mathbf{x}_j^N, \mathbf{x}_k^N)\}$ be the set of triplet constraints used for DML, where $\mathbf{x}_i^t$ is expected to be closer to $\mathbf{x}_j^t$ than to $\mathbf{x}_k^t$. Let $\ell(z)$ be the convex loss function. Define $\Delta(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t; M)$ as

$$\begin{aligned}
&\Delta(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t; M) = \text{dist}_M(\mathbf{x}_i^t, \mathbf{x}_k^t) - \text{dist}_M(\mathbf{x}_i^t, \mathbf{x}_j^t) \\
&= \left\langle M, (\mathbf{x}_i^t - \mathbf{x}_k^t)(\mathbf{x}_i^t - \mathbf{x}_k^t)^\top - (\mathbf{x}_i^t - \mathbf{x}_j^t)(\mathbf{x}_i^t - \mathbf{x}_j^t)^\top \right\rangle = \langle M, A_t \rangle
\end{aligned}$$

where

$$A_t = (\mathbf{x}_i^t - \mathbf{x}_k^t)(\mathbf{x}_i^t - \mathbf{x}_k^t)^\top - (\mathbf{x}_i^t - \mathbf{x}_j^t)(\mathbf{x}_i^t - \mathbf{x}_j^t)^\top$$

Given the triplet constraints in $\mathcal{D}$ and the domain in Ω, we learn an optimal distance metric $M \in \mathbb{R}^{d \times d}$ by solving the following optimization problem

$$\min_{M \in \Omega} \quad \widehat{\mathcal{L}}(M) = \frac{1}{N} \sum_{t=1}^{N} \ell\left(\Delta(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t; M)\right) \tag{3.1}$$

We also define the expectation of the loss function as

$$\mathcal{L}(M) \quad = \quad \mathrm{E}\left[\ell(\Delta(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k; M))\right] \tag{3.2}$$

where the expectation is taken over $\mathbf{x}_i$, $\mathbf{x}_j$ and $\mathbf{x}_k$.

The key idea of online DML is to minimize the empirical loss $\widehat{\mathcal{L}}(M)$ by updating the distance metric based on one sampled constraint at each iteration. More specifically, at iteration t, it samples a triplet constraint $(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t)$, and updates the distance metric M_t to M_{t+1} by

$$M_{t+1} = \Pi_\Omega\left(M_t - \eta \ell'(\Delta(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t; M_t))A_t\right)$$

where $\eta > 0$ is the step size, $\ell'(\cdot)$ is the derivative and $\Pi_\Omega(M)$ projects a matrix M onto the domain Ω. The following proposition shows $\Pi_\Omega(M)$ can be computed in two steps, i.e. first projecting M onto the PSD cone, and then scaling the projected M to fit in with the constraint $\|M\|_F \leq R$.

Proposition 1. *[12] We have*

$$\Pi_\Omega(M) = \frac{1}{\max(\|M'\|_F/R, 1)} P(M)$$

Here $P(M)$ projects matrix M onto the PSD cone and is computed as

$$P(M) = \sum_{i=1}^{d} \max(\lambda_i, 0)\mathbf{v}_i\mathbf{v}_i^\top$$

where $(\lambda_i, \mathbf{v}_i), i = 1, \ldots, d$ are the eigenvalues and corresponding eigenvectors of M.

As indicated by Proposition 1, $\Pi_\Omega(M)$ requires projecting distance metric M onto the

PSD cone, an expensive operation that requires eigen-decomposition of M.

Finally, we approximate the hinge loss by a smooth loss in our study

$$\ell(z) = \frac{1}{L}\log(1 + \exp(-L(z-1))) \tag{3.3}$$

where $L > 0$ is a parameter that controls the approximation error: the larger the L, the closer $\ell(z)$ is to the hinge loss. Note that the smooth approximation of the hinge loss was first suggested in [122] for classification and was later verified by an empirical study in [119]. The key properties of the loss function $\ell(z)$ in (3.3) are given in the following proposition.

Proposition 2. *For the loss function defined in (3.3), we have*

$$\forall z \in \mathbb{R}, \quad |\ell'(z)| \leq 1, \ |\ell'(z)| \leq L\ell(z)$$

Compared to the hinge loss function, the main advantage of the loss function in (3.3) is that it is a smooth loss function. As will be revealed by our analysis, it is the smoothness of the loss function that allows us to effectively explore both the mini-batch and adaptive sampling strategies for more efficient DML without having to sacrifice the prediction performance.

3.1.1 Mini-batch SGD for DML (Mini-SGD)

Mini-batch SGD improves the computational efficiency of online DML by grouping multiple constraints into a mini-batch and only updating the distance metric once for each mini-batch. For brevity, we will refer to this algorithm as **Mini-SGD** for the rest of the chapter.

Algorithm 1 Mini-batch Stochastic Gradient Descent (Mini-SGD) for DML

1: **Input:** triplet constraints $\{(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t)\}_{t=1}^N$, step size η, mini-batch size b, and domain size R
2: Initialize $M_1 = I$ and $T = N/b$
3: **for** $t = 1, \ldots, T$ **do**
4: Sample b triplet constraints $\{(\mathbf{x}_i^{t,s}, \mathbf{x}_j^{t,s}, \mathbf{x}_k^{t,s})\}_{s=1}^b$
5: Update the distance metric by

$$M_{t+1} \quad = \quad \Pi_\Omega \left(M_t - \eta \nabla \ell_t(M_t) \right)$$

6: **end for**
7: **return** $\bar{M} = \frac{1}{T} \sum_{t=1}^T M_t$

Let b be the batch size. At iteration t, it samples b triplet constraints, denoted by

$$(\mathbf{x}_i^{t,s}, \mathbf{x}_j^{t,s}, \mathbf{x}_k^{t,s}), s = 1, \ldots, b,$$

and defines the mini-batch loss at iteration t as

$$\ell_t(M_t) = \frac{1}{b} \sum_{s=1}^b \ell \left(\Delta(\mathbf{x}_i^{t,s}, \mathbf{x}_j^{t,s}, \mathbf{x}_k^{t,s}; M_t) \right)$$

Mini-batch DML updates the distance metric M_t to M_{t+1} using the gradient of the mini-bach loss function $\ell_t(M)$, i.e.,

$$M_{t+1} = \Pi_\Omega \left(M_t - \eta \nabla \ell_t(M_t) \right)$$

Algorithm 1 gives the detailed steps of Mini-SGD for DML, where in step 5 Proposition 1 is used to efficiently compute the projection $\Pi_\Omega(\cdot)$.

The theorem below provides the theoretical guarantee for the Mini-SGD algorithm for DML using the smooth loss function defined in (3.3).

Theorem 1. *Let $\bar{M}$ be the solution output by Algorithm 1 that uses the loss function defined*

in (3.3). Let M_* be the optimal solution that minimizes $\mathcal{L}(M)$. Assuming $\|A_t\|_F \leq A$ for any triplet constraint and $\eta < 1/(3LA^2)$, we have

$$\mathrm{E}[\mathcal{L}(\bar{M})] \leq \frac{\mathcal{L}(M_*)}{1 - 3\eta LA^2} + \frac{bR^2}{2(1 - 3\eta LA^2)\eta N} \tag{3.4}$$

where the expectation is taken over the sequence of triplet constraints.

Remark 1 First, we observe that the second term in the upper bound in (3.4), i.e., $bR^2/[2(1-3\eta LA^2)\eta N]$, has a linear dependence on mini-batch size b, implying that the larger the b, the less accurate the distance metric learned by Algorithm 1. Hence, by adjusting parameter b, the size of mini-batch, we are able to make appropriate tradeoff between the prediction accuracy and the computational efficiency: the smaller the b, the more accurate the distance metric but with more updates and consequentially higher computational cost. When $\mathcal{L}(M_*) = 0$, we have $\mathrm{E}[\mathcal{L}(\bar{M})] = O(1/N)$, i.e. the expected prediction error will be reduced at the rate of b/N, significantly faster than that of the mini-batch SGD algorithm (i.e. $O(1/\sqrt{N})$) given in [28]. Second, if we set η as:

$$\eta = \frac{\rho}{3LA^2(1 + 2\rho)} \quad where \quad \rho = \frac{3bLR^2A^2}{N\mathcal{L}(M_*)} \tag{3.5}$$

we have

$$\mathrm{E}[\mathcal{L}(\bar{M})] \leq 2\mathcal{L}(M_*) + \frac{6bLA^2R^2}{N} \tag{3.6}$$

Although the step size in (3.5) requires the knowledge of $\mathcal{L}(M_*)$ that is usually unavailable, as suggested in [67], $\mathcal{L}(M_*)$ can be estimated empirically using part of training examples. Third, the bound in (3.4) reveals the importance of using a smooth loss function as the

Algorithm 2 Adaptive Sampling Stochastic Gradient Descent (AS-SGD) for DML

Input: triplet constraints $\{(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t)\}_{t=1}^N$, step size η, and domain size R
Initialize $M_1 = I$
for $t = 1, \ldots, N$ **do**
  Sample a binary random variable Z_t with

$$\Pr(Z_t = 1) = |\ell'(\Delta(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t; M_t)|$$

  if $Z_t = 1$ **then**
    Update the distance metric by

$$\tau_t = \text{sign}(\ell'(\Delta(\mathbf{x}_\text{i}^\text{t}, \mathbf{x}_\text{j}^\text{t}, \mathbf{x}_\text{k}^\text{t}; \text{M}_\text{t})), \quad \text{M}_{\text{t}+1} = \Pi_\Omega\left(\text{M}_\text{t} - \eta\tau_\text{t}\text{A}_\text{t}\right)$$

  end if
end for
return $\bar{M} = \frac{1}{N}\sum_{t=1}^N M_t$

last term in (3.4) is proportional to L that measures the smoothness of the loss function. As a result, using a non-smooth loss function (e.g. hinge loss) in DML will not be able to benefit the advantage of the mini-batch strategy. Finally, unlike the analysis in [98] (i.e. Theorem 2) that only consider the case when $b = 1$, Theorem 1 provide a general result for any mini-batch size b.

3.1.2 Adaptive Sampling based SGD for DML (AS-SGD)

We now develop a new approach for reducing the number of updates in SGD in order to improve the computational efficiency of DML. Instead of updating the distance metric at each iteration, the proposed strategy introduces a random binary variable to decide if the distance metric M_t will be updated given a triplet constraint $(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t)$. More specifically, it computes the derivative $\ell'(\Delta(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t; M_t))$, and samples a random variable Z_t with probability

$$\Pr(Z_t = 1) = |\ell'(\Delta(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t; M_t))|$$

The distance metric will be updated only when $Z_t = 1$. According to Proposition 2, we have $|\ell'(\Delta(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t; M_t))| \leq L\ell(\Delta(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t; M_t))$ for the smooth loss function given in (3.3), implying that a triplet constraint has a high chance to be used for updating the distance metric if it has a large loss. Therefore, the essential idea of the proposed adaptive sampling strategy is to give a large chance to update the distance metric when the triplet is difficult to be classified and a low chance when the triplet can be classified correctly with large margin. We note that an alternative strategy is to sample a triplet constraint $(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t)$ base on its loss $\ell(\Delta(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t; M_t))$. We did not choose the loss as the basis for updating because it is the derivative, not the loss, that will be used by SGD for updating the distance metric. The detailed steps of adaptive sampling based SGD for DML is given in Algorithm 2. We refer to this algorithm as **AS-SGD** for short in the rest of this chapter.

The theorem below provides the performance guarantee for AS-SGD. It also bounds the number of updates $\sum_{t=1}^{T} Z_t$ for AS-SGD.

Theorem 2. *Let $\bar{M}$ be the solution output by Algorithm 2 that uses the loss function defined in (3.3). Let M_* be the optimal solution that minimizes $\mathcal{L}(M)$. Assuming $\|A_t\|_F \leq A$ for any triplet constraint and $\eta < 2/LA^2$, we have*

$$\mathrm{E}\left[\mathcal{L}(\bar{M})\right] \leq \frac{\mathcal{L}(M_*)}{1 - \eta LA^2/2} + \frac{R^2}{2\eta N(1 - \eta LA^2/2)} \tag{3.7}$$

and the number of updates bounded by

$$\mathrm{E}\left[\sum_{t=1}^{N} Z_t\right] \leq \frac{NL\mathcal{L}(M_*)}{1 - \eta LA^2/2} + \frac{LR^2}{2\eta(1 - \eta LA^2/2)} \tag{3.8}$$

where the expectation is taken over both the binary random variables $\{Z_t\}_{t=1}^N$ and the se-

quence of triplet constraints.

Remark 2 If we set η as

$$\eta = \frac{2\rho}{LA^2(1+2\rho)} \quad where \quad \rho = \frac{LA^2R^2}{4N\mathcal{L}(M_*)}$$

we have

$$\mathrm{E}[\mathcal{L}(\bar{M})] \leq 2\mathcal{L}(M_*) + \frac{LR^2A^2}{N} \tag{3.9}$$

and

$$\mathrm{E}\left[\sum_{t=1}^{N} Z_t\right] \leq 2NL\mathcal{L}(M_*) + L^2A^2R^2 \tag{3.10}$$

The bounds given in (3.7) and (3.9) share similar structures as those given in (3.4) and (3.6) except that they do not have mini-batch size b that can be used to make tradeoff between the number of updates and the classification accuracy. The number of updates performed by Algorithm 2 is bounded by (3.10). The dominate term in (3.10) is $O(\mathcal{L}(M_*)N)$, implying that Algorithm 2 will have a small number of updates if the optimal distance metric only makes a small number of mistakes for the given set of training triplets. In the extreme case when $\mathcal{L}(M_*) \to 0$, the expected number of updates will be bounded by a constant $L^2A^2R^2$. We note that this is consistent with our intuition: it will be easy to learn a good distance metric when the optimal one only makes a few mistakes, and as a result, only a few updates are needed to find a distance metric that are consistent with most of the training triplets. Compared with the result of perceptron method [96], we do not assume that the dataset is separable, which makes our bound for the number of updates more practically useful.

Algorithm 3 A Framework of Hybrid Stochastic Gradient Descent (Hybrid-SGD) for DML

Input: triplet constraints $\{(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t)\}_{t=1}^N$, step size η, mini-batch size b, and domain size R
Initialize $M_1 = I$ and $T = N/b$
for $t = 1, \ldots, T$ **do**
  Sample b triplets $\{\mathbf{x}_i^{t,s}, \mathbf{x}_j^{t,s}, \mathbf{x}_k^{t,s}\}_{s=1}^b$
  Compute sampling probability $\gamma_t(\{\mathbf{x}_i^{t,s}, \mathbf{x}_j^{t,s}, \mathbf{x}_k^{t,s}\}_{s=1}^b; M_t)$ as in Eqn. 3.11 or 3.12
  Sample a binary random variable Z_t with

$$\Pr(Z_t = 1) = \gamma_t$$

  if $Z_t = 1$ **then**
    Update the distance metric by

$$\begin{aligned} \tau_t &= 1/\gamma_t \\ M_{t+1} &= \Pi_\Omega(M_t - \eta\tau_t \nabla \ell_t(M_t)) \end{aligned}$$

  end if
end for
return $\bar{M} = \frac{1}{T}\sum_{t=1}^T M_t$

3.1.3 Hybrid Approaches: Combining Mini-batch with Adaptive Sampling for DML

Since mini-batch and adaptive sampling improve the computational efficiency of SGD from different aspects, it is natural to combine them together for more efficient DML. Similar to the Mini-SGD algorithm, the hybrid approaches will group multiple triplet constraints into a mini-batch. But, unlike Mini-SGD that updates the distance metric for every mini-batch of constraints, the hybrid approaches follow the idea of adaptive sampling, and introduce a binary random variable to decide if the distance metric will be updated for every mini-batch of constraints. By combining the strength of mini-batch and adaptive sampling for SGD, the hybrid approaches are able to make further improvement in the computational efficiency of DML. Algorithm 3 highlights the key steps of the hybrid approaches.

One of the key steps in the hybrid approaches (step 5 in Algorithm 3) is to choose appropriate sampling probability γ_t for every mini-batch constraints $(\mathbf{x}_i^{t,s}, \mathbf{x}_j^{t,s}, \mathbf{x}_k^{t,s}), s = 1, \ldots, b$. In this work, we study two different choices for sampling probability γ_t:

- The first approach chooses γ_t based on a triplet constraint randomly sampled from a mini-batch. More specifically, given a mini-batch of triplet constraints $\{\mathbf{x}_i^{t,s}, \mathbf{x}_j^{t,s}, \mathbf{x}_k^{t,s}\}_{s=1}^{b}$, it randomly samples an index s' in the range $[1, b]$. It then sets the sampling probability γ_t to be the derivative for the randomly sampled triplet, i.e.,

$$\gamma_t = |\ell'(\Delta(\mathbf{x}_i^{t,s'}, \mathbf{x}_j^{t,s'}, \mathbf{x}_k^{t,s'}; M_t))| \tag{3.11}$$

  We refer to this approach as **HR-SGD**.

- The second approach is based on the average case analysis. It sets the sampling probability as the average derivative measured by the norm of the gradient $\nabla \ell_t(M_t)$, i.e.,

$$\gamma_t = \frac{1}{W} \|\nabla \ell_t(M_t)\|_F \tag{3.12}$$

  where $W = \max_t \|\nabla \ell_t(M_t)\|_F$ and is estimated by sampling. We refer to this approach as **HA-SGD**.

3.2 Experiments

Ten datasets are used to validate the effectiveness of the proposed algorithms. Table 3.1 summarizes the information of these datasets. Datasets *dna*, *letter* [58], *protein* and *sensit* [35] are downloaded from LIBSVM [22]. Datasets *tdt30* and *rcv20* are document corpora: *tdt30* is the subset of tdt2 data [14] comprised of the documents from the 30 most popular categories

Table 3.1: Statistics for the ten datasets used in our empirical study.

	# class	# feature	# train	# test
semeion	10	256	1,115	478
dna	3	180	2,000	1,186
isolet	26	617	6,238	1,559
tdt30	30	200	6,575	2,819
letter	26	16	15,000	5,000
protein	3	357	17,766	6,621
connect4	3	42	47,289	20,268
sensit	3	100	78,823	19,705
rcv20	20	200	477,141	14,185
poker	10	10	1,000,000	25,010

and *rcv20* is the subset of a large rcv1 dataset [5] consisted of documents from the 20 most popular categories. Following [24], we reduce the dimensionality of these document datasets to 200 by principle components analysis (PCA). All the other datasets are downloaded directly from the UCI repository [40]. For all the datasets used in this study, we use the standard training/testing split provided by the original dataset, except for datasets *semeion*, *connect4* and *tdt30* where 70% of data is randomly selected for training and the remaining 30% is used for testing. All the experiments are repeated five times, and both the average results and their standard deviation are reported. All the experiments are run on a laptop with 8GB memory and two 2.50GHz Intel Core i5-2520M CPUs.

3.2.1 Parameter Setting

The parameter L in the loss function (3.3) is set to be 3 according to the suggestion in [122]. The number of triplet constraints N is set to be $100,000$ for all the datasets except for two small datasets *semeion* and *dna* where $N = 20n$. To construct triplet constraints, we follow the active sampling strategy given in [114]: at each iteration t, we first randomly pick a

training example $\mathbf{x}_i^t$, and then $\mathbf{x}_j^t$ from the 3 positive nearest neighbors [2] of $\mathbf{x}_i^t$; we then randomly select a triplet constraint from the set of active constraints [3] involving $\mathbf{x}_i^t$ and $\mathbf{x}_j^t$. We note that this is different from [24], where triplet constraints are selected completely randomly. Our empirical study shows that the active sampling strategy is more effective than choosing triplet constraints completely randomly. This is because the actively sampled constraints are more informative to the learned distance metric than a completely random choice. Furthermore, to verify that the choice of N does not lead to the overfitting of training data, particularly for the two small datasets, in Fig. 3.1, we show the training and test errors for dataset *dna*. It is clear that both errors decline over epoches, suggesting that no overfitting is found even for the small dataset.

For Mini-SGD and the hybrid approaches, we set $b = 10$ for the size of mini-batch as in [98], leading to a total of $T = 10,000$ iterations for these approaches. We evaluate the learned distance metric by the classification error of a k-NN on the test data, where the number of nearest neighbors k is set to be 3 based on our experience.

Parameter R in the proposed algorithms determines the domain size for the distance metric to be learned. We observe that the classification error of k-NN remains almost unchanged when varying R in the range of $\{100, 1000, 10000\}$. We thus set $R = 1,000$ for all the experiments. Another important parameter used by the proposed algorithms is the step size η. We evaluate the impact of step size η by measuring the classification error of a k-NN algorithm that uses the distance metric learned by the Mini-SGD algorithm with $\eta = \{0.1, 1, 10\}$. We observe that $\eta = 1$ yields a low classification error for almost all datasets. We thus fix $\eta = 1$ for the proposed algorithms in all the experiments.

[2] $\mathbf{x}_j$ is a positive nearest neighbor of $\mathbf{x}_i$ if $\mathbf{x}_j$ and $\mathbf{x}_i$ share the same class assignment.

[3] A constraint is active if its hinge loss based on the Euclidean distance is non-zero.

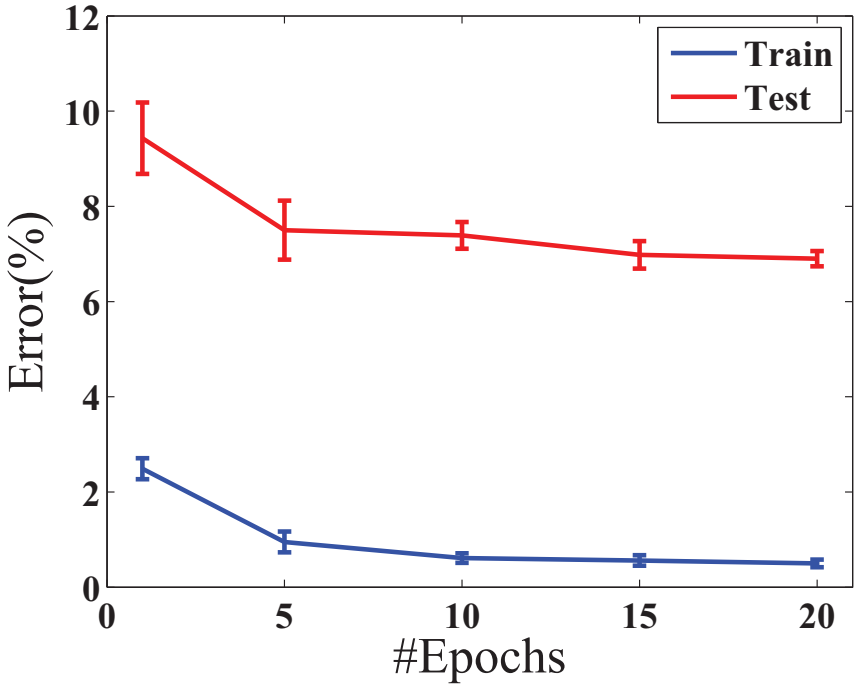


Figure 3.1: The training and testing errors over epoches for dataset *dna*.

3.2.2 Experiment (I): Effectiveness of the Proposed SGD Algorithms for DML

In this experiment, we compare the performance of the proposed SGD algorithms for DML, i.e., Mini-SGD, AS-SGD and two hybrid approaches (HR-SGD and HA-SGD), to the full version of SGD for DML (SGD). We also include Euclidean distance as the reference method in our comparison. Table 3.2 shows the classification error of k-NN ($k = 3$) using the proposed DML algorithms and the baseline algorithms, respectively. First, it is not surprising to observe that all the distance metric learning algorithms improve the classification performance of k-NN compared to the Euclidean distance. Second, for almost all datasets, we observe that all the proposed DML algorithms (i.e., Mini-SGD, AS-SGD, HR-SGD, and HA-SGD) yield similar classification performance as SGD, the full version of SGD algorithm for DML. This result confirms that the proposed SGD algorithms are effective for DML despite the

modifications we made to the SGD algorithm.

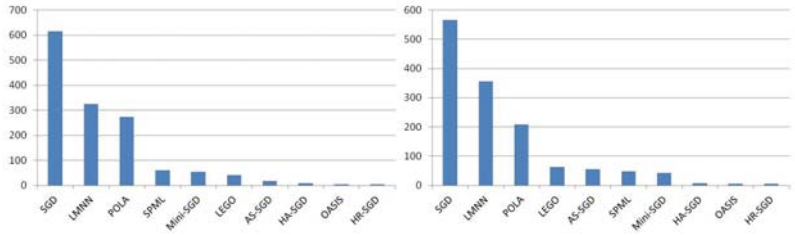
Table 3.2: Classification error (%) of k-NN ($k = 3$) using the distance metrics learned by the proposed SGD methods for DML. Standard deviation computed from five trials is included in the parenthesis.

	Euclid	SGD	Mini-SGD	AS-SGD	HR-SGD	HA-SGD
semeion	8.79	4.39(0.30)	4.60(0.53)	4.23(0.60)	4.27(0.41)	4.18(0.26)
dna	20.71	6.90(0.16)	6.64(0.33)	7.15(0.42)	6.80(0.21)	6.86(0.15)
isolet	8.98	5.98(0.15)	4.23(0.19)	6.09(0.13)	4.59(0.30)	4.77(0.17)
tdt30	5.96	4.51(0.07)	3.52(0.08)	4.53(0.06)	3.70(0.20)	3.65(0.09)
letter	4.42	2.26(0.09)	2.54(0.06)	2.25(0.10)	2.31(0.08)	2.23(0.07)
protein	49.95	39.46(0.42)	38.16(0.24)	39.49(0.51)	40.76(0.20)	40.03(0.30)
connect4	29.48	20.16(0.08)	20.20(0.08)	20.22(0.12)	21.45(0.71)	20.41(0.14)
sensit	27.28	23.62(0.04)	22.95(0.07)	23.70(0.06)	23.39(0.20)	23.33(0.18)
rcv20	9.13	7.76(0.16)	8.42(0.04)	7.74(0.11)	8.40(0.04)	8.37(0.02)
poker	37.98	35.89(0.06)	35.22(0.18)	35.87(0.08)	35.74(0.41)	35.66(0.16)

3.2.3 Experiment (II): Efficiency of the Proposed SGD Algorithms for DML

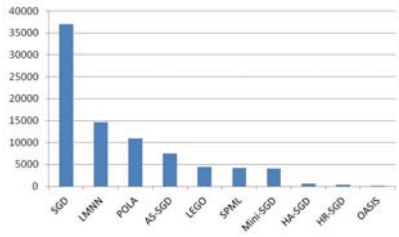
Fig. 3.2 summarizes the running time for the proposed DML algorithms and the baseline SGD algorithm. We note that the running times in Fig. 3.2 do not take into account the time for constructing triplet constraints since it is shared by all the methods in comparison.

It is not surprising to observe that all the proposed SGD algorithms, including Mini-SGD, AS-SGD, HA-SGD and HR-SGD, significantly reduce the running time of SGD. For instance, for dataset *isolet*, it takes SGD more than $35,000$ seconds to learn a distance metric, while the running time is reduced to less than $4,000$ seconds when applying the proposed SGD algorithms, roughly a factor of 10 reduction in running time. Comparing the running time of AS-SGD to that of Mini-SGD, we observe that each method has its own advantage: AS-SGD is more efficient on dataset *semeion* while Mini-SGD is more efficient on the other datasets. This is because different mechanisms are employed by AS-SGD and Mini-SGD to reduce the

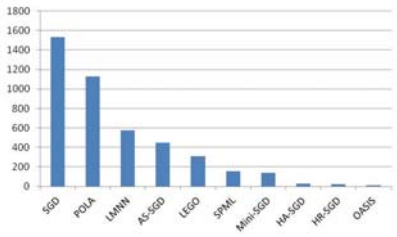


(a) *semeion*

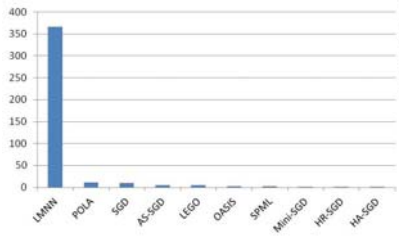
(b) *dna*



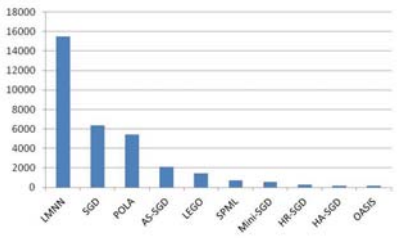
(c) *isolet*



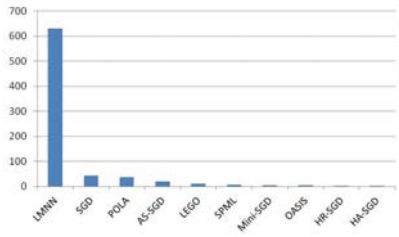
(d) *tdt30*



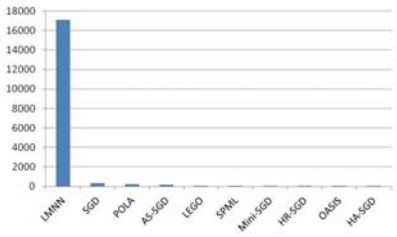
(e) *letter*



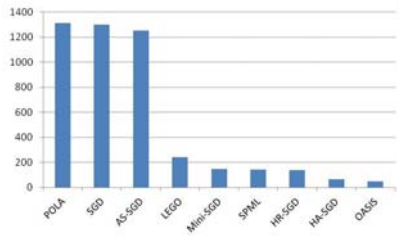
(f) *protein*



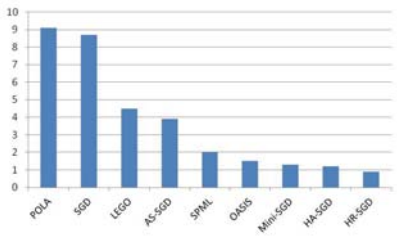
(g) *connect4*



(h) *sensit*



(i) *rcv20*



(j) *poker*

Figure 3.2: The comparison of running time (seconds) for various SGD methods. Note that LMNN, a batch DML algorithm, is mainly implemented in C, which is computationally more efficient than our Matlab implementation. All the other methods are implemented in Matlab.

computational cost: AS-SGD improves the computational efficiency of DML by skipping the constraints that are easy to be classified, while Mini-SGD improves the the computational efficiency of SGD by performing the updating of distance metric once for multiple triplet constraints. Finally, we observe that the two hybrid approaches that combine the strength of both adaptive sampling and mini-batch SGD, are computationally most efficient for almost all datasets. We also observe that HR-SGD appears to be more efficient than HA-SGD on six datasets and only loses it edge on datasets *protein* and *rcv20*. This is because HR-SGD computes the sampling probability γ_t based on one randomly sampled triplet while HA-SGD needs to compute the average derivative for each mini-batch of triplet constraints for the sampling probability.

To further examine the computational efficiency of proposed SGD algorithms for DML, we summarize in Fig. 3.3 the number of updates performed by the proposed SGD algorithms and the baseline SGD algorithm, respectively. We observe that all the proposed SGD algorithms for DML are able to reduce the number of updates significantly compared to SGD. Comparing Mini-SGD to AS-SGD, we observe that for *semeion*, the number of updates performed by AS-SGD is significantly less than Mini-SGD, while it is the other way around for the other datasets. This is again due to the fact that AS-SGD and Mini-SGD deploy different mechanisms for reducing computational costs. As we expect, the two hybrid approaches are able to further reduce the number of updates performed by AS-SGD and Mini-SGD, making them more efficient algorithms for DML.

By comparing the running time in Fig. 3.2 to the number of updates in Fig. 3.3, we observe that a small number of updates does NOT always guarantee a short running time. This is exhibited by the comparison between the two hybrid approaches: although HA-SGD performs the similar number of updates as HR-SGD on datasets *semeion* and *dna*, it

takes HA-SGD significantly longer time to finish the computation than HR-SGD. This is also exhibited by comparing the results across different datasets for a fixed method. For example, for the HA-SGD method, the number of updates for the *protein* dataset is nearly the same as that for the *poker* dataset, but the running time for the *protein* dataset is about 100 times longer than that for the *poker* dataset. This result may sound counter intuitive at the first glance. But, a more careful analysis reveals that in addition to the number of updates, the running time of DML is also affected by the computational cost per iteration, which explains the consistency between Fig. 3.2 and Fig. 3.3. In the case of comparing the two hybrid approaches, we observe that HA-SGD is subjected to a higher computational cost per iteration than HR-SGD because HA-SGD has to compute the norm of the *average* gradient over each mini-batch while HR-SGD only needs to compute the derivative of *one* randomly sampled triplet constraint for each mini-batch. In the case of comparing the running time across different datasets, the *protein* dataset has a significantly higher dimensionality than the *poker* dataset, and therefore is subjected to a higher computational cost per iteration because the computational cost of projecting an updated distance metric onto the PSD cone increases at least quadratically in the dimensionality.

3.2.4 Experiment (III): Comparison with State-of-the-art Online DML Methods

We compare the proposed SGD algorithms to three state-of-the-art online algorithms and one bath method for DML:

- **SPML** [98]: an online learning algorithm for DML that is based on mini-batch SGD and the hinge loss,

- **OASIS** [24]: a state-of-the-art online DML algorithm and symmetric version with only one projection is applied,
- **LEGO** [66]: an online version of the information theoretic based DML algorithm [32].
- **POLA** [96]: a Perception based online DML algorithm.

Finally, for sanity checking, we also compare the proposed SGD algorithms to **LMNN** [114], a state-of-the-art batch learning algorithm for DML.

Both SPML and OASIS use the same set of triplet constraints to learn a distance metric as the proposed SGD algorithms. Since LEGO and POLA are designed for pairwise constraints, for fair comparison, we generate pairwise constraints for LEGO and POLA by splitting each triplet constraint $(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t)$ into two pairwise constraints: a must-link constraint $(\mathbf{x}_i^t, \mathbf{x}_j^t)$ and a cannot-link constraint $(\mathbf{x}_i^t, \mathbf{x}_k^t)$. This splitting operation results in a total of $2N$ pairwise constraints for LEGO and POLA. Finally, we note that since LMNN is a batch learning method, it is allowed to utilize *any* triplet constraint derived from the data, and is not restricted to the set of triplet constraints we generate for the SGD methods. All the baseline DML algorithms are implemented by using the codes from the original authors except for SPML, for which we made appropriate changes to the original code in order to avoid large matrix multiplication and improve the computational efficiency. SPML, OASIS LEGO and POLA are implemented in Matlab, while the core parts of LMNN are implemented by C that is usually deemed to be more efficient than Matlab. The default parameters suggested by the original authors are used in the baseline algorithms. The step size of LEGO is set to be 1, as it was observed in [24] that the prediction performance of LEGO is in general insensitive to the step size. In all experiments, all the baseline methods set the initial solution for distance metric to be an identity matrix.

Table 3.3: Classification error (%) of k-NN ($k = 3$) using the distance metrics learned by baseline SGD method, online learning algorithms and batch learning approach for DML. Standard deviation computed from five trials is included in the parenthesis.

	Baseline	Batch	Online Learning			
	SGD*	LMNN	POLA	LEGO	OASIS	SPML
semeion	4.18	7.11(0.39)	19.25(1.95)	12.89(1.84)	6.74(0.34)	4.81(0.59)
dna	6.64	4.89(0.29)	7.32(0.55)	7.39(0.55)	11.75(0.43)	6.78(0.58)
isolet	4.23	4.11(0.08)	5.18(0.38)	18.08(6.98)	4.37(0.26)	4.36(0.18)
tdt30	3.52	2.80(0.0)	5.93(0.38)	21.11(3.68)	3.92(0.08)	3.47(0.13)
letter	2.23	3.20(0.00)	3.10(0.22)	5.24(0.45)	3.92(0.05)	3.98(0.53)
protein	38.16	39.86(0.16)	38.38(0.56)	42.60(1.13)	37.83(0.23)	40.12(0.53)
connect4	20.16	21.60(0.26)	25.67(0.85)	26.06(1.30)	22.37(0.63)	24.60(0.70)
sensit	22.95	24.45(0.02)	27.51(0.39)	26.50(1.37)	22.12(0.24)	23.48(0.25)
rcv20	7.74	N/A	7.96(0.08)	8.49(0.18)	8.08(0.06)	8.61(0.12)
poker	35.22	N/A	41.26(1.70)	40.58(1.23)	45.12(2.14)	39.42(0.71)

Table 3.3 summarizes the classification results of k-NN ($k = 3$) using the distance metrics learned by the proposed method and by baseline algorithms, respectively. **SGD*** denotes the best result of propose methods in Table 3.2. First, we observe that LEGO and POLA perform significantly worse than the proposed DML algorithms for four datasets, including *semeion*, *connect4*, *sensit* and *poker*. LEGO also performs poorly on *isolet* and *tdt30*. This can be explained by the fact that LEGO and POLA use pairwise constraints for DML while the other methods in comparison use triplet constraints for DML. According to [24, 98, 114], triplet constraints are in general more effective than pairwise constraints. Second, although both SPML and Mini-SGD are based on the mini-batch strategy, SPML performs significantly worse than Mini-SGD on three datasets, i.e. *protein*, *connect4*, and *poker*. The performance difference between SPML and Mini-SGD can be explained by the fact that Mini-SGD uses a smooth loss function while a hinge loss is used by SPML. According to our analysis and the analysis in [28], using a smooth loss function is critical for the success of the mini-batch strategy. Third, OASIS yields similar performance as the proposed algorithms for almost all datasets except for datasets *semeion*, *dna* and *poker*, for which OASIS performs significantly

worse. Overall, we conclude that the proposed DML algorithms yield similar, if not better, performance as the state-of-the-art online learning algorithms for DML.

Compared to LMNN, a state-of-the-art batch learning algorithm for DML, we observe that the proposed SGD algorithms yield similar performance on four datasets. They however perform significantly better than LMNN on datasets *semeion* and *letter*, and significantly worse on datasets *dna* and *tdt30*. We attribute the difference in classification error to the fact that the proposed DML algorithms are restricted to $100,000$ randomly sampled triplet constraints while LMNN is allowed to use *all* the triplet constraints that can be derived from the data. The restriction in triplet constraints could sometimes limit the classification performance but at the other time help avoid the overfitting problem. We also observe that LMNN is unable to run on the two large datasets *rcv20* and *poker*, indicating that LMNN does not scale well to the size of datasets.

Table 3.4: The comparison of running time (seconds) for OASIS and the hybrid methods. Average results over five trials are reported.

Methods	*semeion*	*dna*	*isolet*	*tdt30*	*letter*
OASIS	4.4	5.5	156.6	10.4	2.3
HR-SGD	3.6	5.1	363.2	21.7	1.4
HA-SGD	7.6	8.1	597.9	28.4	1.1
Methods	*protein*	*connect4*	*sensit*	*rcv20*	*poker*
OASIS	161.4	4.5	19.0	46.5	1.5
HR-SGD	275.7	3.0	23.5	139.2	0.9
HA-SGD	164.6	2.5	15.3	65.6	1.2

The running times for the proposed algorithms and the baseline algorithms are summarized in Fig. 3.2. The number of updates for both groups of algorithms are provided in Fig. 3.3. It is not surprising to observe that two online DML algorithms (SPML, OASIS) are significantly more efficient than SGD in terms of both running time and the number of updates. We also observe that Mini-SGD and SPML share the same number of updates and

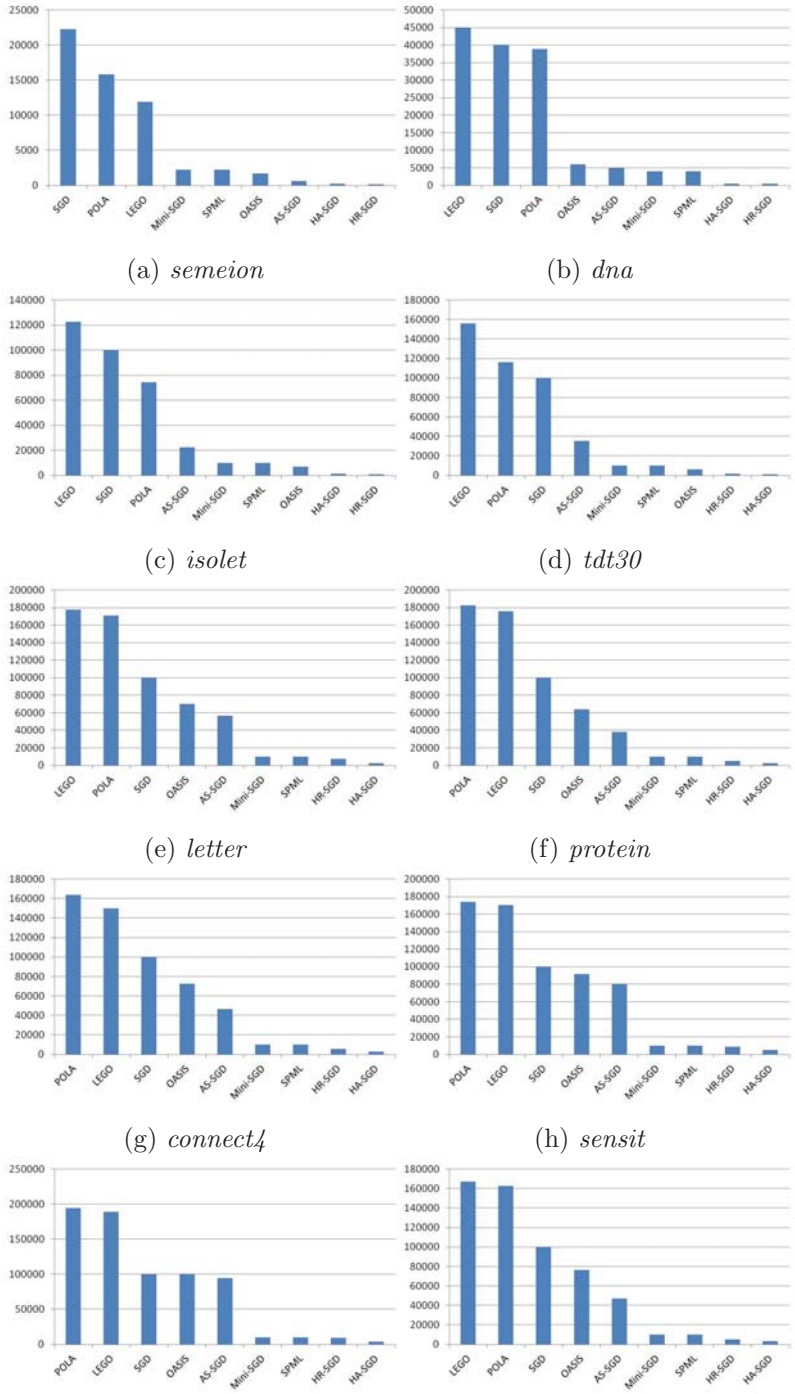


Figure 3.3: The comparison of number of updates for various SGD methods. Note that since POLA and LEGO optimize pairwise constraints, we decompose each triplet constraint into two pairwise constraints for these two methods. As a result, the number of constraints is doubled for these two methods.

similar running time for all datasets because they use the same mini-batch strategy. Furthermore, compared to three online DML algorithms (SPML, LEGO and POLA), the two hybrid approaches are significantly more efficient in both running time and the number of updates. Table 3.4 compares the detailed running time of OASIS and the hybrid methods. We also observe that the hybrid methods are more efficient than OASIS on six datasets (i.e., *semeion*, *dna*, *letter*, *connect4*, *sensit* and *poker*), even though OASIS only performs projection once at the end of the program. We attribute the efficiency of the hybrid approaches to the reduced number of updates they have to perform on the learned metric. Finally, since LMNN is implemented by C, it is not surprising to observe that LMNN shares similar running time as the other online DML algorithms for relatively small datasets. It is however significantly less efficient than the online learning algorithms for datasets of modest size (e.g. *letter*, *connect4* and *sensit*), and becomes computationally infeasible for the two large datasets *rcv20* and *poker*. Overall, we observe that the two hybrid approaches are significantly more efficient than the other DML algorithms in comparison.

3.3 Conclusions

In this chapter, we propose two strategies to improve the computational efficiency of SGD for DML, i.e. mini-batch and adaptive sampling. The key idea of mini-batch is to group multiple triplet constraints into a mini-batch, and only update the distance metric once for each mini-batch; the key idea of adaptive sampling is to perform stochastic updating by giving a difficult triplet constraint more chance to be used for updating the distance metric than an easy triplet constraint. We develop theoretical guarantees for both strategies. We also develop two variants of hybrid approaches that combine mini-batch with adaptive

sampling for more efficient DML. Our empirical study confirms that the proposed algorithms yield similar, if not better, prediction performance as the state-of-the-art online learning algorithms for DML but with significantly less amount of running time. In the future, we could analysis the theoretical guarantee for the hybrid methods.

Chapter 4

Distance Metric Learning for High Dimensional Data

In this chapter, we will address the challenge from large number of variables ($\mathcal{O}(d^2)$). We follow the one-projection paradigm as in Section 2.1 and focus on optimizing $\mathcal{O}(d^2)$ variables. Nowadays, high dimensional representations become more and more popular in various applications, especially for images and video as examples shown in Table 4.1. In contrast, DML is only available for low dimensional data (e.g., some hundred features) due to the involvement of d^2 variables. Some studies try to alleviate the issue as described in Section 2.4, but all of them have to hold some strong assumptions.

Table 4.1: Examples of applications with high dimensional features.

Applications	#Features
Fave Verification [25]	100,000
Image Classification [90]	250,000
Video Event Detection [87]	500,000

The straightforward way to solve high dimensional DML problem is reducing the dimensionality of data using dimensionality reduction methods such as principal component analysis (PCA) [114] or random projection (RP) [103]. Although RP is computationally more efficient than PCA, it often yields significantly worse performance than PCA unless the number of random projections is sufficiently large [39]. We note that RP has been successfully applied to many machine learning tasks, e.g., classification [91], clustering [11] and

regression [81], however, only a few studies examined the application of RP to DML, and most of them with limited success.

In this chapter, we propose a *dual random projection* approach for high dimensional DML. Our approach, on one hand, enjoys the light computation of random projection, and on the other hand, significantly improves the effectiveness of random projection. The main limitation of using random projection for DML is that all the columns/rows of the learned metric will lie in the subspace spanned by the random vectors. We address this limitation of random projection by

- first estimating the *dual* variables based on the random projected vectors and,
- then reconstructing the distance metric using the estimated dual variables and data vectors in the original space.

Since the final distance metric is computed using the original vectors, not the randomly projected vectors, the column/row space of the learned metric will NOT be restricted to the subspace spanned by the random projection, thus alleviating the limitation of random projection. We verify the effectiveness of the proposed algorithms both empirically and theoretically.

We finally note that our work is built upon the recent work [120] on random projection where a dual random projection algorithm is developed for linear classification. Our work differs from [120] in that we apply the theory of dual random projection to DML. More importantly, we have made an important progress in advancing the theory of dual random projection. Unlike the theory in [120] where the data matrix is assumed to be low rank or approximately low rank, our new theory of dual random projection is applicable to any data matrix even when it is NOT approximately low rank. This new analysis significantly broadens the application domains where dual random projection is applicable, which is

further verified by our empirical study.

The rest of the chapter is organized as follows: Section 4.1 describes the proposed dual random projection approach for DML and the detailed algorithm for solving the dual problem in the subspace spanned by random projection. Section 4.2 summarizes the results of the empirical study, and Section 4.3 concludes this work.

4.1 Dual Random Projection for Distance Metric Learning

Let $\mathbf{X} = (\mathbf{x}_1, \cdots, \mathbf{x}_n) \in \mathbb{R}^{d \times n}$ denote the collection of training examples. Given a PSD matrix M, the distance between two examples $\mathbf{x}_i$ and $\mathbf{x}_j$ is given as

$$\text{dist}_M(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^\top M(\mathbf{x}_i - \mathbf{x}_j).$$

The proposed framework for DML will be based on triplet constraints, not pairwise constraints. This is because several previous studies have suggested that triplet constraints are more effective than pairwise constraints [114, 24, 98]. Let $\mathcal{D} = \{(\mathbf{x}_i^1, \mathbf{x}_j^1, \mathbf{x}_k^1), \ldots, (\mathbf{x}_i^N, \mathbf{x}_j^N, \mathbf{x}_k^N)\}$ be the set of triplet constraints used for training, where $\mathbf{x}_i^t$ is expected to be more similar to $\mathbf{x}_j^t$ than to $\mathbf{x}_k^t$. Our goal is to learn a metric function M that is consistent with most of the triplet constraints in $\mathcal{D}$, i.e.

$$\forall(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t) \in \mathcal{D}, \quad (\mathbf{x}_i^t - \mathbf{x}_j^t)^\top M(\mathbf{x}_i^t - \mathbf{x}_j^t) + 1 \leq (\mathbf{x}_i^t - \mathbf{x}_k^t)^\top M(\mathbf{x}_i^t - \mathbf{x}_k^t)$$

Following the empirical risk minimization framework, we cast the triplet constraints based DML into the following optimization problem:

$$\min_{M \in S_d} \frac{\lambda}{2}\|M\|_F^2 + \frac{1}{N}\sum_{t=1}^{N} \ell(\langle M, A_t \rangle) \tag{4.1}$$

where S_d stands for the symmetric matrix of size $d \times d$, $\lambda > 0$ is the regularization parameter, $\ell(\cdot)$ is a convex loss function, $A_t = (\mathbf{x}_i^t - \mathbf{x}_k^t)(\mathbf{x}_i^t - \mathbf{x}_k^t)^\top - (\mathbf{x}_i^t - \mathbf{x}_j^t)(\mathbf{x}_i^t - \mathbf{x}_j^t)^\top$, and $\langle \cdot, \cdot \rangle$ stands for the dot product between two matrices. We note that we did not enforce M in (4.1) to be PSD because we follow the one-projection paradigm proposed in [24] that first learns a symmetric matrix M by solving the optimization problem in (4.1) and then projects the learned matrix M onto the PSD cone. We emphasize that unlike [120], we did not assume $\ell(\cdot)$ to be smooth, making it possible to apply the proposed approach to the hinge loss.

Let $\ell_*(\cdot)$ be the convex conjugate of $\ell(\cdot)$. The dual problem of (4.1) is given by

$$\max_{\alpha_1, \ldots, \alpha_N} -\frac{1}{N}\sum_{t=1}^{N} \ell_*(\alpha_t) - \frac{1}{2\lambda N^2} \left\| \sum_{t=1}^{N} \alpha_t A_t \right\|_F^2$$

which is equivalent to

$$\max_{\boldsymbol{\alpha} \in [-1,0]^N} -\sum_{t=1}^{N} \ell_*(\alpha_t) - \frac{1}{2\lambda N} \boldsymbol{\alpha}^\top G \boldsymbol{\alpha} \tag{4.2}$$

where $\boldsymbol{\alpha} = (\alpha_1, \cdots, \alpha_N)^\top$ and $G = [G_{a,b}]_{N \times N}$ is a matrix of $N \times N$ with $G_{a,b} = \langle A_a, A_b \rangle$. We denote by $M_* \in \mathbb{R}^{d \times d}$ the optimal primal solution to (4.1), and by $\boldsymbol{\alpha}_* \in \mathbb{R}^N$ the optimal

dual solution to (4.2). Using the first order condition for optimality, we have

$$M_* = -\frac{1}{\lambda N}\sum_{t=1}^{N}\alpha_*^t A_t \tag{4.3}$$

4.1.1 Dual Random Projection for Distance Metric Learning

Directly solving the primal problem in (4.1) or the dual problem in (4.2) could be computational expensive when the data is of high dimension and the number of training triplets is very large. We address this challenge by inducing a random matrix $R \in \mathbb{R}^{d\times m}$, where $m \ll d$ and $R_{i,j} \sim \mathcal{N}(0, 1/m)$, and projecting all the data points into the low dimensional space using the random matrix, i.e., $\widehat{\mathbf{x}}_i = R^\top \mathbf{x}_i$. As a result, A_t, after random projection, becomes $\widehat{A}_t = R^\top A_t R$.

A typical approach of using random projection for DML is to obtain a matrix M_s of size $m \times m$ by solving the primal problem with the randomly projected vectors $\{\widehat{\mathbf{x}}_i\}_{i=1}^n$, i.e.

$$\min_{M\in S_m} \frac{\lambda}{2}\|M\|_F^2 + \frac{1}{N}\sum_{t=1}^{N}\ell(\langle M, \widehat{A}_t\rangle) \tag{4.4}$$

Given the learned metric M_s, for any two data points $\mathbf{x}$ and $\mathbf{x}'$, their distance is measured by $(\mathbf{x}-\mathbf{x}')^\top R M_s R^\top (\mathbf{x}-\mathbf{x}') = (\mathbf{x}-\mathbf{x}')^\top M'(\mathbf{x}-\mathbf{x}')$, where $M' = RM_sR^\top \in \mathbb{R}^{d\times d}$ is the effective metric in the original space $\mathbb{R}^d$. The key limitation of this random projection approach is that both the column and row space of M' are restricted to the subspace spanned by vectors in random matrix R.

Instead of solving the primal problem, we proposed to solve the dual problem using the

randomly projected data points $\{\widehat{\mathbf{x}}_i\}_{i=1}^n$, i.e.

$$\max_{\boldsymbol{\alpha}\in\mathbb{R}^N} -\sum_{t=1}^{N} \ell_*(\alpha_t) - \frac{1}{2\lambda N}\boldsymbol{\alpha}^\top \widehat{G}\boldsymbol{\alpha} \tag{4.5}$$

where $\widehat{G}_{a,b} = \langle R^\top A_a R, R^\top A_b R\rangle$. After obtaining the optimal solution $\widehat{\boldsymbol{\alpha}}_*$ for (4.5), we reconstruct the metric by using the dual variables $\widehat{\boldsymbol{\alpha}}_*$ and data matrix X in the original space, i.e.

$$\widehat{M}_* = -\frac{1}{\lambda N}\sum_{t=1}^{N} \widehat{\alpha}_*^t A_t \tag{4.6}$$

It is important to note that unlike the random projection approach, the recovered metric $\widehat{M}_*$ in (4.6) is not restricted by the subspace spanned by the random vectors, a key to the success of the proposed algorithm.

Alg. 4 summarizes the key steps for the proposed dual random projection method for DML. Following one-projection paradigm [24], we project the learned symmetric matrix M onto the PSD cone at the end of the algorithm. The key component of Alg. 4 is to solve the optimization problem in (4.2) at Step 4 accurately. We choose stochastic dual coordinate ascent (SDCA) method for solving the dual problem (4.5) because it enjoys a linear convergence when the loss function is smooth, and is shown empirically to be significantly faster than the other stochastic optimization methods [97]. We use the combination strategy recommended in [97], denoted by **CSDCA**, which uses SGD for the first epoch and then applies SDCA for the rest epochs.

Algorithm 4 Dual Random Projection Method (DuRP) for DML

1: **Input**: the triplet constraints $\mathcal{D}$ and the number of random projections m.
2: Generate a random matrix $R \in \mathbb{R}^{d \times m}$ and $R_{i,j} \sim \mathcal{N}(0, 1/m)$.
3: Project each example as $\widehat{\mathbf{x}} = R^\top \mathbf{x}$.
4: Solve the optimization problem (4.5) and obtain the optimal solution $\widehat{\boldsymbol{\alpha}}_*$
5: Recover the solution in the original space by $\widehat{M}_* = -\frac{1}{\lambda N} \sum_t \widehat{\alpha}_*^t A_t$
6: Output: $\Pi_{PSD}(\widehat{M}_*)$

4.1.2 Main Theoretical Results

First, similar to [120], we consider the case when the data matrix X is of low rank. The theorem below shows that under the low rank assumption, with a high probability, the distance metric recovered by Algorithm 4 is nearly optimal.

Theorem 3. *Let M_* be the optimal solution to (4.1). Let $\widehat{\boldsymbol{\alpha}}_*$ be the optimal solution for (4.5), and let $\widehat{M}_*$ be the solution recovered from $\widehat{\boldsymbol{\alpha}}_*$ using (4.6). Under the assumption that all the data points lie in the subspace of r-dimension, for any $0 < \varepsilon \leq 1/6$, with a probability at least $1 - \delta$, we have*

$$\|\Pi_{PSD}(M_*) - \Pi_{PSD}(\widehat{M}_*)\|_F \leq \frac{3\varepsilon}{1 - 3\varepsilon} \|M_*\|_F$$

provided $m \geq \frac{(r+1)\log(2r/\delta)}{c\varepsilon^2}$ and constant c is at least $1/3$.

The proof of Theorem 3 can be found in appendix. Theorem 3 indicates that if the number of random projections is sufficiently large (i.e. $m = \Omega(r \log r)$), we can recover the optimal solution in the original space with a small error. It is important to note that our analysis, unlike [120], can be applied to non-smooth loss such as the hinge loss.

In the second case, we assume the loss function $\ell(\cdot)$ is γ-smooth (i.e., $|\ell'(z) - \ell'(z')| \leq \gamma |z - z'|$). The theorem below shows that the dual variables obtained by solving the optimization

problem in (4.5) can be close to the optimal dual variables, even when the data matrix X is *NOT* low rank or approximately low rank. For the presentation of theorem, we first define a few important quantities. Define matrices $M^1 \in \mathbb{R}^{N\times N}$, $M^2 \in \mathbb{R}^{N\times N}$, $M^3 \in \mathbb{R}^{N\times N}$, and $M^4 \in \mathbb{R}^{N\times N}$ as

$$\begin{aligned} M^1_{a,b} &= \|(\mathbf{x}_i^a - \mathbf{x}_k^a)\|_2^2 \times \|(\mathbf{x}_i^b - \mathbf{x}_k^b)\|_2^2 \\ M^2_{a,b} &= \|(\mathbf{x}_i^a - \mathbf{x}_j^a)\|_2^2 \times \|(\mathbf{x}_i^b - \mathbf{x}_j^b)\|_2^2 \\ M^3_{a,b} &= \|(\mathbf{x}_i^a - \mathbf{x}_k^a)\|_2^2 \times \|(\mathbf{x}_i^b - \mathbf{x}_j^b)\|_2^2 \\ M^4_{a,b} &= \|(\mathbf{x}_i^a - \mathbf{x}_j^a)\|_2^2 \times \|(\mathbf{x}_i^b - \mathbf{x}_k^b)\|_2^2 \end{aligned}$$

Define κ the maximum of the spectral norm of the four matrices, i.e.

$$\kappa = \max\Big(\|M^1\|_2, \|M^2\|_2, \|M^3\|_2, \|M^4\|_2\Big) \tag{4.7}$$

where $\|\cdot\|_2$ stands for the spectral norm of matrices.

Theorem 4. *Assume $\ell(z)$ is γ-smooth. Let $\boldsymbol{\alpha}_*$ be the optimal solution to the dual problem in (4.2), and let $\widehat{\boldsymbol{\alpha}}_*$ be the approximately optimal solution for (4.5) with suboptimality η. Then, with a probability at least $1-\delta$, we have*

$$\|\boldsymbol{\alpha}_* - \widehat{\boldsymbol{\alpha}}_*\|_2 \leq \max\Big(8\epsilon\gamma\kappa\|\boldsymbol{\alpha}_*\|_2, \sqrt{2\gamma\eta}\Big)$$

where κ is define in (4.7), provided $m \geq \frac{8}{\epsilon^2}\ln\frac{8N}{\delta}$.

The proof of Theorem 4 can be found in the appendix. Unlike Theorem 3 where the data matrix X is assumed to be low rank, Theorem 4 holds without any prior assumption

about the data matrix. It shows that despite the random projection, the dual solution can be recovered approximately using the randomly projected vectors, provided that the number of random projections m is sufficiently large, κ is small, and the approximately optimal solution $\widehat{\boldsymbol{\alpha}}_*$ is sufficiently accurate. In the case when most of the training examples are not linear dependent, we could have $\kappa = \Theta(N/d)$, which could be a modest number when d is very large. The result in Theorem 4 essentially justifies the key idea of our approach, i.e. computing the dual variables first and recovering the distance metric later. Finally, since $\|\boldsymbol{\alpha}_* - \widehat{\boldsymbol{\alpha}}_*\|_2$, the approximation error in the recovered dual variables, is proportional to the square root of the suboptimality η, an accurate solution for (4.5) is needed to ensure a small approximation error. We note that given Theorem 4, it is straightforward to bound $\|M_* - \widehat{M}_*\|_2$ using the relationship between the dual variables and the primal variables in (4.3).

4.2 Experiments

We will first describe the experimental setting, and then present our empirical study for ranking and classification tasks on various datesets.

4.2.1 Experimental Setting

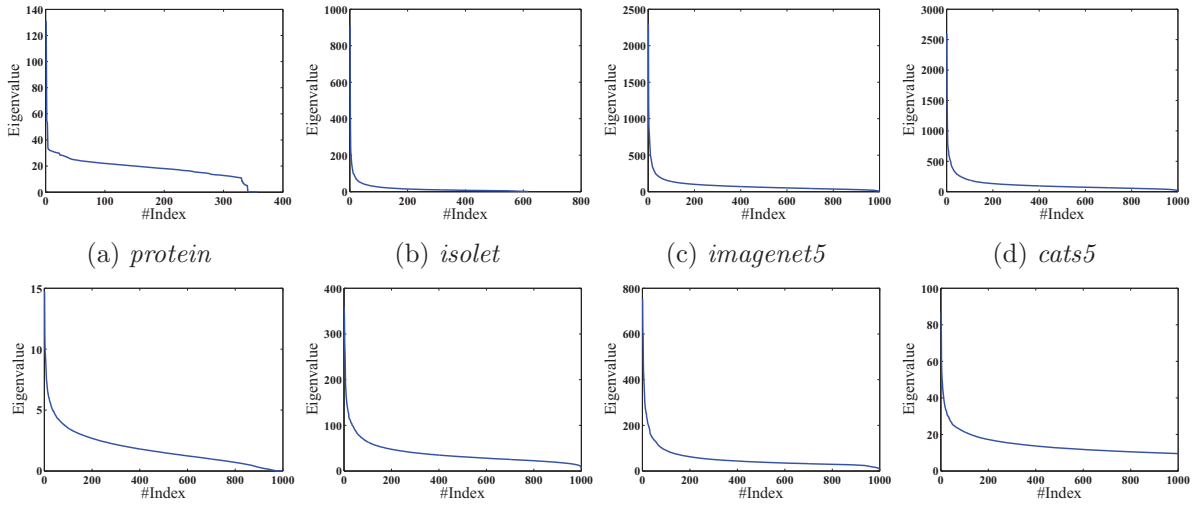
Data sets Eight datasets are used to validate the effectiveness of the proposed algorithm for DML. Table 4.2 summarizes the information of these datasets. *imagenet5* consists of 5 randomly selected classes from ImageNet [92] while *cats5* contains 5 cat species from the same archive. *caltech30* is a subset of Caltech256 image dataset [48] and we use the version pre-processed by [24]. *tdt30* is a subset of tdt2 dataset [14]. Both *caltech30* and *tdt30* are

Table 4.2: Statistics for the datasets used in our empirical study. #C is the number of classes. #F is the number of original features. #Train and #Test represent the number of training data and test data, respectively.

	# C	# F	#Train	#Test
protein	3	357	17,766	6,621
isolet	26	617	6,238	1,559
imagenet5	5	1,000	4,582	1,964
cats5	5	1,000	5,128	2,199
caltech30	30	1,000	5,502	2,355
tdt30	30	1,000	6,575	2,819
20news	20	1,000	15,935	3,993
rcv30	30	1,000	507,585	15,195

comprised of the examples from the 30 most popular categories. All the other datasets are downloaded from LIBSVM [22], where *rcv30* is a subset of the original dataset consisted of documents from the 30 most popular categories. For datasets *tdt30*, *20news* and *rcv30*, they are comprised of documents represented by vectors of $\sim 50,000$ dimensions. Since it is expensive to compute and maintain a matrix of $50,000 \times 50,000$, for these three datasets, we follow the procedure in [24] that maps all documents to a space of $1,000$ dimension. More specifically, we first keep the top $20,000$ most popular words for each collection, and then reduce their dimensionality to $1,000$ by using PCA. We emphasize that for several data sets in our test beds, their data matrices can not be well approximated by low rank matrices. Fig. 4.1 summarizes the eigenvalue distribution of the eight datasets used in our experiment. We observe that four out of these datasets (i.e., *caltech20*, *tdt30*, *20news*, *rcv30*) have a flat eigenvalue distribution, indicating that the associated data matrices can not be well approximated by a low rank matrix. This justifies the importance of removing the low rank assumption from the theory of dual random projection, an important contribution of this work.

For the datasets used in this study, we use the standard training/testing split provided



(a) *protein* (b) *isolet* (c) *imagenet5* (d) *cats5*

(e) *caltech30* (f) *tdt30* (g) *20news* (h) *rcv30*

Figure 4.1: The eigenvalue distribution of datasets used in our empirical study

by the original datasets, except for datasets *imagenet5*, *cats5*, *tdt30*, *caltech30* and *rcv30*. For *imagenet5*, *cats5*, *tdt30* and *caltech30*, we randomly select 70% of the data for training and use the remaining 30% for testing; for *rcv30*, we switch the training and test sets defined by the original package to ensure that the number of training examples is sufficiently large.

Evaluation metrics To measure the quality of learned distance metrics, two types of evaluations are adopted in our study. First, we follow the evaluation protocol in [24] and evaluate the learned metric by its ranking performance. More specifically, we treat each test instance $\mathbf{q}$ as a query, and rank the other test instances in the ascending order of their distance to $\mathbf{q}$ using the learned metric. The mean-average-precision(mAP) given below is used to evaluate the quality of the ranking list

$$mAP = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{r_i}\sum_{j=1}^{r_i}P(\mathbf{x}_{i\sim j})$$

where $|Q|$ is the size of query set, r_i is the number of relevant instances for i-th query and $P(\mathbf{x}_{i\sim j})$ is the precision for the first j ranked instances when the instance ranked at the

j-th position is relevant to the query $\mathbf{q}$. Here, an instance $\mathbf{x}$ is relevant to a query $\mathbf{q}$ if they belong to the same class. Second, we evaluate the learned metric by its classification performance with k-nearest neighbor classifier. More specifically, for each test instance $\mathbf{q}$, we apply the learned metric to find the first k training examples with the shortest distance, and predict the class assignment for k by taking the majority vote among the k nearest neighbors. Finally, we also evaluate the computational efficiency of the proposed algorithm for DML by its efficiency.

Baselines Besides the Euclidean distance that is used as a baseline similarity measure, six state-of-the-art DML methods are compared in our empirical study:

- **DuOri**: This algorithm first applies Combined Stochastic Dual Coordinate Ascent (CSDCA) [97] to solve the dual problem in (4.2) and then computes the distance metric using the learned dual variables.
- **DuRP**: This is the proposed algorithm for DML (i.e. Algorithm 4).
- **SRP**: This algorithm applies random projection to project data into low dimensional space, and then it employs CSDCA to learn the distance metric in this subspace.
- **SPCA**: This algorithm uses PCA as the initial step to reduce the dimensionality, and then applies CSDCA to learn the distance metric in the subspace generated by PCA.
- **OASIS** [24]: A state-of-the-art online learning algorithm for DML that learns the optimal distance metric directly from the original space without any dimensionality reduction.
- **LMNN** [114]: A state-of-the-art batch learning algorithm for DML. It performs the dimensionality reduction using PCA before starting DML.

Implementation details We randomly select $N = 100,000$ active triplets (i.e., incur the positive hinge loss by Euclidean distance) and set the number of epochs to be 3 for all

stochastic methods (i.e., DuOri, DuRP, SRP, SPCA and OASIS), which yields sufficiently accurate solutions in our experiments and is also consistent with the observation in [97]. We search λ in $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$ and fix it as $1/N$ since it is insensitive. The step size of CSDCA is set according to the analysis in [97]. For all stochastic optimization methods, we follow the one-projection paradigm by projecting the learned metric onto the PSD cone. The hinge loss is used in the implementation of the proposed algorithm. Both OASIS and LMNN use the implementation provided by the original authors and parameters are tuned based on the recommendation by the original authors. All methods are implemented in Matlab, except for LMNN, whose core part is implemented in C, which is shown to be more efficient than our Matlab implementation. All stochastic optimization methods are repeated five times and the average result over five trials is reported. All experiments are implemented on a Linux Server with 64GB memory and 12 $\times$ 2.4GHz CPUs and only single thread is permitted for each experiment.

4.2.2 Efficiency of the Proposed Method

Table 4.3: CPUtime (minutes) for different methods for DML. All algorithms are implemented in Matlab except for LMNN whose core part is implemented in C and is more efficient than our Matlab implementation.

	Metric in Original Space			Metric in Subspace		
	DuOri	DuRP	OASIS	SRP	SPCA	LMNN
protein	214.5±5.8	0.6±0.0	81.9±3.3	0.2±0.0	0.2±0.0	488.9
isolet	198.4±7.2	0.6±0.0	12.6±0.2	0.1±0.0	0.1±0.0	384.0
imagenet5	776.3±31.2	1.2±0.0	68.3±1.3	0.1±0.0	0.2±0.0	1,374.1
cats5	782.3±70.9	1.1±0.0	74.1±1.1	0.1±0.0	0.3±0.1	1,467.0
caltech30	1,214.5±229.5	1.3±0.0	640.4±121.2	0.2±0.0	0.5±0.0	2,197.9
tdt30	1,029.9±16.8	0.8±0.0	140.8±4.5	0.2±0.0	0.4±0.0	624.2
20news	1,212.9±154.3	1.0±0.0	216.3±48.8	0.2±0.0	0.5±0.0	1,893.6
rcv30	1,121.3±79.4	1.3±0.0	432.5±7.7	0.2±0.0	4.2±0.0	N/A

In this experiment, we set the number of random projection to be 10, which according to experimental results in Section 4.2.3 and 4.2.4, yields almost the optimal performance for the proposed algorithm. For fair comparison, the number of reduced dimension is also set to be 10 for LMNN.

Table. 4.3 compares the CPUtime (in minutes) of different methods. Notice that the time of sampling triplets is not taken into account as it is consumed by all the methods, and all the other operators (e.g., random projection and PCA) are included. It is not surprising to observe that DuRP, SRP and SPCA have similar CPUtimes, and are significantly more efficient than the other methods due to the effect of dimensionality reduction. Since DuRP and SRP share the same procedure for computing the dual variables in the subspace, the only difference between them lies in the procedure for reconstructing the distance metric from the estimated dual variables, a computational overhead that makes DuRP slightly slower than SRP. For all datasets, we observe that DuRP is at least 200 times faster than DuOri and 20 times faster than OASIS. Compared to the stochastic optimization methods, LMNN is the least efficient on six datasets (i.e., *protein*, *isolet*, *imagenet5*, *cats5*, *caltech30* and *20news*), mostly due to the fact that it is a batch learning algorithm.

4.2.3 Evaluation by Ranking

In first experiment, we set the number of random projections used by SRP, SPCA and the proposed DuRP algorithm to be 10, which is roughly 1% of the dimensionality of the original space. For fair comparison, the number of reduced dimension for LMNN is also set to be 10. We measure the quality of learned metrics by its ranking performance using the metric of mAP.

Table. 4.4 summarizes the performance of different methods for DML. First, we observe

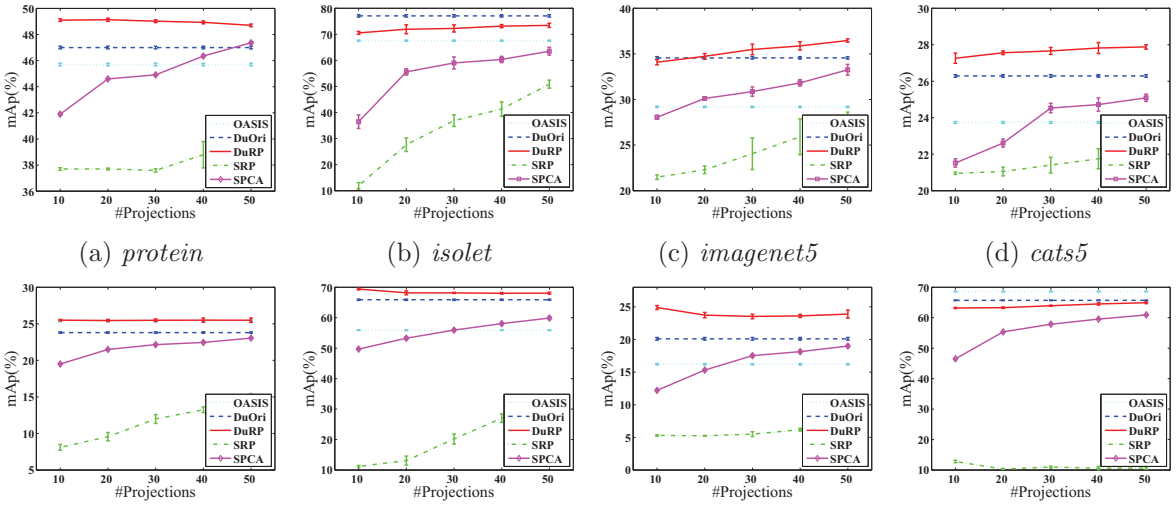
Table 4.4: Comparison of ranking results measured by mAP (%) for different metric learning algorithms.

	Metric in Original Space				Metric in Subspace Metric		
	Euclid	DuOri	DuRP	OASIS	SRP	SPCA	LMNN
protein	39.0	47.0±0.1	49.1±0.1	45.7±0.1	37.7±0.1	41.9±0.1	41.9
isolet	46.7	77.1±0.5	70.5±0.6	67.6±0.2	11.9±1.2	36.5±2.6	68.7
imagenet5	25.4	34.6±0.1	34.1±0.3	29.2±0.1	21.5±0.3	28.1±0.2	31.9
cats5	22.5	26.3±0.1	27.3±0.3	23.7±0.1	21.0±0.1	21.5±0.2	24.8
caltech30	16.4	23.8±0.1	25.5±0.1	25.4±0.2	8.1±0.4	19.5±0.0	16.3
tdt30	36.8	65.9±0.2	69.4±0.3	55.9±0.1	11.2±0.3	49.7±0.2	66.4
20news	8.4	20.1±0.2	24.9±0.3	16.2±0.1	5.3±0.1	12.2±0.1	22.5
rcv30	16.7	65.7±0.1	63.2±0.2	68.6±0.1	12.8±0.4	46.5±0.0	N/A

that DuRP significantly outperforms SRP and SPCA for all datasets. In fact, SRP is worse than Euclidean distance which computes the distance in the original space. SPCA is only able to perform better than the Euclidean distance, and is outperformed by all the other DML algorithms. Second, we observe that for all the datasets, DuRP yields similar performance as DuOri. The only difference between DuRP and DuOri is that DuOri solves the dual problem without using random projection. The comparison between DuRP and DuOri indicates that the random projection step has minimal impact on the learned distance metric, justifying the design of the proposed algorithm. Third, compared to OASIS, we observe that DuRP performs significantly better on three datasets (i.e., *imagenet5*, *tdt30* and *20news*) and has the comparable performance on the other datasets. Finally, we observe that for all datasets, the proposed DuRP method significantly outperforms LMNN, a state-of-the-art batch learning algorithm for DML. We also note that because of limited memory, we are unable to run LMNN on datasets *rcv30*.

In the second experiment, we vary the number of random projections from 10 to 50. All stochastic methods are run with five trails and Fig. 4.2 reports the average results with standard deviation. Note that the performance of OASIS and DuOri remain unchanged with

varied number of projections because they do not use projection. It is surprising to observe that DuRP almost achieves its best performance with only 10 projections for all datasets. This is in contrast to SRP and SPCA, whose performance usually improves with increasing number of projections. We also observe that DuRP outperforms DuOri for several datasets (i.e. *protein*, *imagenet5*, *cats5*, *caltech30*, *tdt30* and *20news*). We suspect that the better performance of DuRP is because of the implicit regularization due to the random projection. We plan to investigate more about the regularization capability of random projection in the future. We finally point out that with sufficiently large number of projections, SPCA is able to outperform OASIS on 5 datasets (i.e., *protein*, *imagenet5*, *cats5*, *tdt30* and *20news*), indicating that the comparison result may be sensitive to the number of projections.



(a) *protein* (b) *isolet* (c) *imagenet5* (d) *cats5*

(e) *caltech30* (f) *tdt30* (g) *20news* (h) *rcv30*

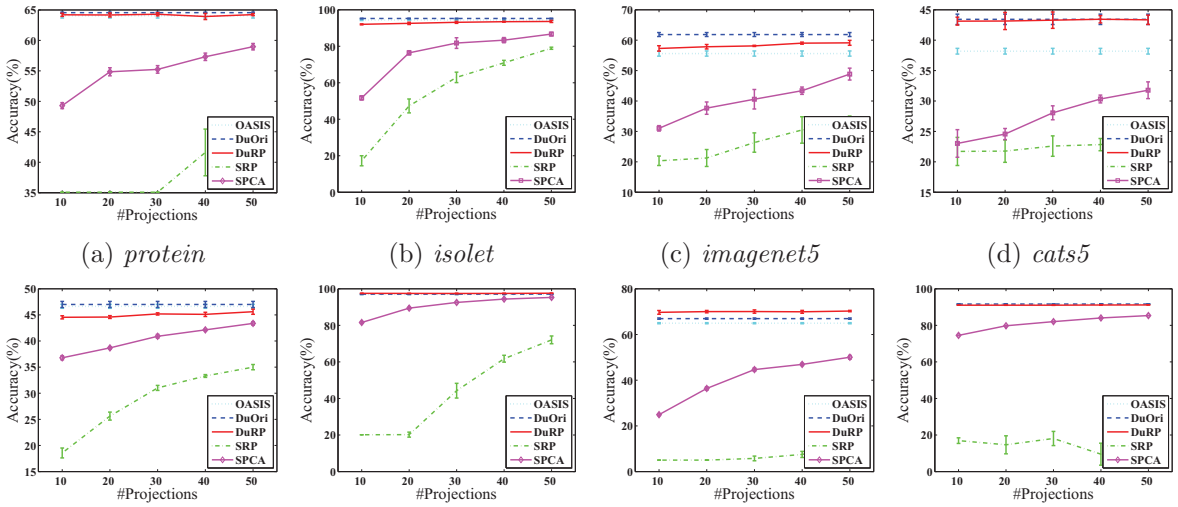
Figure 4.2: The comparison of different stochastic algorithms for ranking

4.2.4 Evaluation by Classification

In this experiment, we evaluate the learned metric by its classification accuracy with k-NN ($k = 5$) classifier. We emphasize that the purpose of this experiment is to evaluate the

metrics learned by different DML algorithms, not to demonstrate that the learned metric will result in the state-of-art classification performance[1]. Similar to the evaluation by ranking, all experiments are run five times and the results averaged over five trials with standard deviation are reported in Fig. 4.3. We essentially have the same observation as that for the ranking experiments reported in Section 4.2.3 except that for most datasets, the three methods DuRP, DuOri, and OASIS yield very similar performance.

Note the main concern of this chapter is time efficiency and the size of learned metric is $d \times d$. It is straightforward to store the learned metric efficiently by keeping a low-rank approximation of it.



(a) *protein* (b) *isolet* (c) *imagenet5* (d) *cats5*

(e) *caltech30* (f) *tdt30* (g) *20news* (h) *rcv30*

Figure 4.3: The comparison of different stochastic algorithms for classification

4.3 Conclusions

In this chapter, we propose a dual random projection method to learn the distance metric for large-scale high-dimensional datasets. The main idea is to solve the dual problem in

[1]Many studies (e.g., [114, 116]) have shown that metric learning do not yield better classification accuracy than the standard classification algorithms (e.g., SVM) given a sufficiently large number of training data.

the subspace spanned by random projection, and then recover the distance metric in the original space using the estimated dual variables. We develop the theoretical guarantee that with a high probability, the proposed method can accurately recover the optimal solution with small error when the data matrix is of low rank, and the optimal dual variables even when the data matrix cannot be well approximated by a low rank matrix. Our empirical study confirms both the effectiveness and efficiency of the proposed algorithm for DML by comparing it to the state-of-the-art algorithms for DML. In the future, we may further improve the efficiency of our method by exploiting the scenario when optimal distance metric can be well approximated by a low rank matrix.

Chapter 5

Fine-Grained Visual Categorization via Multi-stage Metric Learning

In this chapter[1], we will address the challenge from the large number of constraints ($\mathcal{O}(n^3)$) and propose a new DML framework for FGVC by combining the approach in the last chapter. FGVC aims to distinguish objects in subordinate classes. For instance, dog images are classified into different breeds of dogs, such as "Chihuahua", "Pug", "Samoyed" and so on [70, 88]. As a result, FGVC classification has to handle the co-occurrence of two somewhat contradictory requirements: 1) it needs to distinguish many similar classes (e.g., the dog breeds that only have subtle difference), and 2) it needs to deal with large intra-class variance (e.g., caused by different poses, examples, etc.). Fig. 5.1 demonstrates an example and such co-occurring requirements make FVGC very challenging.

The popular pipeline for FVGC consists of two steps, feature extraction step and classification step. The feature extraction step, which sometimes combines with segmentation [2, 21, 88], part localization [8, 118] or both [20], is to extract image level representations, and popular choices are LLC features [2], Fisher vectors [44] and so on. A recent development is to use convolutional neural network (CNN) [72] trained on large-scale image classification dataset (e.g., ImageNet [92]) and then use the trained model to extract features [34]. The so-

[1]This chapter is adapted from the published paper: Q. Qian, R. Jin, S. Zhu and Y. Lin. Fine-Grained Visual Categorization via Multi-stage Metric Learning. In: Proceedings of the 28th IEEE Conference on Computer Vision and Pattern Recognition (CVPR'15), Boston, MA, 2015.

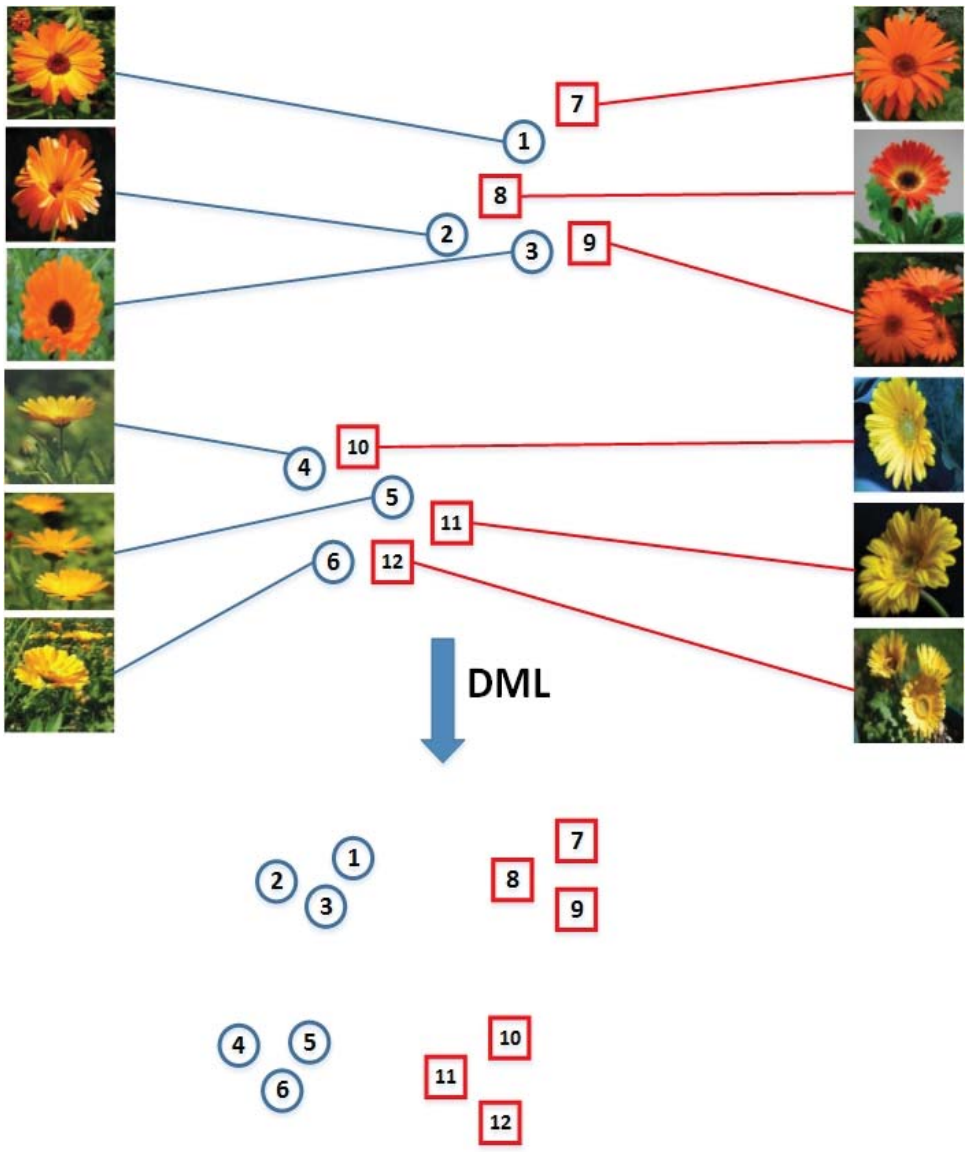


Figure 5.1: Illustration of how DML learns the embedding that pulls together the data points from the same class and pushes apart the data points from different classes. Blue points are from the class "English marigold" while red ones are "Barberton daisy". An important note here is that our DML does not require to collapse data points from each class and this allows the flexibility to model intra-class variance. A big challenge now is how to deal with high-dimension feature representation which is typical for image-level visual features. To this end, we propose a multi-stage scheme for metric learning.

called deep learning features have demonstrated the state-of-the-art performance on FGVC datasets [34]. Note that training CNN directly on FGVC datasets is difficult because the existing FGVC benchmarks are often too small [34] (only several tens of thousands of training

images or less). In this chapter, we simply take the state-of-the-art deep learning features without any other operators (e.g., segmentation) but focus on studying better classification approach to better address the aforementioned two co-occurring requirements in FGVC.

For the classification step, many existing FGVC methods simply learn a single classifier for each fine-grained class according to the one-vs-all strategy [2, 8, 21, 118]. Apparently, this strategy does not scale well to the number of fine-grained classes while the number of subordinate classes in FGVC could be very large (e.g., 200 classes in *birds11* dataset). Additionally, such one-vs-all scheme is only to address the first issue in the two issues, namely, it makes efforts to separate different classes without modeling intra-class variance. In contrast, this chapter proposes a DML approach, aiming to explicitly handle the two co-occurring requirements with a *single* metric. Fig. 5.1 illustrates how DML works. The key idea of DML is in two folds: 1) it learns a distance metric that pulls neighboring data points of the same class close to each other and data points from different classes far apart, and 2) it has the flexibility to define neighboring size and require only a portion of data points from the same class to be close to each other. Consequently, the flexibility of neighboring size acts as an effective way to model large intra-class variance. With a learned metric, a k-nearest neighbor classifier will be applied to find the class assignment for a test image.

There are three challenges in learning a metric directly from the original high dimensional space:

- *Large number of constraints*: A large number of training constraints are usually required to avoid the overfitting of high dimensional DML. The total number of triplet constraints could be up to $\mathcal{O}(n^3)$ where n is the number of examples.
- *Computational challenge*: DML has to learn a matrix of size $d \times d$, where d is the dimensionality of data and $d = 134,016$ in our study. The $\mathcal{O}(d^2)$ number of variables

leads to two computational challenges in finding the optimal metric. First, it results in a slower convergence rate in solving the related optimization problem [89]. Second, to ensure the learned metric to be positive semi-definitive (PSD), most DML algorithms require, at *every* iteration of optimization, projecting the intermediate solution onto a PSD cone, an expensive operation with complexity of $\mathcal{O}(d^3)$ (at least $\mathcal{O}(d^2)$).

- *Storage limitation*: It can be expensive to simply save $\mathcal{O}(d^2)$ number of variables in memory. For instance, in our study, it would take more than 130 GB to store the completed metric in memory, which adds more complexity to the already difficult optimization problem.

In this chapter, we propose a multi-stage metric learning framework for high dimensional DML to explicitly address these challenges. First, to deal with a large number of constraints used by high dimensional DML, we divide the original optimization problem into multiple stages. At each stage, only a small subset of constraints that are difficult to be classified by the currently learned metric will be adaptively sampled and used to improve the learned metric. By setting the regularizer appropriately, we can prove that the final solution is optimized over all appeared constraints. Second, to handle the computational challenge in each subproblem, we extend the theory of *dual random projection* [120], which was originally developed for linear classification problems, to metric learning. The proposed method, on one hand, enjoys the efficiency of random projection, and on the other hand learns a distance metric of size $d \times d$. This is in contrast to most dimensionality reduction methods that learn a metric in a *reduced* space. Finally, to handle the storage problem, we propose to maintain a low rank copy of the learned metric by a randomized algorithm for low rank matrix approximation. It not only accelerates the whole learning process but also regularizes the learned metric to avoid overfitting. Extensive comparisons on benchmark FGVC datasets

verify the effectiveness and efficiency of the proposed method.

The rest of the chapter is organized as follows: Section 5.1 describes the details of the proposed method. Section 5.2 shows the results of the empirical study, and Section 5.3 concludes this work.

5.1 Multi-stage Metric Learning

The proposed DML algorithm focuses on triplet constraints so as to pull the small portion of nearest examples from the same class together [114]. Let $\mathcal{D} = \{(\mathbf{x}_i, y_i), i = 1, \ldots, n\}$ be a collection of n training images, where $\mathbf{x}_i \in \mathbb{R}^d$ and y_i is the class assignment of $\mathbf{x}_i$. Given a distance metric M, the distance between two data points $\mathbf{x}_i$ and $\mathbf{x}_j$ is measured by

$$\text{dist}_M(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^\top M(\mathbf{x}_i - \mathbf{x}_j)$$

Let $\{\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t\}(t = 1, \ldots, N)$ be a set of N triplet constraints derived from the training examples in $\mathcal{D}$. Since in each constraint $(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t)$, $\mathbf{x}_i^t$ and $\mathbf{x}_j^t$ are assumed to share the same class that is different from that of $\mathbf{x}_k^t$, we expect $d_M(\mathbf{x}_i^t, \mathbf{x}_j^t) < d_M(\mathbf{x}_i^t, \mathbf{x}_k^t)$. As a result, the optimal distance metric M is learned by solving the following optimization problem

$$\min_{M \in S_d, M \succeq 0} \frac{\lambda}{2}\|M\|_F^2 + \sum_{t=1}^{N} \ell(d_M(\mathbf{x}_i^t, \mathbf{x}_k^t) - d_M(\mathbf{x}_i^t, \mathbf{x}_j^t)) \tag{5.1}$$

where S_d includes all $d \times d$ real symmetric matrices and $\ell(\cdot)$ is a loss function that penalizes the objective function when $d_M(\mathbf{x}_i^t, \mathbf{x}_k^t)$ is not significantly larger than $d_M(\mathbf{x}_i^t, \mathbf{x}_j^t)$. In this study, we choose the smoothed hinge loss [97] that appears to be more effective for

optimization than the hinge loss while keeping the benefit of large margin

$$\ell(x) \;=\; \begin{cases} 0 & : \quad x > 1 \\ 1 - x - \gamma/2 & : \quad x < 1 - \gamma \\ \frac{1}{2\gamma}(1-x)^2 & : \quad o.w. \end{cases}$$

One main computational challenge of DML comes from the PSD constraint $M \succeq 0$ in (5.1). We address this challenge by following the one-projection paradigm [24] that first learns a metric M without the PSD constraint and then projects M to the PSD cone at the very end of the learning process. Hence, in this study, we will focus on the following optimization problem for FGVC

$$\min_{M \in S_d} \frac{\lambda}{2}\|M\|_F^2 + \sum_{t=1}^{N} \ell(\langle A_t, M \rangle) \tag{5.2}$$

where $A_t = (\mathbf{x}_i^t - \mathbf{x}_k^t)(\mathbf{x}_i^t - \mathbf{x}_k^t)^\top - (\mathbf{x}_i^t - \mathbf{x}_j^t)(\mathbf{x}_i^t - \mathbf{x}_j^t)^\top$ is introduced as a matrix representation for each triplet constraint, and $\langle \cdot, \cdot \rangle$ represents the dot product between two matrices.

We will discuss the strategies to address the three challenges of high dimensional DML, and summarize the framework of high dimensional DML for FGVC at the end of this section.

5.1.1 Constraints Challenge: Multi-stage Division

In order to reliably determine the distance metric in a high dimensional space, a large number of training examples are needed to avoid the overfitting problem. Since the number of triplet constraints can be $\mathcal{O}(n^3)$, the number of summation terms in (5.2) can be extremely large, making it difficult to effectively solve the optimization problem in (5.2). Although active learning may help reduce the number of constraints [114], the number of active constraints

can still be very large since many images in FGVC from different categories are visually similar, leading to many mistakes. To address this challenge, we divide the learning process into multiple stages. At s-th stage, let M_{s-1} be the distance metric learned from the last stage. We sample a subset of active triplet constraints that are difficult to be classified by M_{s-1} (i.e., incur large hinge loss). Given M_{s-1} and the sampled triplet constraints $\mathcal{N}_s$, we update the distance metric by solving the following optimization problem

$$\min_{M_s \in S_d} \frac{\lambda}{2}\|M_s - M_{s-1}\|_F^2 + \sum_{t \in \mathcal{N}_s} \ell(\langle A_t, M_s \rangle) \tag{5.3}$$

Although only a small size of constraints is used to improve the metric at each stage, we have

Theorem 5. *The metric learned by solving the problem (5.3) is equivalent to optimize the following objective function*

$$\min_{M \in S_d} \frac{\lambda}{2}\|M\|_F^2 + \sum_{k=1}^{s} \sum_{t \in \mathcal{N}_k} \ell(\langle A_t, M \rangle)$$

Proof. Consider the objective function for the first s stages

$$\min_{M \in S_d} \underbrace{\frac{\lambda}{2}\|M\|_F^2 + \sum_{k=1}^{s-1} \sum_{t \in \mathcal{N}_k} \ell(\langle A_t, M \rangle)}_{:=\mathcal{L}_{s-1}(M)} + \sum_{t \in \mathcal{N}_s} \ell(\langle A_t, M \rangle) \tag{5.4}$$

It is obvious that $\mathcal{L}_{s-1}$ is strongly convex, so we have (Chapter 9, [12])

$$\begin{aligned}\mathcal{L}_{s-1}(M) = &\mathcal{L}_{s-1}(M_{s-1}) + \langle \nabla \mathcal{L}_{s-1}(M_{s-1}), M - M_{s-1} \rangle \\ &+ \frac{1}{2}\langle (M - M_{s-1}) \nabla^2 \mathcal{L}_{s-1}(M'), M - M_{s-1} \rangle\end{aligned}$$

for some M' between M and M_{s-1}.

Since M_{s-1}, the solution obtained from the first $s-1$ stages, approximately optimizes $\mathcal{L}_{s-1}(M)$ and $\mathcal{L}_{s-1}$ is λ-strongly convex, then

$$\mathcal{L}_{s-1}(M) \approx \mathcal{L}_{s-1}(M_{s-1}) + \frac{\lambda}{2}\|M - M_{s-1}\|_F^2 \tag{5.5}$$

We finish the proof by replacing $\mathcal{L}_{s-1}(M)$ in (5.4) with the approximation in (5.5). □

Remark This theorem demonstrates that the metric learned from the last stage is optimized over constrains from all stages. Therefore, the original problem could be divided into several subproblems and each only has an affordable number of active constraints. Fig. 5.2 summaries the framework of multi-stage learning procedure.

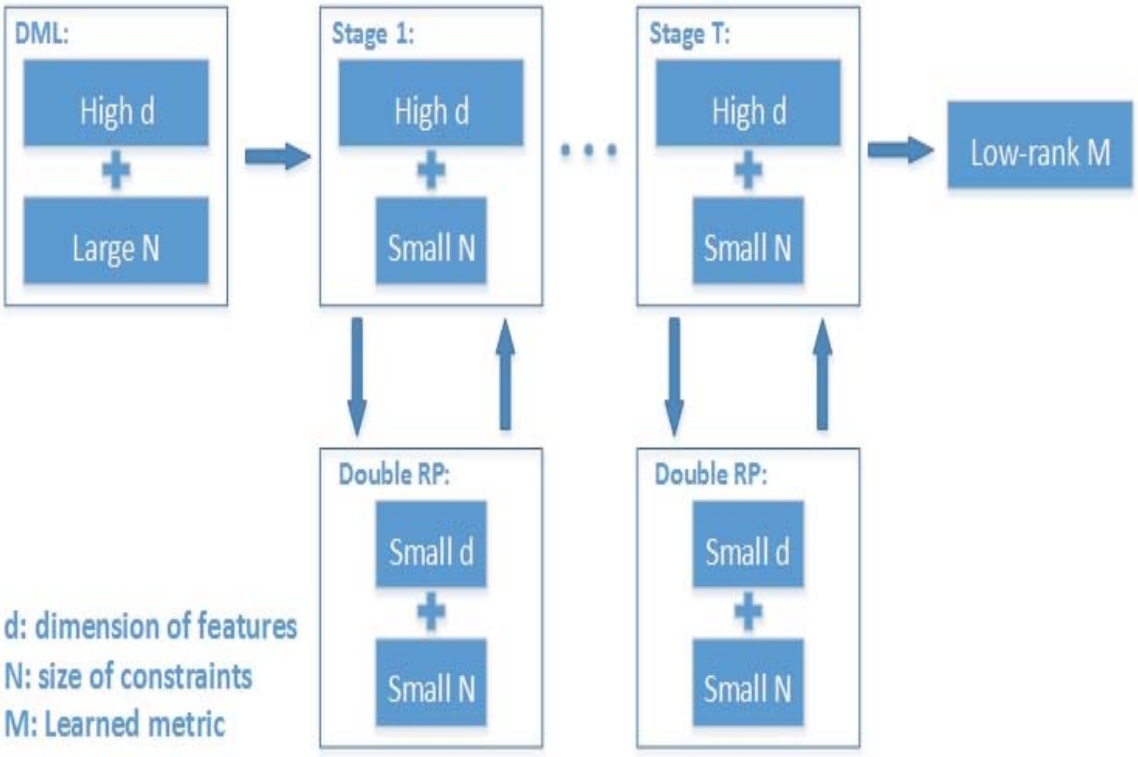


Figure 5.2: The framework of the proposed method.

5.1.2 Computational Challenge: Dual Random Projection

Now we will try to solve the high dimensional subproblem by dual random projection technique which is similar to that in the last chapter. For simplifying the analysis, we will investigate the subproblem at the first stage and following stages could be analyzed in the same way. By introducing the convex conjugate ℓ_* for ℓ in (5.3), the dual problem of DML is

$$\max_{\boldsymbol{\alpha}\in\mathbb{R}^{|\mathcal{N}_1|}} -\sum_{t=1}^{|\mathcal{N}_1|}\ell_*(\alpha_t) - \frac{1}{2\lambda}\boldsymbol{\alpha}^\top G\boldsymbol{\alpha} \tag{5.6}$$

where α_t is the dual variable for A_t and G is a matrix defined as $G_{a,b} = \langle A_a, A_b\rangle$. $M_1 = -\frac{1}{\lambda}\sum_{t=1}^{|\mathcal{N}_1|}\alpha_t A_t$ by setting the gradient with respect to M_1 to zeros. Let $R_1, R_2 \in \mathbb{R}^{d\times m}$ be two Gaussian random matrices, where m is the number of random projections ($m \ll d$) and $R_1^{i,j}, R_2^{i,j} \sim \mathcal{N}(0, 1/m)$. For each triplet constraint, we project its representation A_t into low dimensional space using the random matrices, i.e. $\widehat{A}_t = R_1^\top A_t R_2$. By using double random projections, which is different from the single random projection in [120], we have

Lemma 1. *$\forall A_a, A_b$, the double random projections preserve the pairwise similarity between them:* $\mathrm{E}[\langle\widehat{A}_a, \widehat{A}_b\rangle] = \langle A_a, A_b\rangle$

The proof is straightforward. According to the lemma, the dual variables in (5.6) could be estimated in the low dimension space as

$$\max_{\hat{\alpha}\in\mathbb{R}^{|\mathcal{N}_1|}} -\sum_{t=1}^{|\mathcal{N}_1|}\ell_*(\hat{\alpha}_t) - \frac{1}{2\lambda}\hat{\alpha}^\top \widehat{G}\hat{\alpha} \tag{5.7}$$

where $\widehat{G}(a,b) = \langle\widehat{A}_a, \widehat{A}_b\rangle$. Then, by the definition of convex conjugate and the smoothness of the loss function, which is different from the analysis as in Chapter 4, each dual variable

$\hat{\alpha}_t$ in (5.7) could further be estimated by $\ell'(\langle \widehat{A}_t, \widehat{M}_1 \rangle)$, where $\widehat{M}_1 \in \mathbb{R}^{m \times m}$ is the metric learned in the reduced space. Generally, $\widehat{M}_s$ is learned by solving the following optimization problem

$$\min_{\widehat{M}_s \in S_m} \frac{\lambda}{2} \|\widehat{M}_s - \widehat{M}_{s-1}\|_F^2 + \sum_{t=1}^{|\mathcal{N}_s|} \ell(\langle \widehat{A}_t, \widehat{M}_s \rangle) \tag{5.8}$$

Since the size of $\widehat{M}_s \in \mathbb{R}^{m \times m}$ is significantly smaller than that of M_s, (5.8) can be solved much more efficiently than (5.3). In our implementation, a simple stochastic gradient descent is developed to efficiently solve the optimization problem in (5.8). Given $\widehat{M}_1$, the final distance metric $M_1 \in \mathbb{R}^{d \times d}$ in the original space is estimated as

$$M_1 \;=\; -\frac{1}{\lambda} \sum_{t=1}^{|\mathcal{N}_1|} \hat{\alpha}_t A_t \tag{5.9}$$

5.1.3 Storage Challenge: Low Rank Approximation

Although (5.9) allows us to recover the distance metric M in original d dimensional space from the dual variables $\{\alpha_t\}_{t=1}^{|\mathcal{N}|}$, it is expensive, if not impossible, to save M in the memory since d is very large in FGVC [2]. To address this challenge, instead of saving M, we propose to save the low rank approximation of M. More specifically, let $\sigma_1, \ldots, \sigma_r$ be the first $r \ll d$ eigenvalues of M, and let $\mathbf{u}_1, \ldots, \mathbf{u}_r$ be the corresponding eigenvectors. We approximate M by a low rank matrix $M' = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{u}_i^\top = LL^\top$. Different from existing DML methods that directly optimize L [113], we obtain M first and then decompose it to avoid suboptimal soulition. Unlike M that requires $\mathcal{O}(d^2)$ storage space, it only takes $\mathcal{O}(rd)$ space to save M' and r could be arbitrary value. In addition, the low rank metric also accelerates the sampling step by reducing the cost of computing distance from $\mathcal{O}(d)$ to $\mathcal{O}(r)$. Low rank is

also a popular regularizer to avoid overfitting when learning high dimensional metric [78]. However, the key issue is how to efficiently compute the eigenvectors and eigenvalues of M at *each* stage. This is particularly challenging in our case as M in (5.9) even can not be computed explicitly due to its large size.

To address this problem, first we investigate the structure of the recovering step for s-th stage as in (5.9)

$$\begin{aligned} M_s &= M_{s-1} - \frac{1}{\lambda}\sum_{t=1}^{|\mathcal{N}_s|}\alpha_t^s A_t^s \\ &= M_{s-2} - \frac{1}{\lambda}(\sum_{t=1}^{|\mathcal{N}_s|}\alpha_t^s A_t^s + \sum_{t=1}^{|\mathcal{N}_{s-1}|}\alpha_t^{s-1} A_t^{s-1}) \\ &= -\frac{1}{\lambda}\sum_{k=1}^{s}\sum_{t=1}^{|\mathcal{N}_k|}\alpha_t^k A_t^k \end{aligned}$$

Therefore, we could express the summation as matrix multiplication. In particular, for each triplet $(\mathbf{x}_i^t, \mathbf{x}_j^t, \mathbf{x}_k^t)$, we denote its dual variable by $\alpha = \ell'(\langle \widehat{A}, \widehat{M} \rangle)$ and set the corresponding entries in a sparse matrix C as

$$\begin{aligned} C(i,j) = \frac{\alpha}{\lambda},\ C(j,i) = \frac{\alpha}{\lambda},\ C(j,j) = -\frac{\alpha}{\lambda} \\ C(i,k) = -\frac{\alpha}{\lambda},\ C(k,i) = -\frac{\alpha}{\lambda},\ C(k,k) = \frac{\alpha}{\lambda} \end{aligned} \tag{5.10}$$

It is easy to verify that M can be written as

$$M = XCX^\top \tag{5.11}$$

Second, we exploit the randomized theory [52] to efficiently compute the eigen-decomposition

of M. More specifically, let $R \in \mathbb{R}^{d\times q}$ ($q \ll d$) be an Gaussian random matrix. According to [52], with an overwhelming probability, most of the top r eigenvectors of M lie in the subspace spanned by the column vectors in MR provided $q \geq r + k$, where k is a constant independent from d. The limitation of the method is that it requires the appearance of the matrix M for computing MR while keeping the whole matrix is unaffordable here. Fortunately, by replacing M with $XCX^\top$ according to (5.11), we can approximate the top eigenvectors of M by those of $XCX^\top R$ that is of size $d \times q$ and can be computed efficiently since C is a sparse matrix. The overall computational cost of the proposed algorithm for low rank approximation is only $\mathcal{O}(qnd)$, which is linear in d. Note that the sparse matrix C is cumulated over all stages.

Alg. 5 summarizes the key steps of the proposed approach for low rank approximation, where qr and eig stand for QR and eigen decomposition of a matrix. Note that the distributed computing is particularly effective for the realization of the algorithm because the matrix multiplication $XCX^\top R$ can be accomplished in parallel, which is helpful when n is also large.

Algorithm 5 An Efficient Algorithm for Recovering M and Project It onto PSD Cone from $\widehat{M}$

Input: Dataset $X \in \mathbb{R}^{d\times n}$, $\widehat{M} \in \mathbb{R}^{m\times m}$, the number of random combinations q
Compute a Gaussian random matrix $R \in \mathbb{R}^{d\times q}$
Compute a sparse matrix C using (5.10)
$Y = R \times X^\top,\ Y = Y \times C,\ Y = Y \times X$
$[Q, R] = qr(Y)$
$B = Q^\top \times X^\top,\ B = B \times C,\ B = B \times X$
$[U, \Sigma] = eig(B)$
$U = Q * U$
return $L = [\sqrt{\sigma_1}\mathbf{u}_1, \cdots, \sqrt{\sigma_r}\mathbf{u}_r]$ and $M = LL^\top$, where $\mathbf{u}_i$ is the ith column of U and σ_i is the ith positive diagonal element of Σ

Alg. 6 shows the whole picture of the proposed method. The total cost of the framework

Algorithm 6 The **M**ulti-**s**tage **M**etric **L**earning Framework for High Dimensional DML (MsML)

Input: Dataset $X \in \mathbb{R}^{d \times n}$, the number of random projections m, the number of random combinations q, and the number of stages T
Compute two Gaussian random matrices $R_1, R_2 \in \mathbb{R}^{d \times m}$
Initialize $\widehat{M}_0 = \mathbf{0} \in \mathbb{R}^{m \times m}$ and $M_0 = \mathbf{0} \in \mathbb{R}^{d \times d}$
for $s = 1, \ldots, T$ **do**
    Sample one epoch ($\mathcal{O}(n)$) active triplet constraints using M_{s-1}
    Estimate $\widehat{M}_s$ by solving the optimization problem as in (5.8) using SGD
    Recover the distance metric M_s in the d dimensional space using Alg. 5
end for
return M_T

is only $\mathcal{O}(dn)$.

5.2 Experiments

DeCAF features [34] are extracted as the image representations in the experiments. Although it is from the activation of a deep convolutional network, which is trained on ImageNet [72], it outperforms conventional visual features on many general tasks [34]. We concatenate features from the last three full connected layers (i.e., DeCAF_{5+6+7}) and the dimension of resulting features is $51,456$.

We apply the proposed algorithm to learn a distance metric and use the learned metric together with a smoothed k-nearest neighbor classifier, a variant of k-NN, to predict the class assignments for test examples. Different from traditional k-NN, smoothed k-NN first obtains k reference centers for each class by clustering images in each class into k clusters. To predict the class assignment for a test image $\mathbf{x}$, it computes a distance between $\mathbf{x}$ and a

class C using the reference centers $C_1, \ldots, C_k$ as

$$dis(\mathbf{x}, C) = -\frac{1}{\beta} \log \left(\sum_{j=1}^{k} \exp \left(-\beta |\mathbf{x} - C_j|^2 \right) \right), \tag{5.12}$$

and assigns $\mathbf{x}$ to the class with the shortest distance. The distance function given in (5.12) actually computes the soft min among the distance between $\mathbf{x}$ and C_j, and we use hard min $\min_{1 \leq j \leq k} |\mathbf{x} - C_j|^2$ when $\beta = 0$. We find that smoothed k-NN is more efficient than the traditional k-NN, especially for large-scale learning problems. We refer to the classification approach based on the metric learned by the proposed algorithm and the smoothed k-NN as **MsML**, and the smoothed k-NN with Euclidean distance in the original space as **Euclid**. Although the size of covariance matrix is very large ($51,456 \times 51,456$), its rank is low due to the small number of training examples, and thus PCA can be computed explicitly. The state-of-the-art DML algorithm, i.e. **LMNN** [114] with PCA as preprocess, is also included in comparison. The one-vs-all strategy, based on the implementation of LIBLINEAR [37], is used as a baseline for FGVC, with the regularization parameter varied in the range $\{10^i\}(i = -2, \cdots, 3)$. We refer to it as **LSVM**. We also include the state-of-the-art results for FGVC in our evaluation. All the parameters used by MsML are set empirically, with the number of random projections $m = 100$ and the number of random combinations $q = 600$. PCA is applied for LMNN to reduce the dimensionality to m before DML is learned. LMNN is implemented by the code from the original authors and the recommended parameters are used [2]. To ensure that the baseline method fully exploits training data, we set the maximum number of iterations for LMNN as 10^4. These parameter values are used throughout all the experiments. All training/test splits are provided by datasets. **Mean**

[2] We did vary the parameter slightly from the recommended values and did not find any noticeable change in the classification accuracy.

accuracy, a standard evaluation metric for FGVC, is used to evaluate the classification performance. All experiments are run on a single machine with 16 2.10GHz cores and 96GB memory.

5.2.1 Oxford Cats&Dogs

Cats&dogs contains $7,349$ images from 37 cats and dogs species [88]. There are about 100 images per class for training and rest for testing. Table 5.1 summaries the results. First, we observe that MsML is more accurate than the baseline LSVM. This is not surprising because the distance metric is learned from the training examples of all class assignments. This is in contrast to the one-vs-all approach used in LSVM in which the classification function for a class C is learned only by the class assignments of C. Second, our method performs significantly better than the baseline DML method, indicating that the unsupervised dimension reduction method PCA may result in suboptimal solutions for DML. Fig. 5.3 compares the images that are most similar to the query images using the metric learned by the proposed algorithm (Column 8-10) to those based on the metric learned by LMNN (Column 5-7) and Euclid (Column 2-4). We observe that more similar images are found by the metric learned by MsML than by LMNN. For example, MsML is able to capture the difference between two cats species (longhair v.s. shorthair) while LMNN returns the very similar images with wrong label. More examples are given in Fig. 5.6. Third, MsML has overwhelming performance compared to all state-of-the-art FGVC approaches. Although the method [88] using ground truth head bounding box and segmentation achieves 59.21%, MsML is 20% better than it with only image information, which shows the advantage of the proposed method. Finally, it takes less than 0.2 second to extract DeCAF features per image based on a CPU implementation while a simple segmentation operator costs more than 2.5 seconds as reported in

the study [2], making the proposed method for FGVC practically more appealing.

Table 5.1: Comparison of mean accuracy(%) on *cats&dogs* dataset. "#" means that more information (e.g., ground true segmentation) is used by the method.

Methods	Mean Accuracy (%)
Image only [88]	39.64
Det+Seg [2]	54.30
Image+Head+Body# [88]	59.21
Euclid	72.60
LSVM	77.63
LMNN	76.24
MsML	80.45
MsML+	81.18

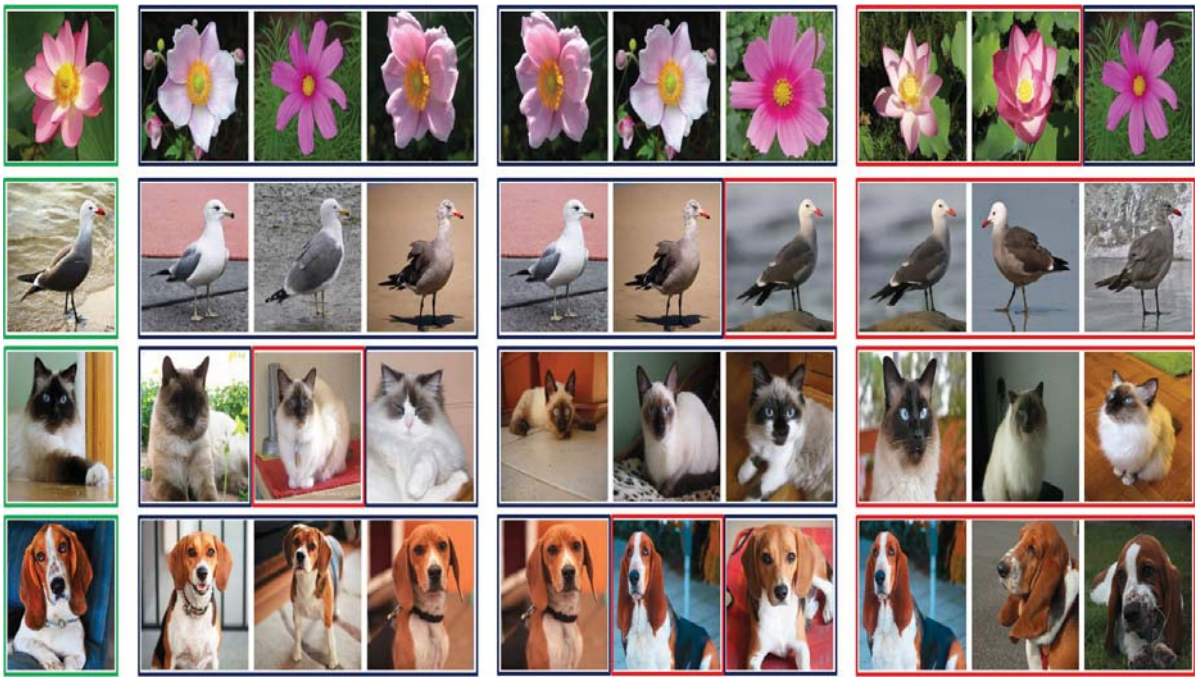


Figure 5.3: Examples of retrieved images. First column are query images highlighted by green bounding boxes. Columns 2-4 include the most similar images measured by Euclid. Columns 5-7 show those by the metric from LMNN. Columns 8-10 are from the metric of MsML. Images in columns 2-10 are highlighted by red bounding boxes when they share the same category as queries, and blue bounding boxes if they are not.

To evaluate the performance of MsML for extremely high dimensional features, we concatenate conventional features by using the pipeline for visual feature extraction that is outlined in [2]. Specifically, we extract HOG [30] features at 4 different scales and encode

them to $8K$ dimensional feature dictionary by the LLC method [113]. A max pooling strategy is then used to aggregate local features into a single vector representation. Finally, $82,560$ features are extracted from each image and the total dimension is up to $134,016$. MsML with the combined features is denoted as **MsML+** and we can observe that it further improves the performance by about 1% as in Table 5.1. Note that the time of extracting this high dimensional conventional features is only 0.5 second per image, which is still much cheaper than any segmentation or localization operator.

5.2.2 Oxford 102 Flowers

102flowers is the Oxford flowers dataset for flower species [86], which consists of 8189 images from 102 classes. Each class has 20 images for training and rest are for testing. Table 5.2 shows the results from different methods. We have the similar conclusion for the methods using the same DeCAF features. That is, MsML outperforms LSVM and LMNN significantly. Although LSVM already performs very well, MsML further improves the accuracy. Additionally, it is observed that the performance of state-of-the-art methods even with segmentation operators are much worse than that of MsML. Note that GT [86] uses hand annotated segmentations followed by multiple kernel SVM, while MsML outperforms it about 3% without any supervised information, which confirms the effectiveness of the proposed method.

Fig. 5.4 illustrates the changing trend of test mean accuracy as the number of stages increases. We observe that MsML converges very fast, which verifies that multi-stage division is essential in our proposed framework.

Table 5.2: Comparison of mean accuracy(%) on *102flowers* dataset. "#" means that more information (e.g., ground true segmentation) is used by the method.

Methods	Mean Accuracy (%)
Combined CoHoG [65]	74.80
Combined Features [85]	76.30
BiCoS-MT [19]	80.00
Det+Seg [2]	80.66
TriCoS [21]	85.20
GT# [86]	85.60
Euclid	76.21
LSVM	87.14
LMNN	81.93
MsML	88.39
MsML+	89.45

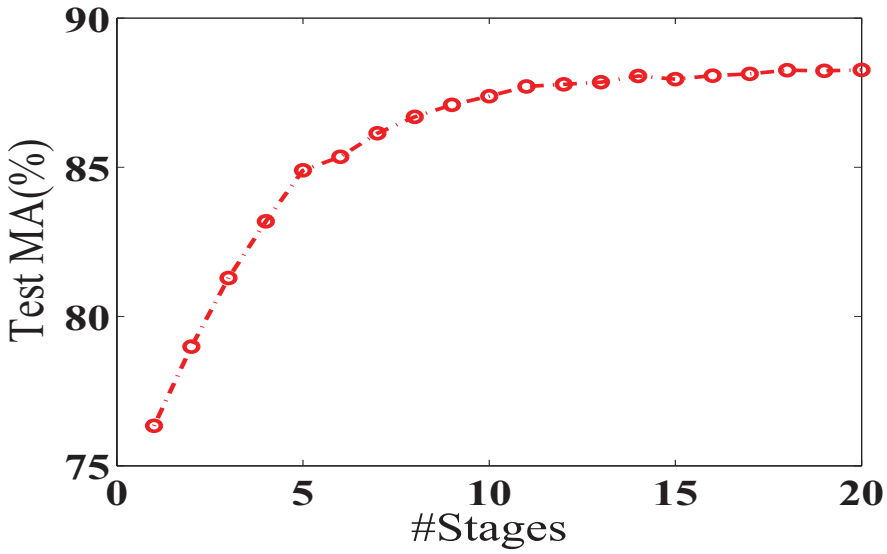


Figure 5.4: Convergence curve of the proposed method on *102flowers*.

5.2.3 Birds-2011

birds11 is the Caltech-USCD-200-2011 birds dataset for bird species [109]. There are 200 classes with total 11,788 images and each class has rough 30 images for training. We use the version with ground truth bounding box. Table 5.3 compares the proposed method to the state-of-the-art baselines. First, it is obvious that the performance of MsML is significantly

better than all baseline methods as the observation above. Second, although Symb [20] combines segmentation and localization, MsML outperforms it by 9% without any time consuming operator. Third, Symb* and Ali* mirror the training images to improve their performances, while MsML is even better without this trick. Finally, MsML outperforms the method combining DeCAF features and DPD models [121], which is due to the fact that most of research of FGVC ignore choosing the appropriate base classifier and simply adopt linear SVM with one-vs-all strategy. For comparison, we also report the result with mirroring training images and is denoted as **MsML+***. It provides another 1% improvement over MsML+ as shown in Table 5.3.

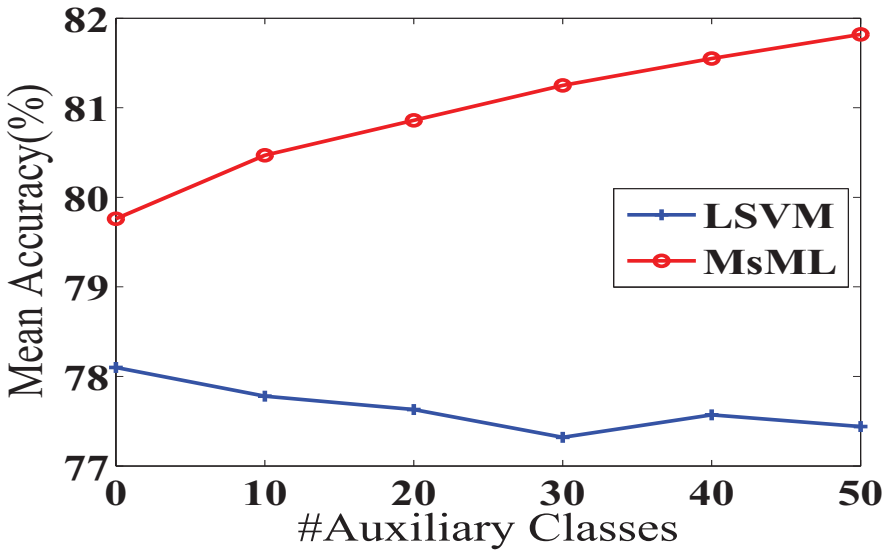


Figure 5.5: Comparison with different size of classes on *birds11*.

To illustrate the capacity of MsML in exploring the correlation among classes, which makes it more effective than a simple one-vs-all classifier for FGVC, we conduct one additional experiment. We randomly select 50 classes from *birds11* as the *target classes* and use the test images from the target classes for evaluation. When learning the metric, besides the training images from 50 target classes, we sample k classes from the 150 unselected ones

Table 5.3: Comparison of mean accuracy(%) on *birds11* dataset. "*" denotes the method that mirrors training images.

Methods	Mean Accuracy (%)
Symb [20]	56.60
POOF [9]	56.78
Symb* [20]	59.40
Ali* [44]	62.70
DeCAF+DPD [34]	64.96
Euclid	46.85
LSVM	61.44
LMNN	51.04
MsML	65.84
MsML+	66.61
MsML+*	67.86

as the *auxiliary* classes, and use training images from the auxiliary classes as additional training examples for distance metric learning. Fig. 5.5 compares the performance of LSVM and MsML with the increasing number of auxiliary classes. It is not surprising to observe that the performance of LSVM decreases a little since it is unable to explore the supervision information in the auxiliary classes to improve the classification accuracy of target classes and more auxiliary classes just intensify the class imbalance problem. In contrast, the performance of MsML improves significantly with increasing auxiliary classes, indicating that MsML is capable of effectively exploring the training data from the auxiliary classes and therefore is particulary suitable for FGVC.

5.2.4 Stanford Dogs

S-dogs is the Stanford dog species dataset [70]. It contains 120 classes and $20,580$ images, where 100 images from each class is used for training. Since it is the subset of ImageNet [92], where DeCAF model is trained from, we just report the result in Table 5.4 as reference.

Table 5.4: Comparison of mean accuracy(%) on *S-dogs* dataset. "*" denotes the method that mirrors training images.

Methods	Mean Accuracy (%)
SIFT [70]	22.00
Edge Templates [118]	38.00
Symb [20]	44.10
Symb* [20]	45.60
Ali* [44]	50.10
Euclid	59.22
LSVM	65.00
LMNN	62.17
MsML	69.07
MsML+	69.80
MsML+*	70.31

5.2.5 Comparison of Efficiency

In this section, we compare the training time of the proposed algorithm for high dimensional DML to that of LSVM and LMNN. MsML is implemented by Julia, which is a little slower than C[3], while LSVM uses the LIBLINEAR package, the state-of-the-art algorithm for solving linear SVM implemented mostly in C. The core part of LMNN is also implemented in C. The time for feature extraction is not included here because it is shared by all the methods in comparison. The running time for MsML includes all operational cost (i.e., the cost for triplet constraints sampling, computing random projection and low rank approximation).

Table 5.5: Comparison of Running time (seconds).

Methods	*cats&dogs*	*102flowers*	*birds11*	*S-dogs*
LSVM	196.2	309.8	1,417.0	1,724.8
LMNN	832.58	702.7	1,178.2	1,643.6
MsML	164.91	174.4	413.1	686.3
MsML+	337.20	383.7	791.3	1,229.7

Table 5.5 summarizes the results of the comparison. First, it takes MsML about 1/3 of the time to complete the computation than LMNN. This is because MsML employs a

[3]Detailed comparison can be found in http://julialang.org

stochastic optimization method to find the optimal distance metric while LMNN is a batch learning method. Second, we observe that the proposed method is significantly more efficient than LSVM on most of datasets. The high computational cost of LSVM mostly comes from two aspects. First, LSVM has to train one classification model for each class, and becomes significantly slower when the number of classes is large. Second, the fact that images from different classes are visually similar makes it computationally difficult to find the optimal linear classifier that can separate images of one class from images from the other classes. In contrast, the training time of MsML is independently from the number of classes, making it more appropriate for FGVC. Finally, the running time of MsML+ with $134,016$ features only doubles that of MsML, which verifies that the proposed method is linear in dimensionality ($\mathcal{O}(d)$).

5.3 Conclusions

In this chapter, we develop a multi-stage metric learning framework of high dimensional FGVC problem, which addresses the challenges arising from conventional DML. More specifically, it divides the original problems into multiple stages to handle the challenge arising from too many triplet constraints, extends the theory of dual random projection to address the computational challenge for high dimensional data, and a randomized algorithm for low rank matrix approximation for the storage challenge. The empirical study shows that the proposed method with general purpose features yields performance that is significantly better than the state-of-the-art approaches for FGVC. In the future, we could try to combine the proposed DML algorithm with segmentation and localization to further improve the performance of FGVC.

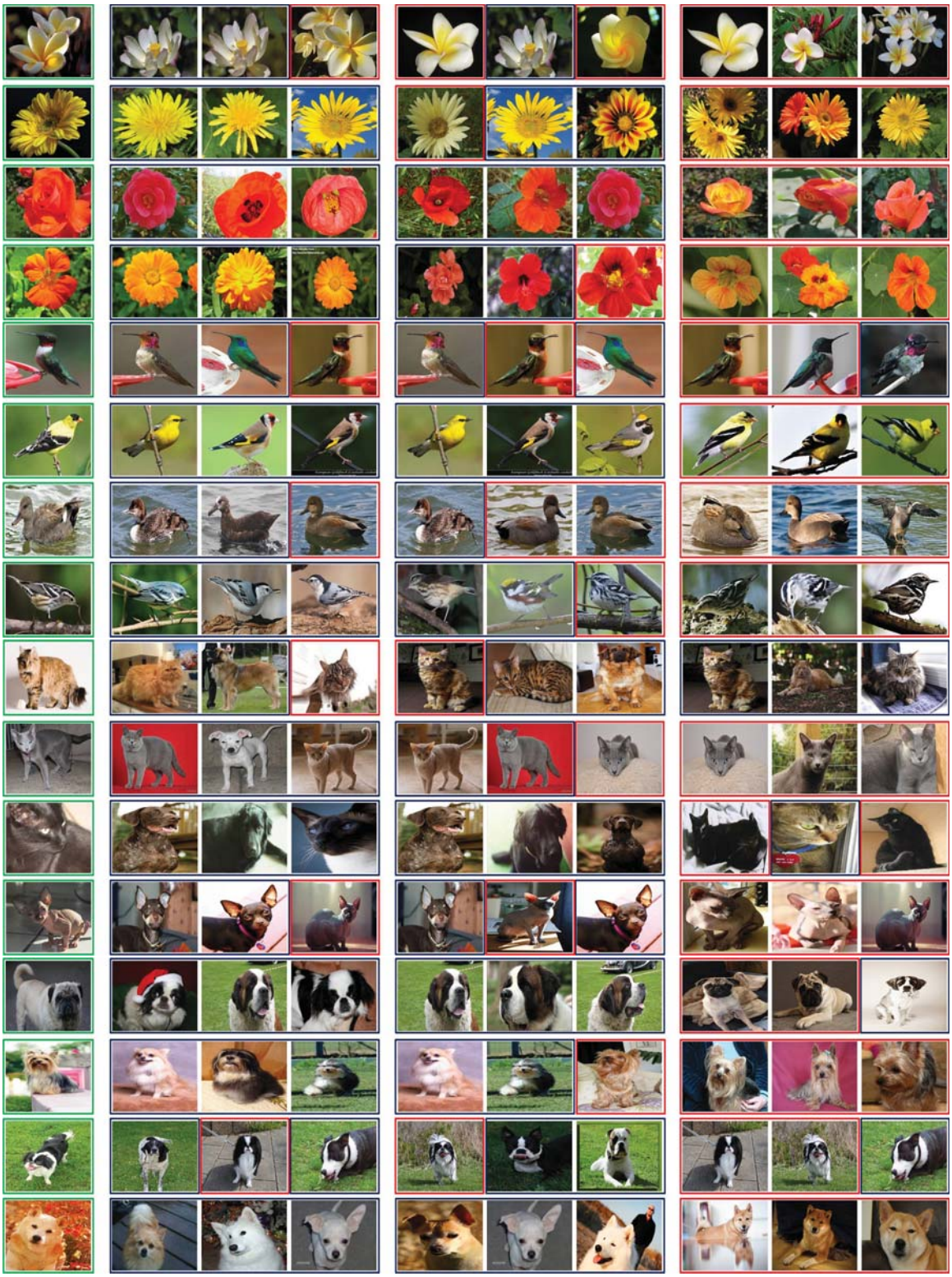


Figure 5.6: Examples of retrieved images. First column are query images highlighted by green bounding boxes. Columns 2-4 include the most similar images measured by Euclid. Columns 5-7 show those by the metric from LMNN. Columns 8-10 are from the metric of MsML. Images in columns 2-10 are highlighted by red bounding boxes when they share the same category as queries, and blue bounding boxes if they are not.

Chapter 6

Feature Selection for Face Recognition: A Distance Metric Learning Approach

In this chapter, we will study feature selection problem in face recognition. Face recognition involves identifying if two face images belong to the same person (i.e., face verification) or which person the face image belongs to (i.e., face classification). It is an important application of computer vision and has been studied extensively in the past decades [26, 100, 3, 62, 25]. Many face recognition methods require estimating the similarity between images appropriately using the technique of distance metric learning [50, 60, 61].

One challenge in applying DML to estimate the similarity of face images is the high dimensionality of face representations used to capture the subtle differences between the faces [25, 3]. The high dimensionality comes from the fact that **over-completed** descriptors are sampled for face images and the total number of features can be up to $1,000,000$ [61]. Most of existing DML methods project the high dimensional face features to a low dimensional space with unsupervised dimension reduction techniques (e.g., PCA) and then learn a metric for the low dimensional space [60, 50]. The resulting solution can be suboptimal since important information of face images is lost after dimension reduction.

By further investigating the problem, it is found that although the dimensionality of feature vectors for face images is very high, the number of features used by each descriptor is significantly small and is only 45 in our study. It is thus efficient to learn a metric for the group of features from each descriptor. On the other hand, since the descriptors are over-completed, a subset of them may be sufficient to verify face images. Fig. 6.1 illustrates the feature selection (i.e., descriptor selection) for face verification. Each descriptor is corresponding to a small patch on the face, and a subset of them (e.g., eyes, nose, etc.) can capture most of differences.

Figure 6.1: Illustration of feature selection for face verification. Although descriptors are over-completed, a subset of descriptors (e.g., eyes, nose, etc) can capture most of differences.

In this chapter, we propose a DML method that can select a subset of descriptors and learn a distance metrics for each selected descriptor simultaneously. Unlike the previous methods that learn a single metric from the original feature space, we learn a metric for each feature group that is corresponding to the selected descriptor at every iteration and the ensemble of metrics is used for the verification. The method avoids dealing with high dimensional features directly while keeping useful information from all descriptors.

The main contributions of this work are summarized as follows.

- We propose a DML method to handle the over-completed descriptors in face verification. At each iteration, a linear optimization problem is solved to simultaneously select a descriptor and learn the corresponding metric. This problem has the closed-form solution, which making the computation efficient.

- We prove that the proposed method has $\mathcal{O}(1/T)$ convergence rate, where T is the number of iterations and equivalent to the number of selected descriptors in this work.
- To reduce the high computational cost of exhaustive search that selects one single descriptor at each iteration, we exploit mini-batch strategy and incremental sampling, respectively. It is shown theoretically and empirically that the accelerated methods are able to yield similar performance with significantly reduced computational costs. The hybrid method with these strategies together achieves the best efficiency in our empirical study.

The rest of this chapter is organized as follows: Section 6.1 describes the proposed approach. Section 6.2 shows the results of the empirical study and Section 6.3 concludes this work with future research directions.

6.1 DML for feature selection

Given a set of face images $X \in \mathbb{R}^{d \times n}$, DML aims to learn a metric that estimates pairwise distances as

$$\text{dist}_M(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^\top M (\mathbf{x}_i - \mathbf{x}_j)$$

In face verification, a good metric can separate the images from the same person and those from different persons as

$$\forall \mathbf{x}_i, \mathbf{x}_j : \ \text{dist}_M(\mathbf{x}_i, \mathbf{x}_j) \leq b - 1; \quad \forall \mathbf{x}_i, \mathbf{x}_k : \ \text{dist}_M(\mathbf{x}_i, \mathbf{x}_k) \geq b + 1$$

where $\mathbf{x}_i$ and $\mathbf{x}_j$ are from the same person while $\mathbf{x}_k$ is from a different person, and b is a pre-defined threshold.

The metric can be learned by solving the corresponding optimization problem, which is

$$\min_{M \in \mathcal{S}_+^d} \mathcal{L}(M) = \frac{1}{N} \sum_{i,j}^{N} \ell(Y_{ij}(\text{dist}_M(\mathbf{x}_i, \mathbf{x}_j) - b)) \tag{6.1}$$

where $\mathcal{S}_+^d$ is the set of positive semi-definite (PSD) matrices with size $d \times d$, and $\ell(\cdot)$ can be any smooth loss function and is the logistic loss in our work as

$$\ell(z) = \log(1 + \exp(z + 1))$$

$Y_{ij} \in \{-1, +1\}$ is the indicator where $Y_{ij} = 1$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ are from the same person and $Y_{ij} = -1$ otherwise.

It is hard to handle the problem in (6.1) directly by conventional DML methods due to the high dimensionality of the feature space. On the other hand, features are known to be redundant and the number of features from each descriptor is small, making it possible to efficiently learn a metric for every group of feature. . Therefore, we propose an iterative algorithm that is able to perform feature selection and metric learning simultaneously: it selects one informative descriptor at each iteration and learn a metric for the features of the selected descriptor.

At the t-th iteration, we select the descriptor as

$$\arg \min_{s=1,\cdots,S} \min_{M_t \in S_+^k, \|M_t\|_F \leq 1} \langle \vec{\ell'}(Z^{t-1}), \vec{z_s^t} \rangle$$

where S is the number of descriptors, k is the dimension of features from a single descriptor and we have $kS = d$. $\vec{\ell'}(Z_{t-1}) = [\ell'(Z_{i,j}^{t-1})]$, $\vec{z_s^t} = [Y_{i,j}(\text{dist}_{M_t}(\mathbf{x}_i^s, \mathbf{x}_j^s) - b)]$ and $\mathbf{x}^s$ is the s-th

feature group of the example. It is a linear optimization problem and is equivalent to

$$\arg \min_{s=1,\cdots,S} \min_{M_t \in S_+^k, \|M_t\|_F \leq 1} \langle \sum_{i,j} Y_{i,j} \ell'^{t-1}_{i,j} A^s_{i,j}, M_t \rangle \tag{6.2}$$

where $A^s_{i,j} = (\mathbf{x}^s_i - \mathbf{x}^s_j)(\mathbf{x}^s_i - \mathbf{x}^s_j)^\top$. The intuition of Eqn. 6.2 is to learn the optimal metric for each feature group and then greedily select the feature group whose distance is the closest to the negative direction of the gradient with the learned metric.

The metric is learned from the subproblem

$$\min_{M_t \in S_+^k, \|M\|_F \leq 1} \langle \sum_{i,j} Y_{i,j} \ell'^{t-1}_{i,j} A^s_{i,j}, M_t \rangle$$

According to the K.K.T. condition [12], the problem has the closed-form solution

$$M_t^* = \Pi_{PSD, \|M_t\|_F = 1}(-\sum_{i,j} Y_{i,j} \ell'^{t-1}_{i,j} A^s_{i,j}) \tag{6.3}$$

where $\Pi_{PSD, \|M\|_F = 1}(M)$ is to project the metric into the PSD cone with the unit Frobenius norm. By taking Eqn. 6.3 back to Eqn. 6.2, the selection criterion becomes

$$\arg \min_{s=1,\cdots,S} \text{Score}_s = -\|\Pi_{PSD}(-\sum_{i,j} Y_{i,j} \ell'^{t-1}_{i,j} A^s_{i,j})\|_F \tag{6.4}$$

After selecting the descriptor with the minimal score, the distance is updated as

$$Z^t_{i,j} = (1 - \alpha_t) Z^{t-1}_{i,j} + \alpha_t z^t_{i,j}$$

The stepsize α_t can be optimized by solving the problem

$$\min_{\alpha_t:0<\alpha_t<1} \frac{1}{N} \sum_{i,j} \ell((1-\alpha_t)Z_{i,j}^{t-1} + \alpha_t z_{i,j}^t)$$

with Newton's method or use the pre-defined size as

$$\alpha_t = \frac{2}{t+1}$$

Alg. 7 summarizes the key steps of the proposed method.

Algorithm 7 Greedy Coordinate Descent Metric Learning for Face Verification (Greco)

1: **Input:** Dataset $X \in \mathbb{R}^{d \times n}$, # Iterations T
2: **for** $t = 1, \cdots, T$ **do**
3: Select a single descriptor according to Eqn. 6.4
4: Compute the corresponding metric as in Eqn. 6.3
5: Update the distance using the selected feature group and the learned metric as $Z_{i,j}^t = (1-\alpha_t)Z_{i,j}^{t-1} + \alpha_t z_{i,j}^t$
6: **end for**
7: **return** Z^T

The greedy coordinate descent method in Alg. 7 is similar as Frank-Wolfe method [41, 43] or conditional gradient descent method [75], but it learns the ensemble of metrics rather than a single model as in previous work. We have the following theorem for the guarantee of the performance.

Theorem 6. *Let Z^T be the solution of Alg. 7 after T iterations and Z^* be the optimal solution. Assume that the loss function $\ell(\cdot)$ is convex and β-smooth and $\forall i, j: \ \|z^i - z^j\|^2 \leq D$. By setting $\alpha_t = \frac{2}{t+1}$, we have*

$$\mathcal{L}(Z^t) - \mathcal{L}(Z^*) \leq \frac{2\beta D}{T+1}$$

The proof is similar to that of Frank-Wolfe method.

Proof.

$$\begin{aligned}
&\mathcal{L}(Z^t) - \mathcal{L}(Z^*) = \mathcal{L}((1-\alpha_t)Z^{t-1} + \alpha_t z^t) - \ell(Z^*) \\
&\leq \mathcal{L}(Z^{t-1}) - \mathcal{L}(Z^*) + \alpha_t \langle \nabla \mathcal{L}(Z^{t-1}), z^t - Z^{t-1} \rangle + \frac{\alpha_t^2 \beta}{2} \|Z^{t-1} - z^t\|^2 \qquad (6.5) \\
&\leq \mathcal{L}(Z^{t-1}) - \mathcal{L}(Z^*) + \alpha_t \langle \nabla \mathcal{L}(Z^{t-1}), Z^* - Z^{t-1} \rangle + \frac{\alpha_t^2 \beta}{2} D \qquad (6.6) \\
&\leq (1-\alpha_t)(\mathcal{L}(Z^{t-1}) - \ell(Z^*)) + \frac{\alpha_t^2 \beta}{2} D \qquad (6.7)
\end{aligned}$$

Eqn. 6.5 is obtained because that the loss function is β-smooth. Eqn. 6.6 is from the fact that z^t is the optimal solution of Problem 6.2 and Z^* is in the convex hull as z^t. Eqn. 6.7 is due to that $\mathcal{L}$ is convex.

By setting $\alpha_t = \frac{2}{t+1}$, we prove the theorem by induction. We hold the assumption that

$$\ell(Z^T) - \ell(Z^*) \leq \frac{2\beta D}{T+1}$$

When $T = 1$, $\alpha_1 = 1$ and it is obvious that

$$\ell(Z^1) - \ell(Z^*) \leq \frac{\beta D}{2} \leq \beta D$$

For $T = t$, we have

$$\begin{aligned}
\ell(Z^t) - \ell(Z^*) &\leq (1 - \frac{1}{t+1})\frac{2\beta D}{t} - \frac{1}{t+1}\frac{2\beta D}{t} + \frac{2\beta D}{(t+1)^2} \\
&\leq \frac{2\beta D}{t+1}
\end{aligned}$$

□

6.1.1 Adaptive Mini-batch Strategy

Although the proposed method can handle the high dimensional challenge in face verification, it has to exhaustively search through all descriptors to select only one descriptor at each iteration, which is computationally inefficient when the number of descriptors is very large. We propose two variants of DML based feature selection methods to improve the efficiency.

First, we investigate the mini-batch strategy, which select a mini-batch descriptors rather than a single descriptor at each iteration. After computing the score of each descriptor as in Eqn. 6.4, top m descriptors will be selected. Then, a larger metric (i.e., size of $mk \times mk$) will be learned from all features of selected descriptors as

$$M_t^* = \Pi_{PSD, \|M_t\|_F = 1/m}(-\sum_{i,j} Y_{i,j} \ell'^{t-1}_{i,j} A_{i,j}^{s_1, \cdots, s_d})$$

Note that the norm of M_t is shrank by $1/m$ to make sure the norm of z^t is still bounded over multiple feature groups. It is because

$$\|A_{i,j}^{s_1, \cdots, s_m}\|_F \leq m(\max_{i,j,s} \|A_{i,j}^s\|_F)$$

This shrinkage ratio also can be estimated empirically as $\frac{\max \|A_{i,j}^{s_1, \cdots, s_m}\|_F}{\max \|A_{i,j}^s\|_F}$ for a tighter approximation.

Given the normalized score of mini-batch descriptors, the updating is adaptively by comparing to the score of the optimal single descriptor. If the former score is better than the later one, the distance will be updated by the mini-batch feature groups. Otherwise, the

single optimal descriptor will be used. Alg. 8 shows the method with mini-batch strategy.

Algorithm 8 Greedy Coordinate Descent Metric Learning with Adaptive Mini-batch (Greco-mini)

Input: Dataset $X \in \mathbb{R}^{d \times n}$, # Iterations T, mini-batch size m
for $t = 1, \cdots, T$ **do**
  Select top d descriptors according to Eqn. 6.4
  Compute a single metric from the selected feature groups
  if $\langle \nabla \mathcal{L}(Z^{t-1}), z_{M_{mini-batch}} \rangle \geq \langle \nabla \mathcal{L}(Z^{t-1}), z_{M_{single}} \rangle$ **then**
    $M_t = M_{mini-batch}$
  else
    $M_t = M_{single}$
  end if
  Update the distance using the selected feature group and the learned metric as $Z^t_{i,j} = (1 - \alpha_t) Z^{t-1}_{i,j} + \alpha_t z^t_{i,j}$
end for
return Z^T

We can prove that the mini-batch strategy will not affect the convergence rate if its score is better than the single one, which is stated in the following theorem.

Theorem 7. *Let Z^T be the solution of Alg. 8 after T iterations and Z^* be the optimal solution. Assume that the loss function $\ell(\cdot)$ is convex and β-smooth and $\forall i, j : \|z^i - z^j\|^2 \leq D$. By setting $\alpha_t = \frac{2}{t+1}$ and assuming that the score of mini-batch descriptors is η better than the score of the optimal single descriptor, we have*

$$\mathcal{L}(Z^t) - \mathcal{L}(Z^*) \leq \frac{2\beta D - 2\eta}{T + 1}$$

The proof is trivial and we list it for completeness.

Proof. We consider the problem when the solution of each iteration is slightly better than the single optimal solution with a factor η where $\eta > 0$ and $\langle \nabla \ell(Z^{t-1}), \tilde{z}^t \rangle \leq \langle \nabla \ell(Z^{t-1}), z^t \rangle - \eta$.

With the similar process as above, we have

$$\begin{aligned}
&\mathcal{L}(Z^t) - \mathcal{L}(Z^*) = \mathcal{L}((1-\alpha_t)Z^{t-1} + \alpha_t \tilde{z}^t) - \ell(Z^*) \\
&\leq \mathcal{L}(Z^{t-1}) - \mathcal{L}(Z^*) + \alpha_t \langle \nabla \mathcal{L}(Z^{t-1}), z^t - Z^{t-1} \rangle - \alpha_t \eta + \frac{\alpha_t^2 \beta}{2} \|Z^{t-1} - \tilde{z}^t\|^2 \\
&\leq (1-\alpha_t)(\mathcal{L}(Z^{t-1}) - \mathcal{L}(Z^*)) - \alpha_t \eta + \frac{\alpha_t^2 \beta}{2} D
\end{aligned}$$

We prove the result by induction. We set $\alpha_t = 2/(t+1)$ as before and the assumption is

$$\ell(Z^T) - \ell(Z^*) \leq \frac{2\beta D - 2\eta}{T+1}$$

When $T = 1$, it is

$$\ell(Z^1) - \ell(Z^*) \leq \frac{\beta D}{2} - \eta \leq \beta D - \eta$$

When $T = t$, we have

$$\begin{aligned}
\ell(Z^t) - \ell(Z^*) &\leq \frac{2\beta D}{t+1} - (1 - \frac{2}{t+1})\frac{2\eta}{t} - \frac{2\eta}{(t+1)^2} \\
&\leq \frac{2\beta D - 2\eta}{t+1}
\end{aligned}$$

□

Remark η is the trade-off between the performance and the number of selected descriptors. Although the number of iterations for the method with mini-batch strategy is less than that of the original method to achieve the same performance, the number of descriptors can be larger than that of original one.

However, if the solution of mini-batch is good enough as

$$\eta \geq \frac{\beta D}{2}(1 - \frac{T+1}{Tm+1})$$

where m is the mini-batch size, we have

$$\ell(Z^T) - \ell(Z^*) \leq \frac{2\beta D}{Tm+1}$$

which means that even with the same number of descriptors, the method with mini-batch performs the same as the original method and the running time is shrank by the factor m in this ideal case.

6.1.2 Incremental Sampling Strategy

In this section, we exploit the sampling strategy to reduce the number of candidate descriptors at each iteration. Different from uniform sampling, a subset with size $\lfloor 1 - \frac{\gamma_t}{t+1} \rfloor S$ will be sampled at the t-th iteration, where $0 < \gamma_t < t+1$ and $\max_t \gamma_t = c$. The size of candidate subset is incremental with the increasing number of iterations, which means the selected descriptor is closer and closer to the optimal one and this step is critical for convergence. Alg. 9 summarizes the proposed method.

We can show that the method with incremental sampling has the similar convergence rate as the original one, which is summarized in the following theorem.

Theorem 8. *Let Z^T be the solution of Alg. 9 after T iterations and Z^* be the optimal solution. Assume that the loss function $\ell(\cdot)$ is convex and β-smooth and $\forall i, j: \ \|z^i - z^j\|^2 \leq$*

Algorithm 9 Greedy Coordinate Descent Metric Learning with Incremental Sampling (Greco-isamp)

1: **Input:** Dataset $X \in \mathbb{R}^{d\times n}$, # Iterations T, sampling ratio γ_t
2: **for** $t = 1, \cdots, T$ **do**
3: Randomly sample a subset of descriptors with size of $\lfloor 1 - \frac{\gamma_t}{t+1} \rfloor S$
4: Select a single descriptor in the subset according to Eqn. 6.4 and compute the corresponding metric
5: Update the distance using the select feature group and the learned metric as $Z^t_{i,j} = (1-\alpha_t)Z^{t-1}_{i,j} + \alpha_t z^t_{i,j}$
6: **end for**
7: **return** Z^T

D. By setting $\alpha_t = \frac{2}{t+1}$ *and* $0 < \gamma_t < \min\{c, t+1\}$*,we have*

$$E[\mathcal{L}(Z^t)] - \mathcal{L}(Z^*) \leq \frac{2\beta D + 4c}{T+1}$$

Proof. We consider the general case that the solution at the t-th iteration is not optimal but close to the optimal one with a factor η where $0 < \eta_t < 1$ and $\langle \nabla\mathcal{L}(Z^{t-1}), \tilde{z}^t\rangle \leq \eta_t \langle \nabla\mathcal{L}(Z^{t-1}), z^t\rangle$ since the optimal score is negative. With the similar strategy as in previous proof, we have

$$\begin{aligned}
&\mathcal{L}(Z^t) - \mathcal{L}(Z^*) = \mathcal{L}((1-\alpha_t)Z^{t-1} + \alpha_t \tilde{z}_t) - \mathcal{L}(Z^*) \\
&\leq \mathcal{L}(Z^{t-1}) - \mathcal{L}(Z^*) + \alpha_t \langle \nabla\mathcal{L}(Z^{t-1}), \eta_t z^t - Z^{t-1}\rangle + \frac{\alpha_t^2\beta}{2}\|Z^{t-1} - \tilde{z}^t\|^2 \\
&\leq \mathcal{L}(Z^{t-1}) - \mathcal{L}(Z^*) + \alpha_t\eta \langle \nabla\mathcal{L}(Z^{t-1}), Z^* - Z^{t-1}\rangle \\
&-\alpha_t(1-\eta_t)\langle \nabla\mathcal{L}(Z^{t-1}), Z^{t-1}\rangle + \frac{\alpha_t^2\beta}{2}D \\
&\leq (1-\alpha_t\eta)(\mathcal{L}(Z^{t-1}) - \mathcal{L}(Z^*)) + \alpha_t(1-\eta_t)(\mathcal{L}(0) - \mathcal{L}(Z^{t-1})) + \frac{\alpha_t^2\beta}{2}D \qquad (6.8) \\
&= (1-\alpha_t)(\mathcal{L}(Z^{t-1}) - \mathcal{L}(Z^*)) + \alpha_t(1-\eta_t)(\mathcal{L}(0) - \mathcal{L}(Z^*)) + \frac{\alpha_t^2\beta}{2}D \\
&\leq (1-\alpha_t)(\mathcal{L}(Z^{t-1}) - \mathcal{L}(Z^*)) + 2\alpha_t(1-\eta_t) + \frac{\alpha_t^2\beta}{2}D \qquad (6.9)
\end{aligned}$$

Eqn. 6.8 is from the fact that $\mathcal{L}$ is convex and Eqn. 6.9 is because

$$\mathcal{L}(0) - \mathcal{L}(Z^*) = \log(1+e) - \ell(Z^*) < 2$$

If we set $\eta_t \geq 1 - \gamma_t/(t+1)$ where $0 < \gamma_t < \min\{c, t+1\}$, Eqn. 6.9 becomes

$$\ell(Z^t) - \ell(Z^*) \leq (1-\alpha_t)(\ell(Z^{t-1}) - \ell(Z^*)) + 2\frac{\alpha_t\gamma_t}{t+1} + \frac{\alpha_t^2\beta}{2}D$$

We prove the result by induction. We set $\alpha_t = 2/(t+1)$ and take the assumption

$$\ell(Z^T) - \ell(Z^*) \leq \frac{2\beta D + 4c}{T+1}$$

When $T = 1$, it is

$$\ell(Z^1) - \ell(Z^*) \leq \gamma_1 + \frac{\beta D}{2} \leq \beta D + 2c$$

When $T = t$, we have

$$\begin{aligned} \ell(Z^t) - \ell(Z^*) &\leq \frac{2\beta D}{t+1} + (1 - \frac{2}{t+1})\frac{4c}{t} + \frac{4\gamma_t}{(t+1)^2} \\ &\leq \frac{2\beta D + 4c}{t+1} \end{aligned}$$

We finish the proof by showing that $\eta_t \geq 1 - \gamma_t/(t+1)$ in Alg. 9, which is

$$\begin{aligned} E[\langle \nabla\mathcal{L}(Z^{t-1}), \tilde{z}^t\rangle] &\leq \Pr[z_t \in \mathcal{S}_{sub}]\langle \nabla\mathcal{L}(Z^{t-1}), z^t\rangle \\ &\leq (1 - \frac{\gamma_t}{t+1})\langle \nabla\mathcal{L}(Z^{t-1}), z^t\rangle \end{aligned}$$

□

The proposed adaptive mini-batch strategy and incremental sampling can be combined to further improve the efficiency and the hybrid algorithm is summarized in Alg. 10.

Algorithm 10 Greedy Coordinate Descent Metric Learning with Hybrid strategies (Greco-hybrid)

Input: Dataset $X \in \mathbb{R}^{d \times n}$, # Iterations T, mini-batch size m, sampling ratio γ_t
for $t = 1, \cdots, T$ **do**
  Randomly sample a subset of descriptors with size of $\lfloor 1 - \frac{\gamma_t}{t+1} \rfloor S$
  Select top d descriptors according to Eqn. 6.4 within the selected subset
  Compute a single metric from the selected feature groups
  if $\langle \nabla \mathcal{L}(Z^{t-1}), z_{M_{mini-batch}} \rangle \geq \langle \nabla \mathcal{L}(Z^{t-1}), z_{M_{single}} \rangle$ **then**
    $M_t = M_{mini-batch}$
  else
    $M_t = M_{single}$
  end if
  Select a single descriptor in the subset according to Eqn. 6.4 and compute the corresponding metric
  Update the distance using the select feature group and the learned metric as $Z^t_{i,j} = (1 - \alpha_t) Z^{t-1}_{i,j} + \alpha_t z^t_{i,j}$
end for
return Z^T

6.2 Experiments

We collect a face image dataset with $4,000$ training examples and $2,040$ test images. There are 400 people in training set and each of them has 10 images. Test set consists of 60 different people with 34 images for each. We use the covariance matrix descriptors [105] to extract features and $7,671$ descriptors are sampled for face images. Each descriptor has 45 features and the final dimensionality of representations is up to $345,195$.

Five variants of proposed methods are compared through the experiments:

- **Greco-random**: randomly select a descriptor at each iteration and learn the optimal metric

- **Greco**: the method that selects the feature group and learns the metric simultaneously as in Alg. 7
- **Greco-mini**: the proposed method with adaptive mini-batch strategy as in Alg. 8
- **Greco-isamp**: the proposed method with incremental sampling strategy as in Alg. 9
- **Greco-hybrid**: the proposed hybrid method as in Alg. 10

The number of iterations is set to be $T = 100$. The size of mini-batch is set as $m = 3$. We set the sampling ratio of Greco-isamp as 10% in the first 50 iterations and $1 - 45/(t + 1)$ in the rest ones. All experiments are repeated three times and the average results are reported.

6.2.1 Experiment I: Face Verification

For face verification, we randomly sample $4,000$ positive pairs and negative pairs respectively to compute the score of each descriptor at each iteration. $40,800$ randomly sampled pairs from test set, where half of them are positive, are used as test pairs. We also randomly sample $80,000$ pairs from training set ,where half of them are positive, to estimate the training error. Besides the method proposed in this work, we include ITML [32] the state-of-the-art DML, in the comparison. Since it can not handle the high dimensional data directly, we reduce the dimension to 500 by PCA before applying the method. The number of iterations for ITML is set to be $1,000,000$ to make sure the approach has explored the data sufficiently. Note that Greco samples $8,000$ pairs at each iteration, the total number of pairs used in our method is only $800,000$, which is less than that of ITML. The threshold is set empirically as $b = 6$.

Figs. 6.2-6.3 compare the performance of different methods on face verification problem. First, it is obvious that all proposed methods perform significantly better than ITML. This

is because the using of PCA in ITML results in a significant loss of information. Second, the performance of Greco-random is worse than the proposed methods, which verifies the effectiveness of the proposed greedy coordinate descent method. Third, the training error of Greco-mini decreased more rapidly than Greco, indicating that the incorporation of mini-batch method is effective, as illustrated in Theorem 7. Finally, Greco-isamp and Greco-hybrid have the similar performance as Greco and Greco-mini, respectively, confirming that the incremental sampling will not sacrifice the performance too much.

Table 6.1 compares the running time of proposed methods. Compared with Greco, Greco-mini saves more than 40% running time while Greco-isamp only uses 28% of that. The hybrid method is the most efficient one and takes 11% running time of Greco. Besides running time, we find that Greco-mini samples 126 descriptors and Greco-hybrid samples 119 descriptors to achieve the same performance as the Greco with 100 descriptors. We thus conclude that Greco-mini and Greco-hybrid trade the number of descriptors for efficiency at limited cost.

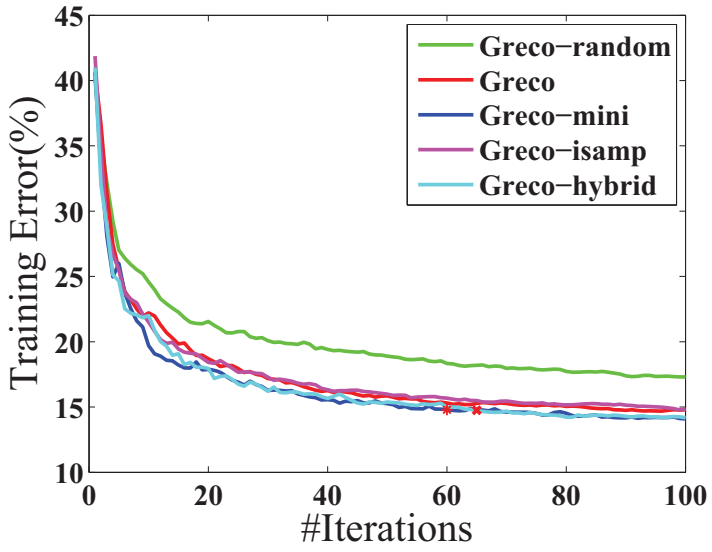


Figure 6.2: Comparison of training error on face verification. Red star denotes the position that Greco-mini achieves the same performance as Greco with 100 iterations. Red cross denotes the position that Greco-hybrid achieves the same performance as Greco with 100 iterations.

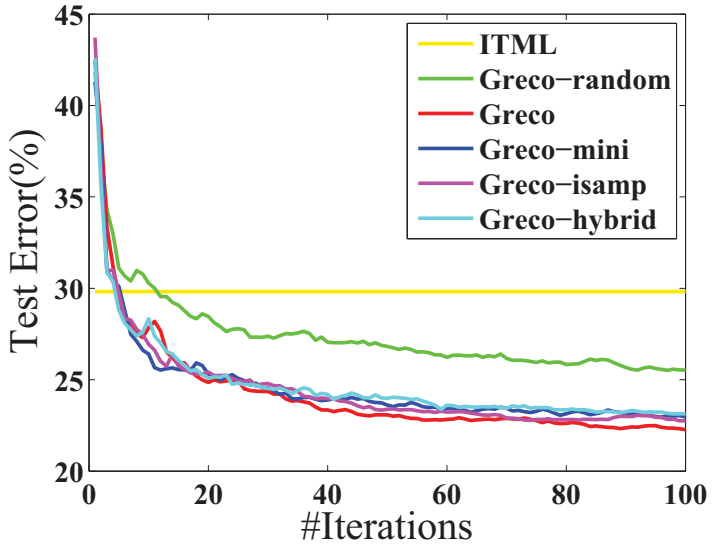


Figure 6.3: Comparison of test error on face verification.

Table 6.1: Comparison of running time (seconds). The reported running time of Greco-mini and Greco-hybrid is that they achieve the same training performance as Greco with 100 iterations.

Face verification			
Greco	Greco-mini	Greco-isamp	Greco-hybrid
3,716.3	2,084.4	1,044.5	412.8
Face classification			
Greco	Greco-mini	Greco-isamp	Greco-hybrid
3,643.8	2,252.8	1,012.5	356.7

6.2.2 Experiment II: Face Classification

In this section, we conduct the experiment for face classification task. The only difference from the face verification is the loss function. Given the triplet constraint $\{\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k\}$ where $\mathbf{x}_i$ and $\mathbf{x}_j$ are from the same person and $\mathbf{x}_k$ is different, we have the learning problem as

$$\min_{M \in \mathcal{S}_+^d} \mathcal{L}(M) = \frac{1}{N} \sum_{i,j,k}^{N} \ell(\mathrm{dist}_M(\mathbf{x}_i, \mathbf{x}_j) - \mathrm{dist}_M(\mathbf{x}_i, \mathbf{x}_k))$$

which can be solved by the proposed methods with appropriate modifications. We include LMNN [114] with 500 PCA features as the baseline DML method in comparison and the

results of leave-one-out 3-NN are summarized in Figs. 6.4-6.5.

We share the same observation as the last experiment, i.e. the learned distance metric with PCA features yields significantly worse performance than the proposed methods, and the proposed methods are significantly better than randomly picking a descriptor at each iteration. Greco-isamp has the similar performance as Greco and Greco-mini is better than Greco. Greco-hybrid has the comparable performance with Greco-mini but takes significantly less running time.

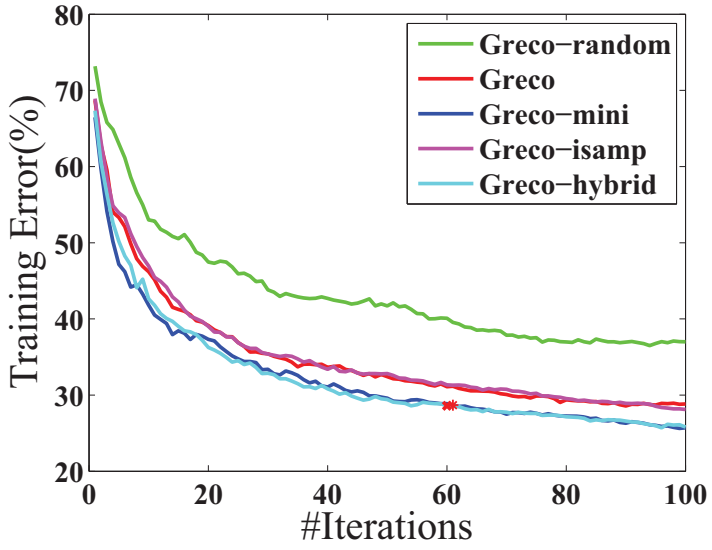


Figure 6.4: Comparison of training error on face classification. Red star denotes the position that Greco-mini achieves the same performance as Greco with 100 iterations. Red cross denotes the position that Greco-hybrid achieves the same performance as Greco with 100 iterations.

Besides comparing to general DML methods, we also conduct the comparison that includes the proposed hybrid method and the DML method specified for face recognition [61]. Since more constraints will approximate the expectation in Eqn. 6.1 better, we increase the number of sampled constraints at every iteration from one epoch (i.e., 4000) to five and ten epochs, which are denoted as Greco-hybrid-1, Greco-hybrid-5 and Greco-hybrid-10, respectively. Figs. 6.6-6.7 summarize the results of the proposed hybrid methods and the baseline

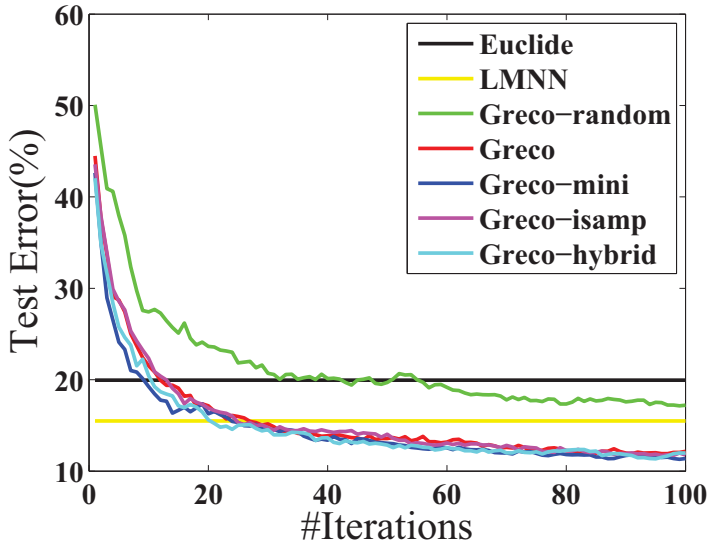


Figure 6.5: Comparison of test error on face classification.

method developed by NEC [61], which also selects features along with learning metrics. It is a commercial software and among the top performers for face recognition. With the sufficient number of constraints at each iteration, our hybrid methods perform even better than the well deployed software. The NEC's baseline method consumes 14, 799.1 seconds on the server with 32 core 2.10 GHz CPUs, while our hybrid methods cost only 532.4 and 583.7 seconds for Greco-hybrid-5 and Greco-hybird-10, respectively, to achieve the same performance by using the server with 24 core 2.30 GHz CPUs, which further verifies the effectiveness and efficiency of the proposed method.

6.3 Conclusion

In this chapter, we propose a DML method to select descriptors (i.e., feature groups) and learn corresponding metrics simultaneously for face recognition. Our method exploits the fact that the dimensionality of the features from a single descriptor is small and greedily selects one descriptor from all candidates at each iteration. To improve the computational

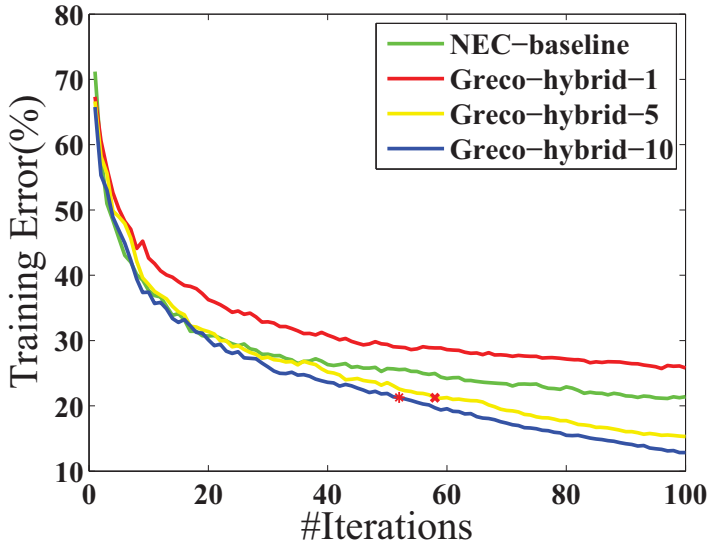


Figure 6.6: Comparison of training error on face classification with the baseline method developed by NEC. Red cross denotes the position that Greco-hybrid-5 achieves the same performance as NEC's baseline method with 100 iterations. Red star denotes the position that Greco-hybrid-10 achieves the same performance as NEC's baseline method with 100 iterations.

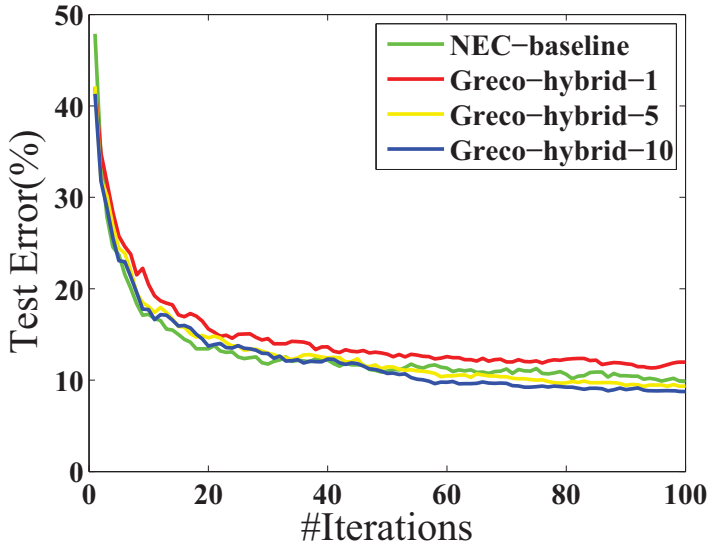


Figure 6.7: Comparison of test error on face classification with the baseline method developed by NEC.

efficiency, we propose two variants with mini-batch and incremental sampling strategies, respectively. Both the efficiency and effectiveness of the hybrid method with these two

strategies are also verified in the empirical study. In the future, we could try the method for the face images with different kinds of descriptors where the dimensionality of features from different descriptor can be different.

Chapter 7

Conclusions & Future Plan

In this dissertation, we study the fundamental challenges in applying DML to large-scale high dimensional data and evaluate the performance of the proposed methods with computer vision applications. We summarize them as follows.

- Expensive PSD projections ($\mathcal{O}(d^3)$): We alleviate the cost of PSD projections by reducing the number of updates in SGD. To limit its impact on the prediction performance, we exploit the mini-batch strategy and adaptive sampling method, respectively. We verify the effectiveness for both of them theoretically and empirically. Then, we propose two hybrid methods by combining these two strategies and further improve the efficiency while obtaining the similar performance as the standard SGD.

- Large number of variables ($\mathcal{O}(d^2)$): We develop the dual random projection technique for high dimensional DML. We first project the data from the original space to the low dimensional space by random projection. Then, the dual variables are estimated there and the final metric is reconstructed in the original space according to them. The proposed method is as efficient as random projection methods while it could recover the optimal metric theoretically. The empirical study confirms its effectiveness and efficiency.

- Huge number of triplet constraints ($\mathcal{O}(n^3)$): To handle huge number of constraints, we divide the original problem into multiple stages where each subproblem only keeps

a small subset of active constraints. After learning through all stages, the final metric is nearly optimal over all appeared constraints as shown in the theoretical analysis. Then, we evaluate the proposed multi-stage metric learning framework for the challenging FGVC task. It achieves the significantly better performance than state-of-the-art FGVC methods while it is even more efficient than the linear SVM.

- Feature selection: We propose a greedy method to select the feature group and learn the corresponding metric simultaneously at each iteration with guaranteed performance. Then, we investigate the adaptive mini-batch strategy and incremental sampling to alleviate the high computational cost of exhausted search that finds one feature group at each iteration. In addition, the hybrid approach is also studied in this dissertation that combines the strength of adaptive mini-batch and incremental sampling. The empirical study on face verification and classification confirms the effectiveness and efficiency of the proposed methods.

In the future, the work on distance metric learning inspired by this dissertation can be further continued in the following direction:

- **Improving Efficiency of Distance Metric Learning by Effectively Leveraging the Sparse Structure of Data**

  In this direction, we plan to generalize the feature selection work presented in Chapter 6 to a general sparse structure in the distance metric to be learned. We plan to explore the theory of compressive sensing, group lasso, and matrix completion for more effective DML for high dimensional large-scale data.

APPENDIX

Appendix

Proofs

Proof of Theorem 1

Proof. Using the standard analysis for online learning (Chapter12, [18]), we have

$$
\begin{aligned}
&\ell_t(M_t) - \ell_t(M_*) \leq \langle M_t - M_*, \nabla \ell_t(M_t) \rangle \\
&\leq \frac{\|M_t - M_*\|_F^2}{2\eta} - \frac{\|M_{t+1} - M_*\|_F^2}{2\eta} - \frac{\|M_t - M_{t+1}\|_F^2}{2\eta} + \langle M_t - M_{t+1}, \nabla \ell_t(M_t) \rangle \\
&\leq \frac{\|M_t - M_*\|_F^2}{2\eta} - \frac{\|M_{t+1} - M_*\|_F^2}{2\eta} - \frac{\|M_t - M_{t+1}\|_F^2}{2\eta} + \frac{1}{b} \sum_{s=1}^{b} \langle M_t - M_{t+1}, \nabla \ell_{t,s}(M_t) \rangle \\
&\leq \frac{\|M_t - M_*\|_F^2}{2\eta} - \frac{\|M_{t+1} - M_*\|_F^2}{2\eta} + \frac{\eta}{2b} \sum_{s=1}^{b} \|\nabla \ell_{t,s}(M_t)\|_F^2
\end{aligned}
$$

By taking the expectation with respect to the t-th mini-batch of triplet constraint, we have

$$
\mathcal{L}(M_t) - \mathcal{L}(M_*) \leq \frac{\|M_t - M_*\|_F^2}{2\eta} - \frac{\mathrm{E}_t[\|M_{t+1} - M_*\|_F^2]}{2\eta} + \frac{\eta}{2b} \sum_{s=1}^{b} \mathrm{E}_{t,s}[\|\nabla \ell_{t,s}(M_t)\|_F^2]
$$

By adding the inequalities of all iterations and taking expectation over the sequence of triplet constraints, we have

$$
\sum_{t=1}^{T} \mathrm{E}[\mathcal{L}(M_t)] - \mathcal{L}(M_*) \leq \frac{1}{2\eta} \|M_1 - M_*\|_F^2 + \frac{\eta}{2b} \underbrace{\sum_{t=1}^{T} \sum_{s=1}^{b} \mathrm{E}[\|\nabla \ell_{t,s}(M_t)\|_F^2]}_{:= C_T}
$$

According to Proposition 2, we have $\ell'(z)^2 \leq |\ell'(z)| \leq L\ell(z)$. Using $A = \max_{1\leq t\leq N} \|A_t\|_F$, we have

$$C_T = \sum_{t=1}^{T}\sum_{s=1}^{b} \mathrm{E}_{t,s}[\|\nabla \ell_{t,s}(M_t)\|_F^2] \leq LA^2 b \sum_{t=1}^{T} \mathrm{E}[\mathcal{L}(M_t)]$$

Using the result for C_T, we have

$$\left(1 - 3\eta LA^2\right) \mathcal{L}(\bar{M}) \leq \mathcal{L}(M_*) + \frac{R^2}{2\eta T}$$

We complete the proof by dividing both sides with $1 - 3\eta LA^2$ and replacing T with N/b. □

Proof of Theorem 2

Proof. To bound the number of updates, we have

$$\mathrm{E}\left[\sum_{t=1}^{N} Z_t\right] = \mathrm{E}\left[\sum_{t=1}^{N} |\ell_t'(M_t)|\right] \leq L\mathrm{E}\left[\sum_{t=1}^{N} \mathcal{L}(M_t)\right]$$

where the last step follows from $|\ell_t'(M)| \leq L\ell_t(M)$.

Using the standard analysis for online learning (Chapter 12, [18]), we have:

$$\begin{aligned}
& \ell(M_t) - \ell(M_*) \leq \langle \ell'(M_t) A_t, M_t - M_* \rangle \\
= \;& \tau_t Z_t \langle A_t, M_t - M_* \rangle + (\ell'(M_t) - \tau_t Z_t)\langle A_t, M_t - M_* \rangle \\
\leq \;& \frac{\|M_t - M_*\|_F^2 - \|M_{t+1} - M_*\|_F^2}{2\eta} + \frac{\eta A^2 Z_t}{2} + \tau_t(|\ell'(M_t)| - Z_t)\langle A_t, M_t - M_* \rangle
\end{aligned}$$

Taking the sum from $t = 1$ to N and expectation over both binary variables $\{Z_t\}_{t=1}^N$ and

the sequence of triplet constraints, we have:

$$\sum_{t=1}^{N} \mathrm{E}[\mathcal{L}(M_t)] - \mathcal{L}(M_*) \leq \frac{\|M_1 - M_*\|_F^2}{2\eta} + \frac{\eta A^2}{2} \mathrm{E}\left[\sum_{t=1}^{N} Z_t\right]$$

We complete the proof by reorganizing the above inequality. □

Proof of Theorem 3

Proof. First, we want to prove that $\widehat{G}$ is a good estimation for G. We rewrite $G_{a,b}$ by Kronecker product:

$$\begin{aligned}
G_{a,b} &= \langle A_a, A_b \rangle \\
&= \langle (\mathbf{x}_i^a - \mathbf{x}_k^a)(\mathbf{x}_i^a - \mathbf{x}_k^a)^\top - (\mathbf{x}_i^a - \mathbf{x}_j^a)(\mathbf{x}_i^a - \mathbf{x}_j^a)^\top, (\mathbf{x}_i^b - \mathbf{x}_k^b)(\mathbf{x}_i^b - \mathbf{x}_k^b)^\top - (\mathbf{x}_i^b - \mathbf{x}_j^b)(\mathbf{x}_i^b - \mathbf{x}_j^b)^\top \rangle \\
&= \langle (\mathbf{x}_i^a - \mathbf{x}_k^a) \otimes (\mathbf{x}_i^a - \mathbf{x}_k^a) - (\mathbf{x}_i^a - \mathbf{x}_j^a) \otimes (\mathbf{x}_i^a - \mathbf{x}_j^a), (\mathbf{x}_i^b - \mathbf{x}_k^b) \otimes (\mathbf{x}_i^b - \mathbf{x}_k^b) - (\mathbf{x}_i^b - \mathbf{x}_j^b) \otimes (\mathbf{x}_i^b - \mathbf{x}_j^b) \rangle \\
&= \langle \mathbf{z}_a, \mathbf{z}_b \rangle
\end{aligned}$$

where $\mathbf{z}_t = (\mathbf{x}_i^t - \mathbf{x}_k^t) \otimes (\mathbf{x}_i^t - \mathbf{x}_k^t) - (\mathbf{x}_i^t - \mathbf{x}_j^t) \otimes (\mathbf{x}_i^t - \mathbf{x}_j^t)$. Define $Z = [\mathbf{z}_1, \cdots, \mathbf{z}_N]$, we have $G = Z^\top Z$.

Under the low rank assumption that all training examples lie in the subspace of r-dimension, the dataset $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_n]$ can be decomposed as:

$$\mathbf{X} = U \Sigma V = \sum_{i=1}^{r} \lambda_i \mathbf{u}_i \mathbf{v}_i^\top$$

where λ_i is the i-th singular value of $\mathbf{X}$, and $\mathbf{u}_i$ and $\mathbf{v}_i$ are the corresponding left and right singular vectors of $\mathbf{X}$. Given the property of Kronecker product that $(A \otimes B)(C \otimes D) =$

$(AC) \otimes (BD)$, we have:

$$\begin{aligned}
\mathbf{z}_t &= (\mathbf{x}_i^t - \mathbf{x}_k^t) \otimes (\mathbf{x}_i^t - \mathbf{x}_k^t) - (\mathbf{x}_i^t - \mathbf{x}_j^t) \otimes (\mathbf{x}_i^t - \mathbf{x}_j^t) \\
&= [U(\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_k^t)] \otimes [U(\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_k^t)] - [U(\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_j^t)] \otimes [U(\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_j^t)] \\
&= (U \otimes U) \left[(\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_k^t) \otimes (\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_k^t) - (\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_j^t) \otimes (\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_j^t) \right]
\end{aligned}$$

where $\widetilde{\mathbf{x}}_i^t = U^\top \mathbf{x}_i^t$. Define $\widetilde{Z} = (\widetilde{\mathbf{z}}_1, \ldots, \widetilde{\mathbf{z}}_n)$, where $\widetilde{\mathbf{z}}_t = (\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_j^t) \otimes (\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_j^t) - (\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_k^t) \otimes (\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_k^t)$, we have:

$$G = \widetilde{Z}^\top (U^\top \otimes U^\top)(U \otimes U)\widetilde{Z} = \widetilde{Z}^\top (I_r \otimes I_r)\widetilde{Z} = \widetilde{Z}^\top \widetilde{Z}$$

where $I_r \otimes I_r$ equals to the identity operator of $r^2 \times r^2$.

With the random projection approximation, we have:

$$\begin{aligned}
\mathbf{z}_t &= (R^\top(\mathbf{x}_i^t - \mathbf{x}_k^t)) \otimes (R^\top(\mathbf{x}_i^t - \mathbf{x}_k^t)) - (R^\top(\mathbf{x}_i^t - \mathbf{x}_j^t)) \otimes (R^\top(\mathbf{x}_i^t - \mathbf{x}_j^t)) \\
&= [R^\top U(\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_k^t)] \otimes [R^\top U(\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_k^t)] - [R^\top U(\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_j^t)] \otimes [R^\top U(\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_j^t)] \\
&= (R^\top \otimes R^\top)(U \otimes U) \left[(\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_k^t) \otimes (\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_k^t) - (\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_j^t) \otimes (\widetilde{\mathbf{x}}_i^t - \widetilde{\mathbf{x}}_j^t) \right]
\end{aligned}$$

So,

$$\begin{aligned}
\widehat{G} &= \widetilde{Z}^\top (U^\top \otimes U^\top)(RR^\top \otimes RR^\top)(U \otimes U)\widetilde{Z} \\
&= \widetilde{Z}([U^\top RR^\top U] \otimes [U^\top RR^\top U])\widetilde{Z}
\end{aligned}$$

In order to bound the difference between G and $\widehat{G}$, we need the following corollary:

Corollary 9. *[120] Let $S \in \mathbb{R}^{r\times m}$ be a standard Gaussian random matrix. Then, for any $0 < \varepsilon \leq 1/2$, with a probability $1-\delta$, we have*

$$\left\| \frac{1}{m} SS^\top - I \right\|_2 \leq \varepsilon$$

provided

$$m \geq \frac{(r+1)\log(2r/\delta)}{c\varepsilon^2}$$

where constant c is at least $1/4$.

Define $\Delta = U^\top RR^\top U - I_r$. Using Corollary. 9, with a probability $1-\delta$, we have $\|\Delta\|_2 \leq \varepsilon$. Using the notation Δ, we have the following expression for $\widehat{G} - G$

$$\begin{aligned} \widehat{G} - G &= \widetilde{Z}^\top \left((I_r + \Delta) \otimes (I_r + \Delta) - I_r \otimes I_r \right) \widetilde{Z} \\ &= \widetilde{Z}^\top \left(\Delta \otimes I_r + I_r \otimes \Delta + \Delta \otimes \Delta \right) \widetilde{Z} = \widetilde{Z}^\top \Gamma \widetilde{Z} \end{aligned}$$

where $\Gamma = \Delta \otimes I_r + I_r \otimes \Delta + \Delta \otimes \Delta$. Using the fact that the eigenvalue values of $A \otimes B$ is given by $\lambda_i(A)\lambda_j(B)$, it is easy to verify that,

$$\|\Gamma\|_2 \leq \|\Delta\|_2 + \|\Delta\|_2 + \|\Delta\|_2\|\Delta\|_2$$

Using the fact that $\|\Delta\|_2 \leq \varepsilon$ and taking $\epsilon \leq 1/6$ which results in $c \geq 1/3$, with a probability $1-\delta$, we have

$$\|\Gamma\|_2 \leq 3\varepsilon \tag{.1}$$

Define $L(\boldsymbol{\alpha})$ and $\widehat{L}(\boldsymbol{\alpha})$ as

$$L(\boldsymbol{\alpha}) = -\sum_{i=1}^{n} \ell_*(\alpha_i) - \frac{1}{2\lambda N}\boldsymbol{\alpha}^\top G\boldsymbol{\alpha},$$
$$\widehat{L}(\boldsymbol{\alpha}) = -\sum_{i=1}^{n} \ell_*(\alpha_i) - \frac{1}{2\lambda N}\boldsymbol{\alpha}^\top \widehat{G}\boldsymbol{\alpha}$$

We are now ready to give the proof for Theorem 1. The basic logic is straightforward. Since $\widehat{G}$ is close to G, we would expect $\widehat{\boldsymbol{\alpha}}_*$, the optimal solution to $\widehat{L}(\boldsymbol{\alpha})$, to be close to $\boldsymbol{\alpha}_*$, the optimal solution to $L(\boldsymbol{\alpha})$. Since both M_* and $\widehat{M}_*$ are linear in the dual variables $\boldsymbol{\alpha}_*$ and $\widehat{\boldsymbol{\alpha}}_*$, we would expect $\widehat{M}_*$ to be close to M_*.

Since $\widehat{\boldsymbol{\alpha}}_*$ maximizes $\widehat{L}(\boldsymbol{\alpha})$ over its domain, which means $(\boldsymbol{\alpha}_* - \widehat{\boldsymbol{\alpha}}_*)^\top \nabla\widehat{L}(\widehat{\boldsymbol{\alpha}}_*) \leq 0$, we have

$$\widehat{L}(\widehat{\boldsymbol{\alpha}}_*) \geq \widehat{L}(\boldsymbol{\alpha}_*) + \frac{1}{2\lambda N}(\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*)^\top \widehat{G}(\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*) \tag{.2}$$

Using the concaveness of $\widehat{L}(\boldsymbol{\alpha})$ and the fact that $\boldsymbol{\alpha}_*$ maximizes $L(\boldsymbol{\alpha})$ over its domain, we have:

$$\begin{aligned}
& \widehat{L}(\widehat{\boldsymbol{\alpha}}_*) \leq \widehat{L}(\boldsymbol{\alpha}_*) + (\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*)^\top \nabla\widehat{L}(\boldsymbol{\alpha}_*) - \frac{1}{2\lambda N}(\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*)^\top \widehat{G}(\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*) \\
= \;& \widehat{L}(\boldsymbol{\alpha}_*) - \frac{1}{2\lambda N}(\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*)^\top \widehat{G}(\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*) \\
+ \;& (\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*)^\top \left(\nabla\widehat{L}(\boldsymbol{\alpha}_*) - \nabla L(\boldsymbol{\alpha}_*) + \nabla L(\boldsymbol{\alpha}_*)\right) \\
\leq \;& \widehat{L}(\boldsymbol{\alpha}_*) + \frac{1}{\lambda N}(\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*)^\top (G - \widehat{G})(\boldsymbol{\alpha}_*) - \frac{1}{2\lambda N}(\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*)^\top \widehat{G}(\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*) \qquad (.3)
\end{aligned}$$

Combining the inequalities in (.2) and (.3), we have

$$\frac{1}{\lambda N}(\widehat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}_*)^\top (G - \widehat{G})\boldsymbol{\alpha}_* \geq \frac{1}{\lambda N}(\widehat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}_*)^\top \widehat{G}(\widehat{\boldsymbol{\alpha}} - \boldsymbol{\alpha}_*)$$

or

$$(\boldsymbol{\alpha}_* - \widehat{\boldsymbol{\alpha}}_*)^\top (\widehat{G} - G)\boldsymbol{\alpha}_* \geq (\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*)^\top G(\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*) + (\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*)^\top (\widehat{G} - G)(\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*)$$

Define $\mathbf{p}_* = \widetilde{Z}\boldsymbol{\alpha}_*, \widehat{\mathbf{p}}_* = \widetilde{Z}\widehat{\boldsymbol{\alpha}}_*$, we have:

$$(\mathbf{p}_* - \widehat{\mathbf{p}}_*)^\top \Gamma \mathbf{p}_* \geq \|\widehat{\mathbf{p}}_* - \mathbf{p}_*\|_2^2 + (\widehat{\mathbf{p}}_* - \mathbf{p}_*)^\top \Gamma (\widehat{\mathbf{p}}_* - \mathbf{p}_*)$$

Using the bound given in (.1), with a probability $1 - \delta$, we have

$$\|\widehat{\mathbf{p}}_* - \mathbf{p}_*\|_2 \leq \frac{3\varepsilon}{1 - 3\varepsilon}\|\mathbf{p}_*\|_2$$

We complete the proof by using the fact

$$\|M_* - \widehat{M}_*\|_F = \frac{1}{\lambda N}\|\mathbf{p}_* - \widehat{\mathbf{p}}_*\|_2, \ \|M_*\|_F = \frac{1}{\lambda N}\|\mathbf{p}_*\|_2$$

and [99]

$$\|\Pi_{PSD}(M_*) - \Pi_{PSD}(\widehat{M}_*)\|_F \leq \|M_* - \widehat{M}_*\|_F$$

□

Proof of Theorem 4

Proof. Our analysis is based on the following two theorems.

Theorem 10. *(Theorem 2 [10]) Let* $\mathbf{x} \in \mathbb{R}^d$*, and* $\widehat{\mathbf{x}} = R^\top \mathbf{x}/\sqrt{m}$*, where* $R \in \mathbb{R}^{d\times m}$ *is a random matrix whose entries are chosen independently from* $\mathcal{N}(0,1)$*. Then*

$$\Pr\left\{(1-\varepsilon)\|\mathbf{x}\|_2^2 \le \|\widehat{\mathbf{x}}\|_2^2 \le (1+\varepsilon)\|\mathbf{x}\|_2^2\right\} \ge 1 - 2\exp\left(-\frac{m}{4}(\varepsilon^2-\varepsilon^3)\right).$$

Theorem 11. *(Lemma B-1 [69]) Suppose* M *is a real symmetric matrix with non-negative entries, and* E *is a real symmetric matrix such that* $\max_{i,j}|E_{i,j}| \le \xi$*. Then,* $\|E\circ M\|_2 \le \xi\|M\|_2$*, where* $\|\cdot\|_2$ *stands for the spectral norm of matrix and* $E\circ M$ *is the element-wise product between matrices* E *and* M*.*

Define $L(\boldsymbol{\alpha})$ and $\widehat{L}(\boldsymbol{\alpha})$ as

$$L(\boldsymbol{\alpha}) = -\sum_{i=1}^N \ell_*(\alpha_i) - \frac{1}{2\lambda N}\boldsymbol{\alpha}^\top G\boldsymbol{\alpha},$$
$$\widehat{L}(\boldsymbol{\alpha}) = -\sum_{i=1}^N \ell_*(\alpha_i) - \frac{1}{2\lambda N}\boldsymbol{\alpha}^\top \widehat{G}\boldsymbol{\alpha}$$

Since $\ell(z)$ is γ-smooth, we have $\ell_*(\alpha)$ be γ^{-1}-strongly-convex. Using the fact that $\widehat{\boldsymbol{\alpha}}_*$ approximately maximizes $\widehat{L}(\boldsymbol{\alpha})$ with η-suboptimality and $\ell_*(\cdot)$ is γ^{-1}-strongly-convex, we have

$$\widehat{L}(\widehat{\boldsymbol{\alpha}}_*) \ge \widehat{L}(\boldsymbol{\alpha}_*) + \frac{1}{2\lambda N}(\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*)^\top(\gamma^{-1}I + \widehat{G})(\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*) - \eta \tag{.4}$$

Using the concaveness of $\widehat{L}(\boldsymbol{\alpha})$ and the fact that $\boldsymbol{\alpha}_*$ maximizes $L(\boldsymbol{\alpha})$ over its domain, we

have:

$$\widehat{L}(\widehat{\boldsymbol{\alpha}}_*) \leq \widehat{L}(\boldsymbol{\alpha}_*)$$
$$+\frac{1}{\lambda N}(\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*)^\top (G - \widehat{G})\boldsymbol{\alpha}_* - \frac{1}{2\lambda N}(\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*)^\top (\gamma^{-1} I + \widehat{G})(\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*) \qquad (.5)$$

Combining the inequalities in (.4) and (.5), we have

$$\eta + (\boldsymbol{\alpha}_* - \widehat{\boldsymbol{\alpha}}_*)^\top (\widehat{G} - G)\boldsymbol{\alpha}_* \geq \frac{1}{\gamma}\|\widehat{\boldsymbol{\alpha}}_* - \boldsymbol{\alpha}_*\|_2^2$$

when we set $\lambda = 1/N$. Using the fact $(\boldsymbol{\alpha}_* - \widehat{\boldsymbol{\alpha}}_*)(\widehat{G} - G)\boldsymbol{\alpha}_* \leq \|\boldsymbol{\alpha}_* - \widehat{\boldsymbol{\alpha}}_*\|_2 \|\boldsymbol{\alpha}_*\|_2 \|\widehat{G} - G\|_2$, we have

$$\|\boldsymbol{\alpha}_* - \widehat{\boldsymbol{\alpha}}_*\|_2^2 \leq \gamma \|\boldsymbol{\alpha}_* - \widehat{\boldsymbol{\alpha}}_*\|_2 \|\boldsymbol{\alpha}_*\|_2 \|\widehat{G} - G\|_2 + \gamma\eta,$$

implying that

$$\|\boldsymbol{\alpha}_* - \widehat{\boldsymbol{\alpha}}_*\|_2 \leq \max\left(2\gamma \|\widehat{G} - G\|_2 \|\boldsymbol{\alpha}_*\|_2, \sqrt{2\gamma\eta}\right). \qquad (.6)$$

To bound $\|\boldsymbol{\alpha}_* - \widehat{\boldsymbol{\alpha}}_*\|_2$, we need to bound $\|\widehat{G} - G\|_2$. To this end, we write the $G_{a,b}$ as

$$\begin{aligned} & G_{a,b} = \langle A_a, A_b \rangle \\ = & \left[(\mathbf{x}_i^a - \mathbf{x}_k^a)^\top (\mathbf{x}_i^b - \mathbf{x}_k^b)\right]^2 + \left[(\mathbf{x}_i^a - \mathbf{x}_j^a)^\top (\mathbf{x}_i^b - \mathbf{x}_j^b)\right]^2 \\ - & \left[(\mathbf{x}_i^a - \mathbf{x}_k^a)^\top (\mathbf{x}_i^b - \mathbf{x}_j^b)\right]^2 - \left[(\mathbf{x}_i^a - \mathbf{x}_j^a)^\top (\mathbf{x}_i^b - \mathbf{x}_k^b)\right]^2 \end{aligned}$$

Similarly, we write $\widehat{G}_{a,b}$ as

$$\begin{aligned}
&\widehat{G}_{a,b} = \langle R^\top A_a R, R^\top A_b R \rangle \\
=&\Big[(\mathbf{x}_i^a - \mathbf{x}_k^a)^\top R R^\top (\mathbf{x}_i^b - \mathbf{x}_k^b)\Big]^2 + \Big[(\mathbf{x}_i^a - \mathbf{x}_j^a)^\top R R^\top (\mathbf{x}_i^b - \mathbf{x}_j^b)\Big]^2 \\
-&\Big[(\mathbf{x}_i^a - \mathbf{x}_k^a)^\top R R^\top (\mathbf{x}_i^b - \mathbf{x}_j^b)\Big]^2 - \Big[(\mathbf{x}_i^a - \mathbf{x}_j^a)^\top R R^\top (\mathbf{x}_i^b - \mathbf{x}_k^b)\Big]^2
\end{aligned}$$

Hence, we can write $\widehat{G} - G = B^1 + B^2 + B^3 + B^4$, where B^1, B^2, B^3, and B^4 are defined as

$$\begin{aligned}
B^1_{a,b} =&\Big[(\mathbf{x}_i^a - \mathbf{x}_k^a)^\top R R^\top (\mathbf{x}_i^b - \mathbf{x}_k^b)\Big]^2 - \Big[(\mathbf{x}_i^a - \mathbf{x}_k^a)^\top (\mathbf{x}_i^b - \mathbf{x}_k^b)\Big]^2 \\
B^2_{a,b} =&\Big[(\mathbf{x}_i^a - \mathbf{x}_j^a)^\top R R^\top (\mathbf{x}_i^b - \mathbf{x}_j^b)\Big]^2 - \Big[(\mathbf{x}_i^a - \mathbf{x}_j^a)^\top (\mathbf{x}_i^b - \mathbf{x}_j^b)\Big]^2 \\
B^3_{a,b} =&\Big[(\mathbf{x}_i^a - \mathbf{x}_k^a)^\top (\mathbf{x}_i^b - \mathbf{x}_j^b)\Big]^2 - \Big[(\mathbf{x}_i^a - \mathbf{x}_k^a)^\top R R^\top (\mathbf{x}_i^b - \mathbf{x}_j^b)\Big]^2 \\
B^4_{a,b} =&\Big[(\mathbf{x}_i^a - \mathbf{x}_j^a)^\top (\mathbf{x}_i^b - \mathbf{x}_k^b)\Big]^2 - \Big[(\mathbf{x}_i^a - \mathbf{x}_j^a)^\top R R^\top (\mathbf{x}_i^b - \mathbf{x}_k^b)\Big]^2
\end{aligned}$$

Using the result from Theorem 10 and the definition of matrices M^1, M^2, M^3, M^4, we have, with a probability $1 - \delta$, for any a, b,

$$|B^i_{a,b}| \leq \epsilon M^i_{a,b},\ i = \{1, 2, 3, 4\}$$

provided that $\epsilon \leq 1/2$ and

$$m \geq \frac{8}{\epsilon^2} \ln \frac{8N}{\delta} \tag{.7}$$

Using Theorem 11, under the condition in (.7), we have, with a probability $1-\delta$,

$$\|\widehat{G}-G\|_2 \leq \epsilon \left(\|M^1\| + \|M^2\| + \|M^3\| + \|M^4\| \right) \leq 4\kappa\epsilon$$

where the last step uses the definition of κ. We complete the proof by plugging the bound for $\|\widehat{G} - G\|_2$ into (.6). □

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] David W. Aha, Dennis F. Kibler, and Marc K. Albert. Instance-based learning algorithms. *ML*, 6:37–66, 1991.

[2] Anelia Angelova and Shenghuo Zhu. Efficient object detection and segmentation for fine-grained recognition. In *CVPR*, 2013.

[3] Oren Barkan, Jonathan Weill, Lior Wolf, and Hagai Aronowitz. Fast high dimensional vector multiplication face recognition. In *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*, pages 1960–1967, 2013.

[4] Hila Becker, Mor Naaman, and Luis Gravano. Learning similarity metrics for event identification in social media. In *WSDM*, pages 291–300, 2010.

[5] Ron Bekkerman and Martin Scholz. Data weaving: scaling up the state-of-the-art in data clustering. In *CIKM*, pages 1083–1092, 2008.

[6] Aurélien Bellet and Amaury Habrard. Robustness and generalization for metric learning. *CoRR*, abs/1209.1086, 2012.

[7] Alex Berg, Ryan Farrell, Aditya Khosla, Jonathan Krause, Fei-Fei Li, Jia Li, and Subhransu Maji. Fine-grained challenge 2013. 2013.

[8] Thomas Berg and Peter N. Belhumeur. Poof: Part-based one-vs-one features for fine-grained categorization, face verification, and attribute estimation. In *CVPR*, 2013.

[9] Thomas Berg and Peter N. Belhumeur. POOF: part-based one-vs.-one features for fine-grained categorization, face verification, and attribute estimation. In *CVPR*, pages 955–962, 2013.

[10] Avrim Blum. Random projection, margins, kernels, and feature-selection. In *SLSFS*, pages 52–68, 2005.

[11] Christos Boutsidis, Anastasios Zouzias, and Petros Drineas. Random projections for k-means clustering. In *NIPS*, pages 298–306, 2010.

[12] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[13] Lev M Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR computational mathematics and mathematical physics*, 7(3):200–217, 1967.

[14] Deng Cai, Xuanhui Wang, and Xiaofei He. Probabilistic dyadic data analysis with local and global consistency. In *ICML*, pages 105–112, 2009.

[15] Colin Campbell. Kernel methods: a survey of current techniques. *Neurocomputing*, 48(1-4):63–84, 2002.

[16] Qiong Cao, Zheng-Chu Guo, and Yiming Ying. Generalization bounds for metric and similarity learning. *CoRR*, abs/1207.5437, 2012.

[17] Qiong Cao, Yiming Ying, and Peng Li. Similarity metric learning for face recognition. In *ICCV*, pages 2408–2415, 2013.

[18] Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[19] Yuning Chai, Victor S. Lempitsky, and Andrew Zisserman. Bicos: A bi-level co-segmentation method for image classification. In *ICCV*, pages 2579–2586, 2011.

[20] Yuning Chai, Victor S. Lempitsky, and Andrew Zisserman. Symbiotic segmentation and part localization for fine-grained categorization. In *ICCV*, pages 321–328, 2013.

[21] Yuning Chai, Esa Rahtu, Victor S. Lempitsky, Luc J. Van Gool, and Andrew Zisserman. Tricos: A tri-level class-discriminative co-segmentation method for image classification. In *ECCV*, pages 794–807, 2012.

[22] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM TIST*, 2(3):27, 2011.

[23] Hong Chang and Dit-Yan Yeung. Locally linear metric adaptation for semi-supervised clustering. In *ICML*, pages 153–160, 2004.

[24] Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. Large scale online learning of image similarity through ranking. *JMLR*, 11:1109–1135, 2010.

[25] Dong Chen, Xudong Cao, Fang Wen, and Jian Sun. Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification. In *CVPR*, pages 3025–3032, 2013.

[26] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA*, pages 539–546, 2005.

[27] Yang Cong, Junsong Yuan, and Yandong Tang. Object tracking via online metric learning. In *ICIP*, pages 417–420, 2012.

[28] Andrew Cotter, Ohad Shamir, Nati Srebro, and Karthik Sridharan. Better mini-batch algorithms via accelerated gradient methods. In *NIPS*, pages 1647–1655, 2011.

[29] Zhen Cui, Wen Li, Dong Xu, Shiguang Shan, and Xilin Chen. Fusing robust face region descriptors via multiple metric learning for face recognition in the wild. In *CVPR*, pages 3554–3561, 2013.

[30] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pages 886–893, 2005.

[31] Jason V. Davis and Inderjit S. Dhillon. Structured metric learning for high dimensional problems. In *KDD*, pages 195–203, 2008.

[32] Jason V. Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S. Dhillon. Information-theoretic metric learning. In *ICML*, pages 209–216, 2007.

[33] Huyen Do, Alexandros Kalousis, Jun Wang, and Adam Woznica. A metric learning perspective of SVM: on the relation of LMNN and SVM. In *AISTATS*, pages 308–317, 2012.

[34] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, pages 647–655, 2014.

[35] Marco F. Duarte and Yu Hen Hu. Vehicle classification in distributed sensor networks. *J. Parallel Distrib. Comput.*, 64(7):826–838, 2004.

[36] Sandra Ebert, Mario Fritz, and Bernt Schiele. Active metric learning for object recognition. In *Pattern Recognition*, pages 327–336, 2012.

[37] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *JMLR*, 9:1871–1874, 2008.

[38] Zheyun Feng, Rong Jin, and Anil K. Jain. Large-scale image annotation by efficient and robust kernel metric learning. In *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*, pages 1609–1616, 2013.

[39] Dmitriy Fradkin and David Madigan. Experiments with random projections for machine learning. In *KDD*, pages 517–522, 2003.

[40] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[41] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

[42] Andrea Frome, Yoram Singer, Fei Sha, and Jitendra Malik. Learning globally-consistent local distance functions for shape-based image retrieval and classification. In *ICCV*, pages 1–8, 2007.

[43] Dan Garber and Elad Hazan. Faster rates for the frank-wolfe method over strongly-convex sets. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 541–549, 2015.

[44] Efstratios Gavves, Basura Fernando, Cees G. M. Snoek, Arnold W. M. Smeulders, and Tinne Tuytelaars. Fine-grained categorization by alignments. In *ICCV*, pages 1713–1720, 2013.

[45] Amir Globerson and Sam T. Roweis. Metric learning by collapsing classes. In *NIPS*, pages 451–458, 2005.

[46] Jacob Goldberger, Sam T. Roweis, Geoffrey E. Hinton, and Ruslan Salakhutdinov. Neighbourhood components analysis. In *NIPS*, 2004.

[47] Gene Howard Golub and Charles Van Loan. Matrix computations. 1989.

[48] Greg Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset, 2007.

[49] Matthieu Guillaumin, Thomas Mensink, Jakob J. Verbeek, and Cordelia Schmid. Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation. In *ICCV*, pages 309–316, 2009.

[50] Matthieu Guillaumin, Jakob J. Verbeek, and Cordelia Schmid. Is that you? metric learning approaches for face identification. In *ICCV*, pages 498–505, 2009.

[51] Zheng-Chu Guo and Yiming Ying. Guaranteed classification via regularized similarity learning. *Neural Computation*, 26(3):497–522, 2014.

[52] N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *ArXiv e-prints*, September 2009.

[53] John A Hartigan. Clustering algorithms. 1975.

[54] Trevor Hastie and Robert Tibshirani. Discriminant adaptive nearest neighbor classification. *IEEE Trans. Pattern Anal. Mach. Intell.*, 18(6):607–616, 1996.

[55] Elad Hazan and Satyen Kale. Projection-free online learning. In *ICML*, 2012.

[56] Xiaofei He, Wei-Ying Ma, and HongJiang Zhang. Learning an image manifold for retrieval. In *ACM Multimedia*, pages 17–23, 2004.

[57] Steven C. H. Hoi, Wei Liu, and Shih-Fu Chang. Semi-supervised distance metric learning for collaborative image retrieval. In *CVPR*, 2008.

[58] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *IEEE Trans. on Neural Netw.*, 13(2):415–425, 2002.

[59] Junlin Hu, Jiwen Lu, and Yap-Peng Tan. Discriminative deep metric learning for face verification in the wild. In *CVPR*, pages 1875–1882, 2014.

[60] Junlin Hu, Jiwen Lu, Junsong Yuan, and Yap-Peng Tan. Large margin multi-metric learning for face and kinship verification in the wild. In *Computer Vision - ACCV 2014 - 12th Asian Conference on Computer Vision, Singapore, Singapore, November 1-5, 2014, Revised Selected Papers, Part III*, pages 252–267, 2014.

[61] Chang Huang, Shenghuo Zhu, and Kai Yu. Large scale strongly supervised ensemble metric learning, with applications to face verification and retrieval. *CoRR*, abs/1212.6094, 2012.

[62] Gary B. Huang, Honglak Lee, and Erik G. Learned-Miller. Learning hierarchical representations for face verification with convolutional deep belief networks. In *2012 IEEE*

Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012, pages 2518–2525, 2012.

[63] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[64] Ratnawati Ibrahim and Zalhan Mohd Zin. Study of automated face recognition system for office door access control application. In *ICCSN*, pages 132–136. IEEE, 2011.

[65] Satoshi Ito and Susumu Kubota. Object classification using heterogeneous co-occurrence features. In *ECCV*, pages 209–222, 2010.

[66] Prateek Jain, Brian Kulis, Inderjit S. Dhillon, and Kristen Grauman. Online metric learning and fast similarity search. In *NIPS*, pages 761–768, 2008.

[67] Rong Jin. Stochastic optimization of smooth loss. In *arXiv:1312.0048*, 2013.

[68] Rong Jin, Shijun Wang, and Yang Zhou. Regularized distance metric learning: Theory and algorithm. In *NIPS*, pages 862–870, 2009.

[69] Noureddine El Karoui. The spectrum of kernel random matrices. In *The Annals of Statistics*, 2010.

[70] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-fei Li. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, CVPR*, 2011.

[71] Martin Köstinger, Martin Hirzer, Paul Wohlhart, Peter M. Roth, and Horst Bischof. Large scale metric learning from equivalence constraints. In *CVPR*, pages 2288–2295, 2012.

[72] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1106–1114, 2012.

[73] Brian Kulis. Metric learning: A survey. *Foundations and Trends in Machine Learning*, 5(4):287–364, 2013.

[74] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. Sparse distance learning for object recognition combining rgb and depth information. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4007–4013. IEEE, 2011.

[75] Guanghui Lan. The complexity of large-scale convex programming under a linear optimization oracle. *arXiv preprint arXiv:1309.5550*, 2013.

[76] Marc Teva Law, Nicolas Thome, and Matthieu Cord. Quadruplet-wise image similarity learning. In *ICCV*, pages 249–256, 2013.

[77] Marc Teva Law, Nicolas Thome, and Matthieu Cord. Fantope regularization in metric learning. In *CVPR*, pages 1051–1058, 2014.

[78] Daryl Lim, Gert Lanckriet, and Brian McFee. Robust structural metric learning. In *ICML*, 2013.

[79] Prasanta Chandra Mahalanobis. On the generalized distance in statistics. *Proceedings of the National Institute of Sciences (Calcutta)*, 2:49–55, 1936.

[80] M. Mahdavi, T. Yang, R. Jin, S. Zhu, and J. Yi. Stochastic gradient descent with only one projection. In *NIPS*, pages 503–511, 2012.

[81] Odalric-Ambrym Maillard and Remi Munos. Linear regression with random projections. In *JMLR*, volume 13, 2012.

[82] Thomas Mensink, Jakob J. Verbeek, Florent Perronnin, and Gabriela Csurka. Metric learning for large scale image classification: Generalizing to new classes at near-zero cost. In *ECCV*, pages 488–501, 2012.

[83] Tom M. Mitchell. *Machine learning*. McGraw Hill series in computer science. McGraw-Hill, 1997.

[84] Yang Mu, Wei Ding, and Dacheng Tao. Local discriminative distance metrics ensemble learning. *Pattern Recognition*, 46(8):2337–2349, 2013.

[85] M-E. Nilsback. *An Automatic Visual Flora – Segmentation and Classification of Flowers Images*. PhD thesis, University of Oxford, 2009.

[86] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *ICVGIP*, pages 722–729, 2008.

[87] Dan Oneata, Jakob J. Verbeek, and Cordelia Schmid. Action and event recognition with fisher vectors on a compact feature set. In *ICCV*, pages 1817–1824, 2013.

[88] Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *CVPR*, pages 3498–3505, 2012.

[89] Guo-Jun Qi, Jinhui Tang, Zheng-Jun Zha, Tat-Seng Chua, and Hong-Jiang Zhang. An efficient sparse metric learning in high-dimensional space via l_1-penalized log-determinant regularization. In *ICML*, page 106, 2009.

[90] Qi Qian, Rong Jin, Shenghuo Zhu, and Yuanqing Lin. An integrated framework for high dimensional distance metric learning and its application to fine-grained visual categorization. *CoRR*, abs/1402.0453, 2014.

[91] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *NIPS*, 2007.

[92] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge, 2014.

[93] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, pages 213–226, 2010.

[94] Lawrence K. Saul and Sam T. Roweis. Think globally, fit locally: Unsupervised learning of low dimensional manifold. *JMLR*, 4:119–155, 2003.

[95] Bernhard Schölkopf, Alex J. Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[96] Shai Shalev-Shwartz, Yoram Singer, and Andrew Y. Ng. Online and batch learning of pseudo-metrics. In *ICML*, 2004.

[97] Shai Shalev-Shwartz and Tong Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *CoRR*, abs/1209.1873, 2012.

[98] Blake Shaw, Bert C. Huang, and Tony Jebara. Learning a distance metric from a network. In *NIPS*, pages 1899–1907, 2011.

[99] Henry Stark, Yongi Yang, and Yongyi Yang. *Vector space projections: a numerical approach to signal and image processing, neural nets, and optics*. John Wiley & Sons, Inc., 1998.

[100] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 1701–1708, 2014.

[101] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[102] Lorenzo Torresani and Kuang-chih Lee. Large margin component analysis. In *NIPS*, pages 1385–1392, 2006.

[103] Grigorios Tsagkatakis and Andreas E. Savakis. Manifold modeling with learned distance in random projection space for face recognition. In *ICPR*, pages 653–656, 2010.

[104] Grigorios Tsagkatakis and Andreas E. Savakis. Online distance metric learning for object tracking. *IEEE Trans. Circuits Syst. Video Techn.*, 21(12):1810–1821, 2011.

[105] Oncel Tuzel, Fatih Porikli, and Peter Meer. Pedestrian detection via classification on riemannian manifolds. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(10):1713–1727, 2008.

[106] Vladimir Vapnik. *The nature of statistical learning theory*. springer, 2000.

[107] Vladimir Naumovich Vapnik and Vlamimir Vapnik. *Statistical learning theory*, volume 2. Wiley New York, 1998.

[108] Yashaswi Verma and C. V. Jawahar. Image annotation using metric learning in semantic neighbourhoods. In *ECCV*, pages 836–849, 2012.

[109] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, 2011.

[110] Fei Wang, Jimeng Sun, and Shahram Ebadollahi. Integrating distance metrics learned from multiple experts and its application in inter-patient similarity assessment. In *SDM*, pages 59–70, 2011.

[111] Fei Wang, Jimeng Sun, Jianying Hu, and Shahram Ebadollahi. imet: Interactive metric learning in healthcare applications. In *SDM*, pages 944–955, 2011.

[112] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. In *CVPR*, pages 1386–1393, 2014.

[113] Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas S. Huang, and Yihong Gong. Locality-constrained linear coding for image classification. In *CVPR*, pages 3360–3367, 2010.

[114] Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *JMLR*, 10:207–244, 2009.

[115] Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart J. Russell. Distance metric learning with application to clustering with side-information. In *NIPS*, pages 505–512, 2002.

[116] Zhixiang Eddie Xu, Kilian Q. Weinberger, and Olivier Chapelle. Distance metric learning for kernel machines. *CoRR*, abs/1208.3422, 2012.

[117] Liu Yang and Rong Jin. Distance metric learning: A comprehensive survey. *[Online]*.

[118] Shulin Yang, Liefeng Bo, Jue Wang, and Linda G. Shapiro. Unsupervised template learning for fine-grained object recognition. In *NIPS*, pages 3131–3139, 2012.

[119] Jian Zhang, Rong Jin, Yiming Yang, and Alexander G. Hauptmann. Modified logistic regression: An approximation to SVM and its applications in large-scale text categorization. In *ICML*, pages 888–895, 2003.

[120] Lijun Zhang, Mehrdad Mah-davi, Rong Jin, Tian-Bao Yang, and Shenghuo Zhu. Recovering optimal solution by dual random projection. In *arXiv:1211.3046*, 2013.

[121] Ning Zhang, Ryan Farrell, Forrest N. Iandola, and Trevor Darrell. Deformable part descriptors for fine-grained recognition and attribute prediction. In *ICCV*, pages 729–736, 2013.

[122] T. Zhang and F.J. Oles. Text categorization based on regularized linear classification methods. *Information Retrieval*, 4(1):5–31, 2001.

[123] Xiaojin Zhu and Andrew B. Goldberg. *Introduction to Semi-Supervised Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2009.