Geometry-Aware Learning Algorithms for Histogram Data Using Adaptive Metric Embeddings and Kernel Functions

Le Thanh Tam

Department of Intelligent Science and Technology
Graduate School of Informatics
Kyoto University

This dissertation is submitted for the degree of
Doctor of Informatics

December 2015

Acknowledgements

First of all, I would like to express my deepest gratitude to my supervisors Professor Akihiro Yamamoto and Professor Marco Cuturi. Their advices and support were very important to help me improve my research skills and gain experience during my PhD program.

Secondly, I would like to thank Professor Zaid Harchaoui for his advices during my internship at Lear team, INRIA, France. Additionally, I am also grateful to all members of the Lear team, especially Professor Cordelia Schmid, for their support. I gained many valuable experiences throughout conversations and seminars in Lear, INRIA.

Thirdly, I would like to thank Professor Ryo Yoshinaka and my colleagues in the Yamamoto-Cuturi laboratory for their support and encouragement on various aspects during my PhD program.

Moreover, I also gratefully acknowledge the financial support of Japanese Government (Monbukagakusho) Scholarships. It gave me a chance to pursue a PhD program in Japan.

Finally, I would like to thank my family, my wife Le Bao Tran and my son Le Nhat Minh, for their support during my PhD studies.

Abstract

Many algorithms in machine learning rely on a notion of similarity between input objects to carry out inference. In most cases, the performance of algorithms improves when such a similarity measure between objects of interest is learned, rather than simply chosen a priori. Among such objects of interests, histograms – the normalized representation for bags-of-features (BoF) – play a fundamental role since they appear in many research fields such as computer vision, natural language processing, and speech processing. Ample empirical evidence shows that the Euclidean geometry is not the best choice to measure similarity between histograms in the simplex – the set of nonnegative and normalized vectors. In this thesis, we propose new methodologies within the framework of metric learning and kernel methods to quantify adaptively similarity for histograms.

This thesis proposes first to learn a metric for histograms by generalizing a family of embeddings proposed by Aitchison (1982). These embeddings map histograms in the probability simplex onto a suitable Euclidean space. Instead of relying on a few mappings defined by a priori, such as those proposed in Aitchison (1982), we provide algorithms to *learn* such maps using labeled histograms by building upon previous work in metric learning. These algorithms lead to representations that outperform alternative approaches to compare histograms in a variety of contexts.

This thesis proposes next to learn a Riemannian metric on the simplex using only *unlabelled* histograms. We follow the approach of Lebanon (2006), who proposed to estimate such a metric within a parametric family by maximizing the inverse volume of a given data set of points under that metric. The metrics we consider on the simplex are the Fisher information metrics parameterized by Aitchison's transformations in the simplex. We propose an algorithmic approach to maximize the inverse volume using sampling and contrastive divergences. We also provide experimental evidence that the metric obtained under our proposal outperforms alternative approaches.

Finally, this thesis proposes a kernel that measures similarity between images using the BoF model approach for representation. We build upon spatial pyramid matching, an efficient extension of the BoF model. This method partitions images into increasingly fine sub-regions, and measures similarity between these sub-regions by applying the BoF model, which is

limited in its capacity to quantify similarity between sets of unordered features. We propose a hierarchical spatial matching kernel that uses a coarse-to-fine model for the sub-regions to obtain better similarity measure. In experiments, our proposed kernel improves performances of the spatial pyramid matching kernel on many benchmark datasets.

In conclusion, learning algorithms for general vectors may not work well in practice when dealing with input structured data such as metric learning for histograms. Each structured data has its own geometry. So, geometry-aware learning algorithms for structured data is a promising direction for future work, for example, metric learning for structured data such as trees, graphs, strings, time series, molecules or materials. Additionally, in metric learning for histograms, some traditional distances for histograms such as the χ^2 distance and the transportation distance are used instead of the Mahalanobis distance in metric learning for general vectors. Therefore, a fruitful direction for future work is to explore an appropriate distance or divergence for a given structured data in metric learning.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

In this chapter, we give an introduction about histogram representation, metric learning and kernel functions. We introduce the bag-of-features representation for complex objects, which is popular in many research fields. Then, we review metric learning and kernel functions, which are our approaches to quantify adaptively similarity for histogram data.

1.1 Histograms

1.1.1 Bag of Features Representation

The *bag of features* is a popular representation for complex objects in many different research fields such as computer vision (Julesz, 1981; Sivic and Zisserman, 2003; Perronnin et al., 2010; Vedaldi and Zisserman, 2012), natural language processing (Salton and McGill, 1983; Salton, 1989; Joachims, 2002; Blei et al., 2003; Blei and Lafferty, 2006, 2009), speech processing (Doddington, 2001; Campbell et al., 2003; Campbell and Richardson, 2007) and bioinformatics (Erhan et al., 1980; Burge et al., 1992; Leslie et al., 2002). For each object of interest, we extract a set of features. Then, we count occurrences of each type of features to form a count vector for each object representation. Another way to represent each object is that, from the set of features, we compute a frequency for each type of features to form a histogram vector to represent each object of interest. However, if the set of all features has a very large cardinality, we need to quantize the space of features into a small discrete set since the cardinality of the set will be the dimension of the histograms. Then, we can form a histogram of frequencies for these quantized features to represent each object of interest. A histogram is a vector which has nonnegative elements and which sums to 1.

In the next subsections, we list two main fields, natural language processing and computer vision, where histograms are popularly used to represent objects of interest (Julesz, 1981;

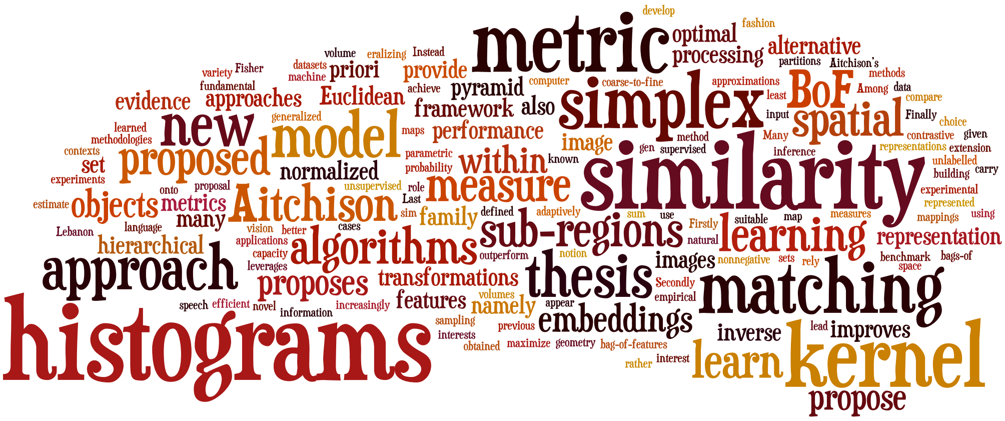


Fig. 1.1 An illustration of a word cloud – a visual representation of text data – for this thesis's abstract. The more frequently a word appears, the more prominently it is displayed.

Salton and McGill, 1983; Salton, 1989; Joachims, 2002; Blei et al., 2003; Sivic and Zisserman, 2003; Blei and Lafferty, 2006, 2009; Perronnin et al., 2010; Vedaldi and Zisserman, 2012).

1.1.2 Histograms in Natural Language Processing

In natural language processing, objects of interest are text documents. A simple way to represent a document is to discard the order of words in the document, gather all words into a *bag*, and then compute a histogram vector of word frequencies to construct a representation for the document. This approach is known as the *bag of word* representation for text documents (Salton and McGill, 1983; Joachims, 2002). In this approach, words are considered as features. For example, Figure 1.1 illustrates a word cloud, which is a visual representation of text data, for this thesis's abstract. The more frequently a word appears, the more prominently it is displayed. Typically, the number of words in a dictionary is more than tens of thousands. Consequently, a histogram, representing a document, has a high dimension that equals to the number of different words.

To embed histograms in a low-dimensional space, we need a higher-level feature than words. An idea is to represent text documents by a mixture of *topics*, which are each a distribution of words. Intuitively, we can consider that each word in a document is generated by one of the document's topics. This research problem is known as topic model, studied for instance in the Latent Dirichlet Allocation (LDA) approach (Blei et al., 2003; Teh et al., 2006; Newman et al., 2007; Hoffman et al., 2010). The number of topics is usually much smaller than the number of words. Therefore, the dimension of a histogram of topics is

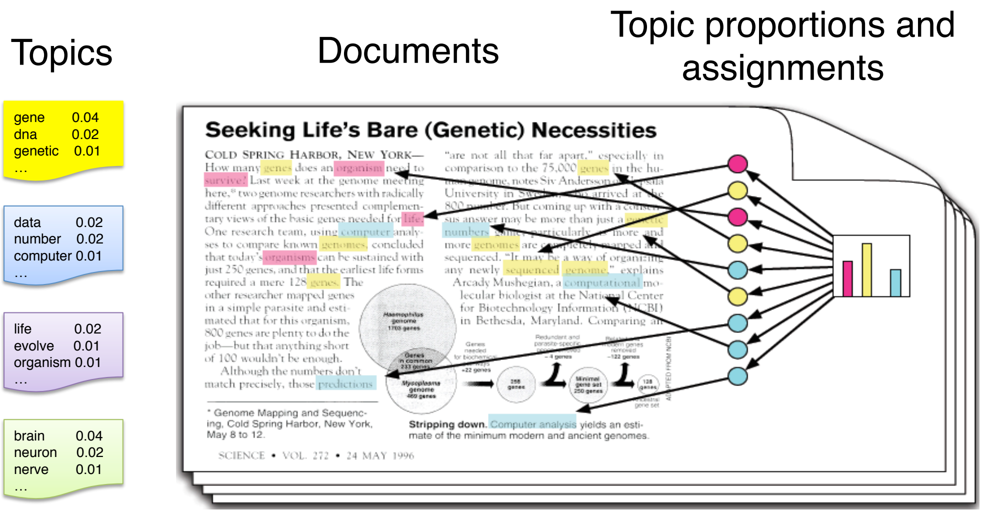


Fig. 1.2 An illustration of using topic modeling to represent a document as a histogram of topics. Topic is a distribution of words such as the yellow box (genetics), the blue box (computer science), the purple box (ecology) and the green box (neuroscience). Each word in a document is assumed to be generated either of these topics. We can also represent each document as the histogram of topics inferred by the topic model. (Edited from David Blei's figure.)

much smaller than that of a histogram of words. Figure 1.2 is an illustration of using the topic model to represent a document as the histogram of topics. In this illustration, we have some topics such as genetics (the yellow box), computer science (the blue box), ecology (the purple box) and neuroscience (the green box). Each word in a document is assumed to be generated either of these topics. We can also obtain each document representation as the histogram of topics, inferred by the topic model.

1.1.3 Histograms in Computer Vision

In computer vision, objects of interest are images (videos can be considered as sequences of images). Researchers in this area have adapted the *bag of words* representation in natural language processing to propose the *bag of visual words* representation (Julesz, 1981; Sivic and Zisserman, 2003; Perronnin et al., 2010; Vedaldi and Zisserman, 2012) for images. For each image, they extract features such as color, SIFT (Lowe, 2004) or SURF (Bay et al., 2006). Then, each image can be considered as a set of features. They use a histogram of features to represent each image, such as a histogram of pixel colors illustrated in Figure 1.3.

In practice, a space of features is usually very large, the researchers in this area quantize the space into a much smaller discrete set. For example, they use k-means to cluster all

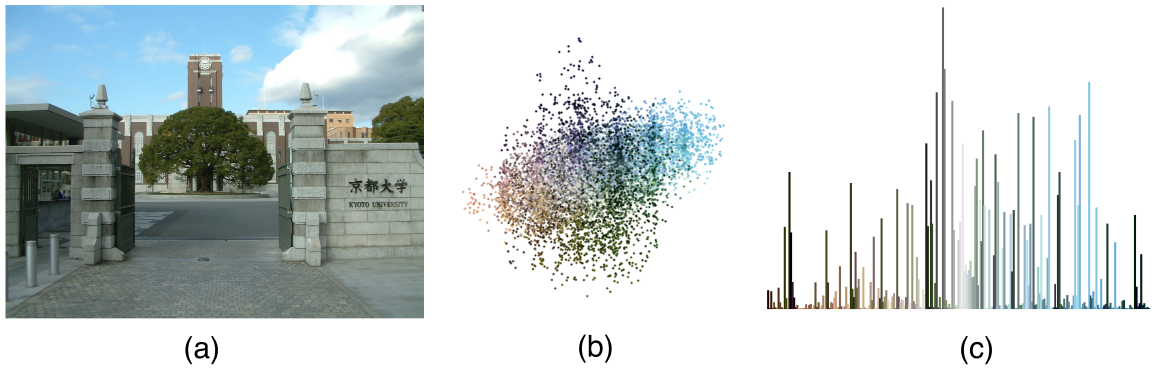


Fig. 1.3 An illustration of a histogram of colors representation for an image. From the given image (a), we discard the location of pixel colors and consider it as a set of pixel colors (b). Then, we compute a histogram of pixel colors frequencies to represent the image (c).

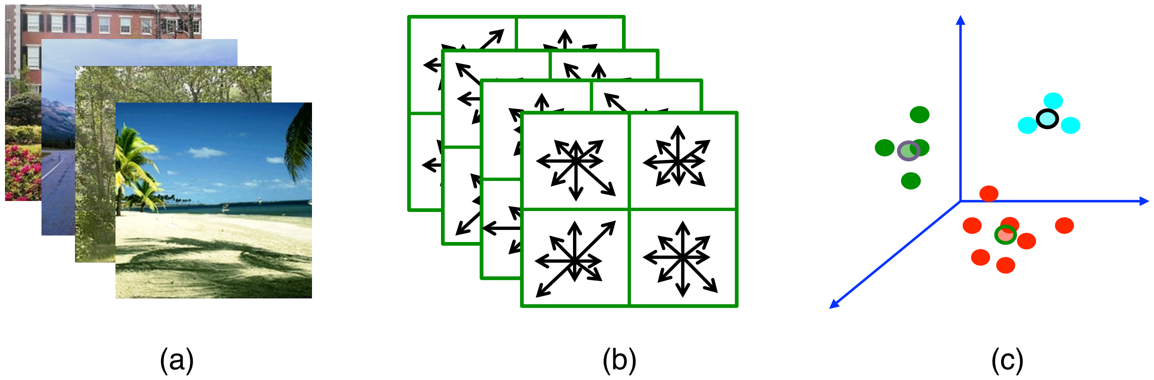


Fig. 1.4 Illustration of building visual words in the bag of visual words representation. From a set of images (a), features are extracted from each image such as SIFT or SURF (b). Each feature is considered as a high-dimensional data point. We can use a clustering method such as k-means to cluster all features extracted from the set of all images. Then, each centroid is regarded as a *visual word* (c).

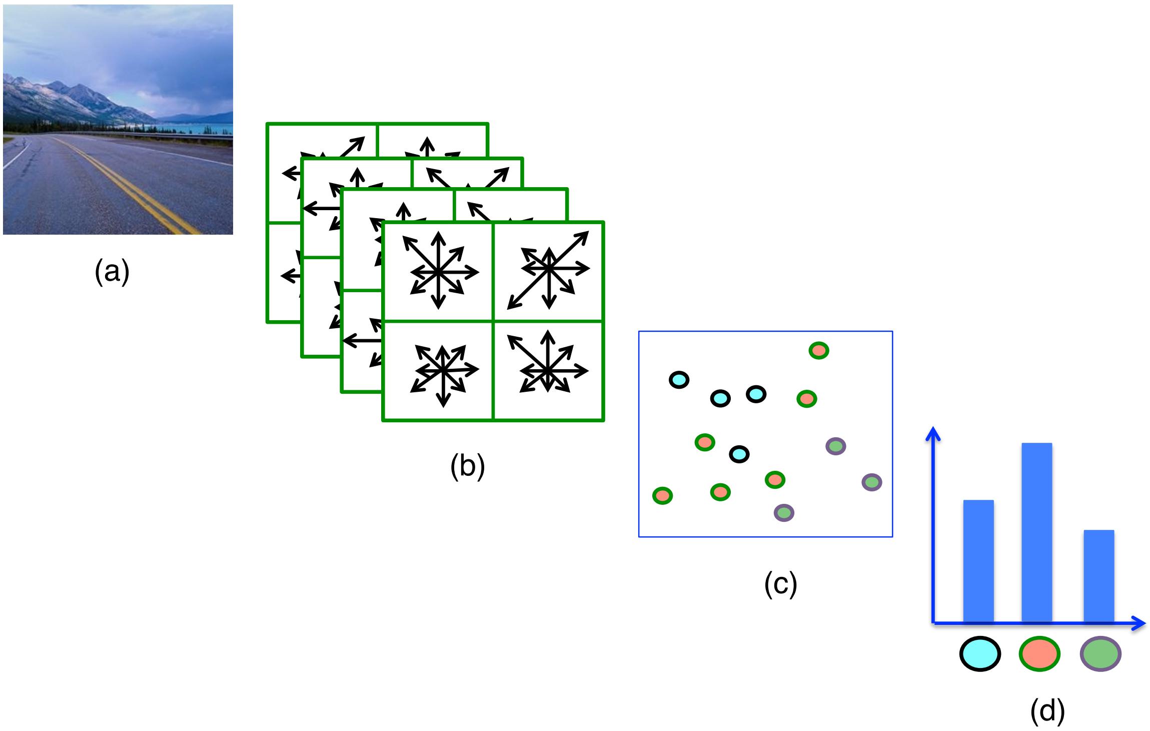


Fig. 1.5 Illustration of a histogram of visual words representation for an image. For each given image (a), we extract features (b). Each feature is then approximated by its nearest centroid (visual word). So, each image can be regarded as a set of visual words (c). Then, we can compute a histogram of visual words frequencies to represent each image (d).

features in an image dataset. Each centroid of k-means is regarded as a *visual word*. This procedure of building visual words is illustrated in Figure 1.4. Then, the features in each image can be approximated by their nearest centroid (visual word). Therefore, they can construct a histogram representation for each image by computing frequencies of *visual words* as illustrated in Figure 1.5.

1.2 Metric Learning for General Vectors

1.2.1 Motivation of Metric Learning

Many algorithms in machine learning rely on a distance between data points such as k-means, k-medoids in clustering or k-nearest neighbors in classification. However, choosing a right distance is difficult when input data are complex. So, distances should be adaptively *learned* from dataset, rather than simply chosen a priori.

Consider a toy dataset as in Figure 1.6(a), there are two features for each data point. The first feature is color and the second one is shape. We would like to cluster the toy dataset into two groups by using the squared Mahalanobis distance. Recall that the squared Mahalanobis distance between two vectors $\mathbf{x}$ and $\mathbf{z}$ is

$$d^2_{\mathbf{M}}(\mathbf{x},\mathbf{z}) = (\mathbf{x}-\mathbf{z})^T \mathbf{M} (\mathbf{x}-\mathbf{z}), \tag{1.1}$$

where $\mathbf{M}$ is a positive semidefinite matrix. $d^2_{\mathbf{M}}$ is equal to the squared Euclidean distance when $\mathbf{M}$ is the identity matrix. For the first scenario, we would like to obtain a result of clustering as in Figure 1.6(b). Or, we want to cluster the toy dataset by only focusing on the color feature (the first feature). A suitable value of matrix $\mathbf{M}$ is

$$\mathbf{M} = \begin{pmatrix} 1 & 0 \\ 0 & \varepsilon \end{pmatrix},$$

where ε is a very small positive real value. Therefore, only the first feature (color) affects the result of clustering while the second feature (shape) does not affect it much. For the second scenario, we would like to obtain a result of clustering as in Figure 1.6(c) where we only focus on the second feature (shape) and do not want an effect of the first feature (color) on the clustering result. A suitable value of $\mathbf{M}$, in this case, is

$$\mathbf{M} = \begin{pmatrix} \varepsilon & 0 \\ 0 & 1 \end{pmatrix}.$$

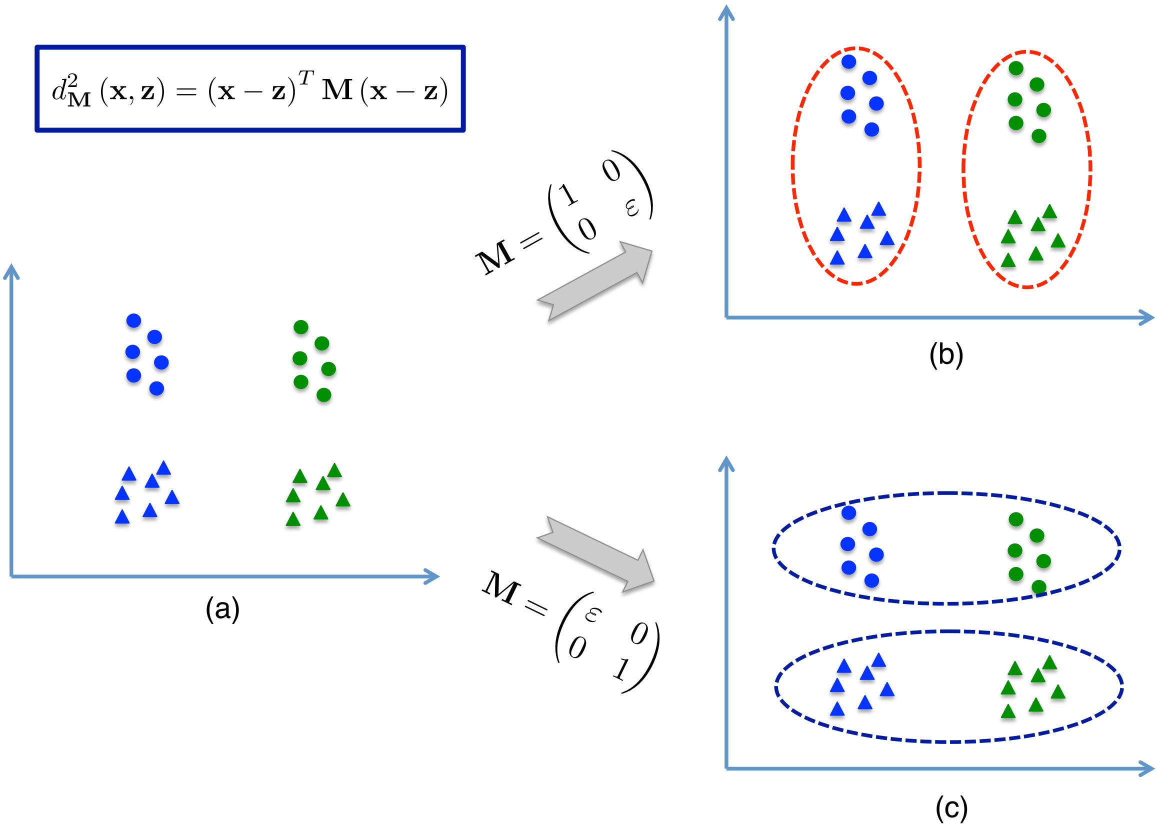

$d^2_{\mathbf{M}}(\mathbf{x},\mathbf{z}) = (\mathbf{x}-\mathbf{z})^T\,\mathbf{M}\,(\mathbf{x}-\mathbf{z})$
$\mathbf{M} = \begin{pmatrix} 1 & 0 \\ 0 & \varepsilon \end{pmatrix}$
$\mathbf{M} = \begin{pmatrix} \varepsilon & 0 \\ 0 & 1 \end{pmatrix}$
(a)
(b)
(c)

Fig. 1.6 An impact of distance on a result of clustering algorithms such as k-means or k-medoids. Consider a dataset as in (a), there are two features for each data point: color (the first dimension) and shape (the second dimension). We would like to cluster the data into two groups following their color feature (b) or shape feature (c) by using the squared Mahalanobis distance $d^2_{\mathbf{M}}$. We can obtain those goals by *learning* a suitable value for the matrix $\mathbf{M}$.

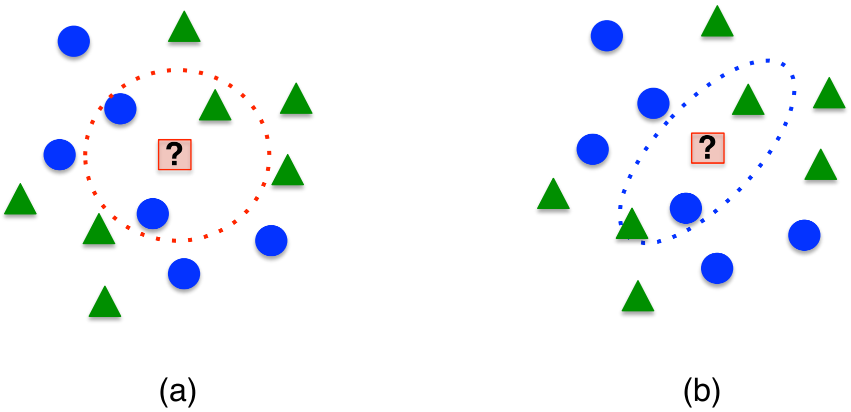


Fig. 1.7 An impact of distance on a result of k-nearest neighbors classification ($k = 3$). Given a dataset and a query data point, a result of the query data point is depend on a metric used in k-nearest neighbors classification. In the case (b), the query data point is assigned by the blue circle while in the case (c), the result is the green triangle.

Furthermore, as in Figure 1.7, we illustrate an impact of distances on a result of k-nearest neighbors classification where we consider $k = 3$. With different metrics to identify nearest neighbors of a query sample, we can obtain very different results. For the metric in Figure 1.7(a), the query sample is assigned as a blue circle label, but as a green triangle one for the metric in Figure 1.7(b).

1.2.2 Metric Learning Approaches

There are two different settings for metric learning: supervised and unsupervised ones.

Supervised Metric Learning

In the supervised setting, a distance will be learned from *labeled* data points. Most of supervised metric learning algorithms are based on a simple and intuitive framework of the Mahalanobis distance. The matrix $\mathbf{M}$ in the Mahalanobis distance (Equation (1.1)) is *learned* from labeled data.

Intuitively, we would like a distance between a pair of samples small if they have a same label and large if they have different labels. Furthermore, we can also consider a triplet of samples. For example, a triplet may be formed as two samples in a same class and the other sample in a different class. Thus, we would like a distance between the two samples in the same class is smaller than a distance between samples in different classes. Figure 1.8 illustrates these basic ideas for constraints on pairs and triplets in supervised metric learning.

There are many proposed algorithms for supervised metric learning (Xing et al., 2002; Schultz and Joachims, 2003; Kwok and Tsang, 2003; Goldberger et al., 2004; Shalev-Shwartz et al., 2004; Globerson and Roweis, 2005; Weinberger et al., 2006; Davis et al., 2007; Weinberger and Saul, 2008, 2009; Kunapuli and Shavlik, 2012; Jain et al., 2012; Trivedi et al., 2014). Particularly, it seems that the Large Margin Nearest Neighbors (Weinberger et al., 2006; Weinberger and Saul, 2008, 2009) and the Information-Theoretic Metric Learning (Davis et al., 2007) approaches have risen as popular tools in this setting.

Unsupervised Metric Learning

In the unsupervised setting for metric learning, a distance is learned from *unlabelled* data points. We can use some basic intuitions about a distance to form criteria. For example, paths go through dense data points should be shorter than relative paths going through sparse data points. A geodesic distance should go through dense data point to capture a geometry of data.

Most of proposed algorithms in metric learning are in the supervised setting. Only a few approaches in metric learning use the unsupervised setting such as (Lebanon, 2002, 2006; Wang et al., 2007).

1.2.3 Limitations and Advantages of Metric Learning

Metric learning aims to learn adaptive distances with respect to a given dataset. Thus, it can better measure similarity between data points to improve performances of algorithms. However, for different tasks with different settings such as metric learning for ranking (McFee and Lanckriet, 2010), multi-task metric learning (Parameswaran and Weinberger, 2010; Yang et al., 2013), domain transfer metric learning (Zhang and Yeung, 2010), multi-modal metric learning (Xie and Xing, 2013) or metric learning for structured data (Norouzi et al., 2012; Kedem et al., 2012; Cuturi and Avis, 2014), a general metric learning may not work well in practice. A suitable learning strategy should be studied for a specific task.

Moreover, metric learning can be interpreted to learn an apt embedding for data points. For example, the Mahalanobis metric learning is equivalent to learn a linear mapping since

$$\begin{aligned} d_{\mathbf{M}}(\mathbf{x},\mathbf{z}) &= \sqrt{(\mathbf{x}-\mathbf{z})^T \mathbf{M} (\mathbf{x}-\mathbf{z})} \\ &= \sqrt{(\mathbf{x}-\mathbf{z})^T \mathbf{L}^T\mathbf{L} (\mathbf{x}-\mathbf{z})} \\ &= \sqrt{(\mathbf{L}\mathbf{x}-\mathbf{L}\mathbf{z})^T (\mathbf{L}\mathbf{x}-\mathbf{L}\mathbf{z})} \\ &= \|(\mathbf{L}\mathbf{x}-\mathbf{L}\mathbf{z})\|_2, \end{aligned}$$

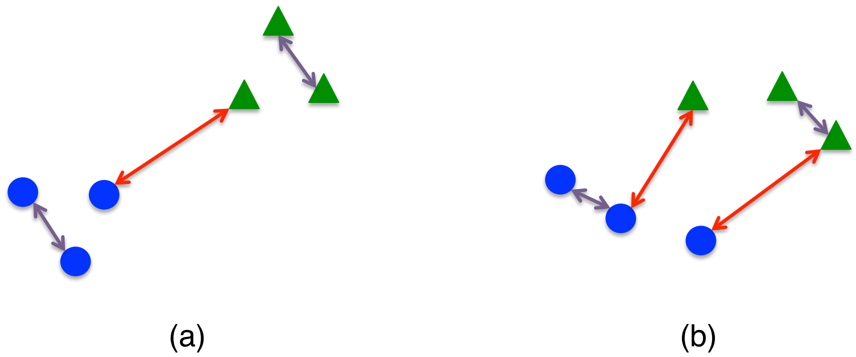


Fig. 1.8 An illustration for constraints on pairs and triplets in metric learning. (a) For a pair of samples, a distance between them should be small if they have a same label and large if they have different labels. (b) Furthermore, a triplet may be formed by two samples having a same label and the other sample has a different label. A constraint on a triplet is that a distance between two samples having the same label should be smaller than a distance between samples having different labels.

where $\mathbf{M} = \mathbf{L}^T\mathbf{L}$ by using the Cholesky decomposition, or singular value decomposition. For nonlinear embeddings, there are some approaches such as kernelized metric learning (Chatpatanasiri et al., 2008; Jain et al., 2012) and local metric learning (Frome et al., 2007; Weinberger and Saul, 2008, 2009; Noh et al., 2010; Wang et al., 2012). Furthermore, deep metric learning is also considered to learn deep embeddings for data points (Chopra et al., 2005; Hu et al., 2014).

However, metric learning still has some limitations such as scalability and robustness. Most of proposed metric learning algorithms can not handle large-scale datasets. There are some attempts to tackle this issue such as online metric learning and batch metric learning (Shalev-Shwartz et al., 2004; Jain et al., 2009). Additionally, Lim et al. (2013) try to deal with robustness limitation of metric learning for noisy data.

1.3 Metric Learning for Histograms

1.3.1 Distances for Histograms

In this subsection, we review some tradition distances for histograms such as the Hellinger distance, the χ^2 distance, the transportation distance (also known as the earth mover's distance) and the Fisher's information metric.

Hellinger distance

The Hellinger distance between two histograms **x** and **z** is

$$d_{\mathrm{H}}(\mathbf{x},\mathbf{z}) = \sqrt{\sum_i \left(\sqrt{\mathbf{x}_{(i)}} - \sqrt{\mathbf{z}_{(i)}}\right)^2}, \tag{1.2}$$

where we denote $\mathbf{x}_{(i)}$ for the i^{th} coordinate of a vector **x**.

Transportation distance

The transportation distance between two histograms **x** and **z** is defined as

$$d_{\mathbf{G}}(\mathbf{x},\mathbf{z}) = \min_{\substack{\mathbf{X1}=\mathbf{x},\mathbf{X}^{\mathbf{T}}\mathbf{1}=\mathbf{z}\\ \mathbf{X}\geq\mathbf{0}}} \langle \mathbf{X},\mathbf{G}\rangle,$$

where **1** is a vector of ones and **G** is a ground metric matrix,

$$\mathbf{G}_{ij} \geq 0, \mathbf{G}_{ii} = 0, \mathbf{G}_{ij} \leq \mathbf{G}_{ik} + \mathbf{G}_{kj}, \forall i,j,k.$$

χ^2 distance

The χ^2 distance between two histograms **x** and **z** is

$$\chi^2(\mathbf{x},\mathbf{z}) = \frac{1}{2}\sum_i \frac{\left(\mathbf{x}_{(i)} - \mathbf{z}_{(i)}\right)^2}{\mathbf{x}_{(i)} + \mathbf{z}_{(i)}}. \tag{1.3}$$

In case, there is an i^{th} coordinate where $\mathbf{x}_{(i)} = \mathbf{z}_{(i)} = 0$, we can eliminate that coordinate for both two histograms before compute the χ^2 distance between them. This is consistent with histograms since **x** and **z** are still histograms after eliminated the i^{th} coordinate if $\mathbf{x}_{(i)} = \mathbf{z}_{(i)} = 0$.

Fisher's information metric

The Fisher's information metric between two histograms **x** and **z** is computed as

$$d(\mathbf{x},\mathbf{z}) = \text{arcos}\left(\sum_i \sqrt{\mathbf{x}_{(i)}\mathbf{z}_{(i)}}\right).$$

The Fisher's information metric is also known as the pull-back metric from the Euclidean metric in a positive sphere through the Hellinger transformation, which is an element-wise square root function for histograms ($\mathbf{x} \mapsto \sqrt{\mathbf{x}}$).

1.3.2 Metric Learning Approaches for Histograms

Most of metric learning algorithms are based on a simple and intuitive framework of the Mahalanobis distances. However, when the input data are histograms, metric learning based on Mahalanobis distance is not the best choice. Empirical evidence shows that Euclidean distance or its general Mahalanobis distance does not work well for histograms in practice since histograms belong to the simplex which is the set of nonnegative and normalized vectors.

Researchers recently try to tackle metric learning for histograms. For instance, Cuturi and Avis propose to learn the ground matrix in the transportation distance framework (2011; 2014), namely ground metric learning (GML). In computer vision, Wang and Guibas (2012) also propose a very similar approach to Cuturi and Avis (2011), which is built upon the GML and the successful Mahalanobis framework large margin nearest neighbor (LMNN). Another approach is the work of Kedem et al. (2012). The authors propose to learn a linear mapping $\mathbf{L}$ for histograms from and onto the simplex ($\mathbf{x} \mapsto \mathbf{L}\mathbf{x}$), where

$$\mathbf{L}\mathbf{1} = \mathbf{1} \text{ and } \mathbf{L} \geq 0,$$

and applies the χ^2 distance for those mapped points $\chi^2\left(\mathbf{L}\mathbf{x}, \mathbf{L}\mathbf{z}\right)$.

There are also a few unsupervised metric learning approaches for histograms. Notably, Lebanon proposes to learn a Riemannian metric on the simplex from *unlabelled* histogram data (2002; 2006). This approach can be interpreted that Lebanon learns a transformation from and onto the simplex for histogram data and then applies the Fisher's information metric for those mapped points. Another work is (Wang et al., 2007). The authors propose to learn a general Dirichlet distribution for histograms to form a ground matrix. Then, they plug it into the transportation distance framework to quantify distance between histograms.

1.4 Kernel Functions

1.4.1 Kernel Functions for Histograms

In this subsection, we review some traditional kernels for histograms such as the histogram intersection kernel, the χ^2 kernel and the Bhattacharyya kernel.

Histogram intersection kernel

The formulation of the histogram intersection kernel between two histograms $\mathbf{x}$ and $\mathbf{z}$ is

$$k_{\text{HIK}}(\mathbf{x},\mathbf{z}) = \sum_i \min\left(\mathbf{x}_{(i)}, \mathbf{z}_{(i)}\right).$$

χ^2 kernel

The χ^2 kernel is computed as

$$k_{\chi^2}(\mathbf{x},\mathbf{z}) = \sum_i \frac{2\mathbf{x}_{(i)}\mathbf{z}_{(i)}}{\mathbf{x}_{(i)} + \mathbf{z}_{(i)}}.$$

If there is an i^{th} coordinate where $\mathbf{x}_{(i)} = \mathbf{z}_{(i)} = 0$, we can eliminate that coordinate for both two histograms before compute the χ^2 kernel between them as in the case of computing the χ^2 distance (Equation (1.3)).

Bhattacharyya kernel

The Bhattacharyya kernel is

$$k_{\text{B}}(\mathbf{x},\mathbf{z}) = \sum_i \sqrt{\mathbf{x}_{(i)}\mathbf{z}_{(i)}},$$

which trivially related to the better-known Hellinger distance (Equation (1.2)) by $d_H(\mathbf{x},\mathbf{z}) = \sqrt{2 - 2k_{\text{B}}(\mathbf{x},\mathbf{z})}$.

1.4.2 Kernel Functions for Images Using the Bag of Features Representation

In this subsection, we review some kernel functions for images using the bag of features (BoF) representation in computer vision such as spatial pyramid matching kernel (Lazebnik et al., 2006) and pyramid match kernel (Grauman and Darrell, 2005).

Spatial Pyramid Matching Kernel

One of the main weak points for the BoF approach is that it discards spatial information of an object of interest. However, the spatial information especially plays an important role for some objects of interest such as images in computer vision. To tackle this issue, Lazebnik et al. (2006) propose the spatial pyramid matching kernel (SPMK) to embed the spatial information for the BoF. Lazebnik et al. (2006) use a sequence of grids to partition an image into fixed sub-regions as illustrated in Figure 1.9, and then compute the bag of visual words

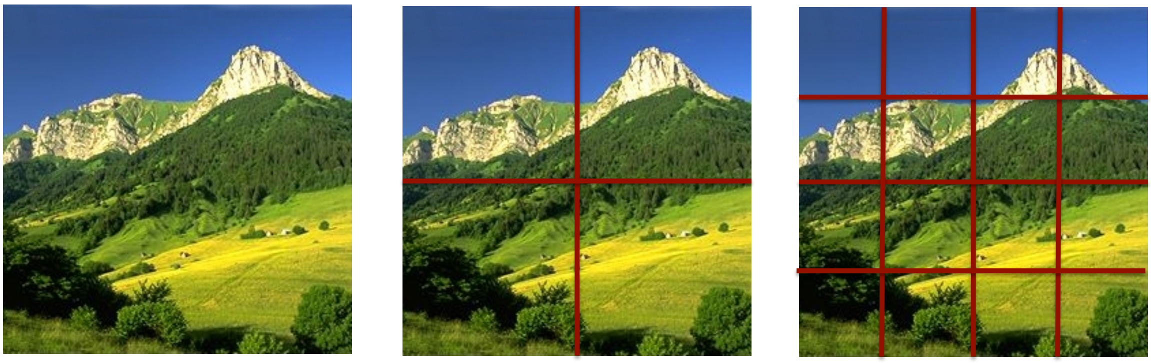


Fig. 1.9 An illustration of a sequence of grids to partition an image into fixed sub-regions in spatial pyramid matching kernel. Spatial pyramid matching kernel use a sequence of grids $2^\ell \times 2^\ell$ where $\ell = 0, 1, 2$ to partition the image into 1, 4 and 16 sub-region respectively.

for each sub-regions. Finally, these histograms for the sub-regions are weighted by some priori constants, and then concatenated to form a representation for the image. Empirical evidence shows that SPMK should be applied with some traditional kernels for histograms to obtain good performances as in the BoF. Furthermore, instead of using some priori constants to weight for the histograms of the sub-regions, Wang and Wang (2010) propose a multiple scale learning framework which employs the multiple kernel learning method to learn the optimal weights for those histograms of the sub-regions.

Pyramid Matching Kernel

Another approach to improve similarity measure between images using the BoF representation approach is the pyramid match kernel proposed by (Grauman and Darrell, 2005). The authors propose to use a sequence of different resolutions for an image as illustrated in Figure 1.10, and then apply the bag of visual words for each image resolution. Finally, these histograms for the image resolutions are also weighted by some prior constants, and concatenated to represent the image. This approach also gives a good performance if applied with some traditional kernels for histograms as well.

1.5 Outline of This Thesis

This thesis is organized as following:

Chapter 2:

In this chapter, we consider metric learning for histograms in a supervised setting. Learning

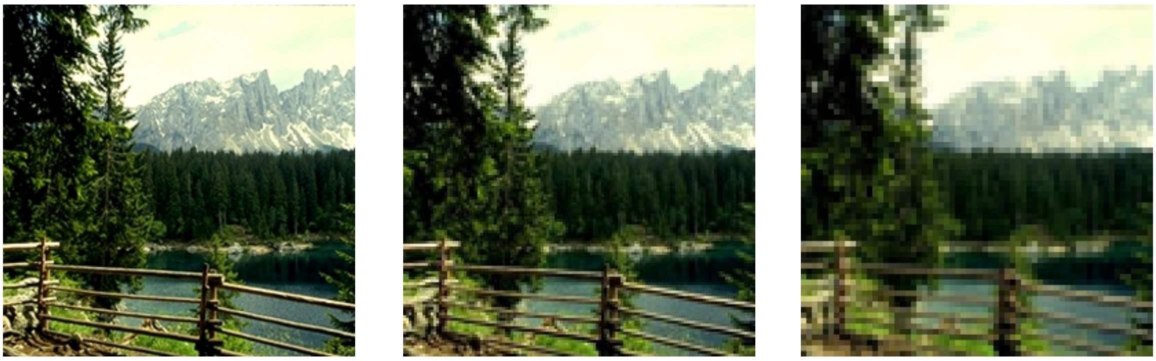


Fig. 1.10 An illustration of a sequence of different resolutions for an image in pyramid match kernel. Pyramid matching kernel considers the image in a sequence of different resolutions: $2^{-r} \times 2^{-r}$ with $r = 0, 1, 2$ scale of the original resolution.

distances for histogram data in the simplex has recently attracted the attention of the machine learning community. Learning such distances is important because the bags of features are popularly used to represent complex objects in many research fields, rather than simple vectors. Ample empirical evidence suggests that the Euclidean distance in general and Mahalanobis metric learning in particular may not be suitable to quantify distances between points in the simplex. We propose in this chapter a new contribution to address this problem by generalizing a family of embeddings proposed by Aitchison (1982) to map the simplex onto a suitable Euclidean space. We provide algorithms to estimate the parameters of such maps by building on previous work on metric learning approaches. The criterion we study is not convex, and we consider alternating optimization schemes as well as accelerated gradient descent approaches. These algorithms lead to representations that outperform alternative approaches to compare histograms in a variety of contexts. The results presented in this chapter are published in [J1, P1].

Chapter 3:

In this chapter, we deal with metric learning for histograms in an unsupervised setting. Many applications in machine learning handle bags of features or histograms rather than simple vectors. In that context, defining a proper geometry to compare histograms can be crucial for many machine learning algorithms. While one might be tried to use a default metric such as the Euclidean metric, empirical evidence shows this may not be the best choice when dealing with observations that lie in the simplex. Moreover, it might be desirable to choose a metric adaptively based on data. We consider in this chapter the problem of learning a Riemannian metric on the simplex given *unlabeled* histogram data. We follow the approach of Lebanon (2006), who proposed to estimate such a metric within a parametric family by maximizing the inverse volume of a given data set of points under that metric. The metrics we consider on the multinomial simplex are pull-back metrics of the Fisher information parameterized by operations within the simplex known as Aitchison (1982) transformations. In this chapter, we propose an algorithmic approach to maximize inverse volumes using sampling and contrastive divergences. We provide experimental evidence that the metric obtained under our proposal outperforms alternative approaches. The results presented in this chapter are published in [P2].

Chapter 4:

In this chapter, we consider to improve similarity measure for images in computer vision which use the bag of features representation. We build upon the spatial pyramid matching proposed by Lazebnik et al. (2006) which is an effective extension for bag of features representation. This method partitions an image into increasingly fine sub-regions and compute histograms of features at each sub-region. Although spatial pyramid matching is the efficient extension of the bag of features image representation, it still measures the similarity between sub-regions by applying the bag of features model. Thus, it is limited in its capacity to quantify similarity between sets of unordered features. To overcome this limitation, we propose in this chapter a hierarchical spatial matching kernel that uses a *coarse to fine* model for the sub-regions to obtain better similarity measure. Our proposed kernel can robustly deal with unordered feature sets as well as a variety of cardinalities. In experiments, the results of hierarchical spatial matching kernel outperformed those of spatial pyramid matching kernel and led to state of the art performance on several well-known benchmark databases in image categorization. The results presented in this chapter are published in [J2].

Chapter 5:

In this chapter, we summary our contributions in this thesis and discuss about some directions for future work.

Chapter 2

Generalized Aitchison Embeddings for Histograms

In this chapter, we propose a new approach for metric learning in a supervised setting when the input data are bags of features.

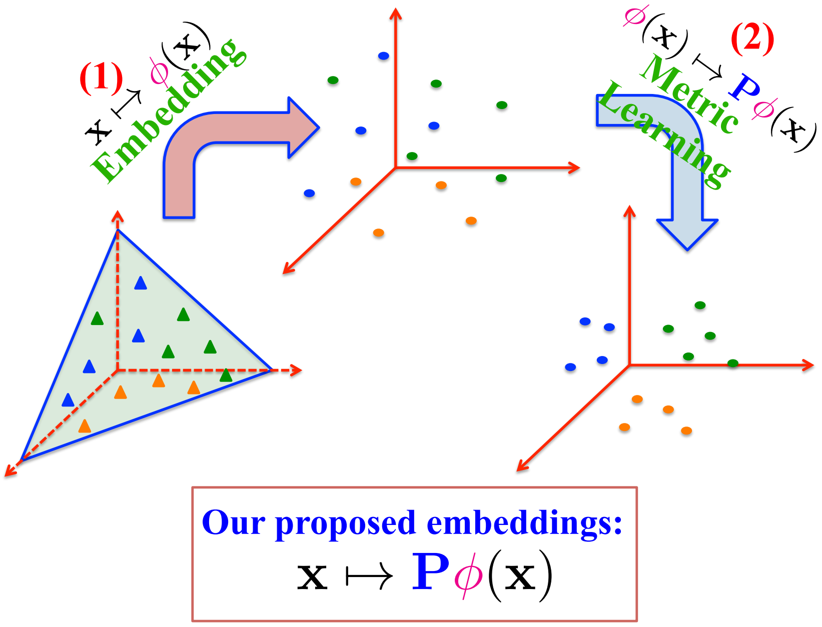


Fig. 2.1 Illustration for our general framework. There are 2 main components: (1) the embedding from the simplex into a suitable Euclidean to take into account geometrical constraints from the simplex, (2) metric learning to improve the performance of classification algorithms based on distance. We unify those 2 components into a compact framework by proposing a family of embeddings.

2.1 Overview

Defining a distance to compare objects of interest is an important problem in machine learning. Many metric learning algorithms were proposed to tackle this problem by considering labeled datasets, most of which exploit the simple and intuitive framework of Mahalanobis distances (Xing et al., 2002; Schultz and Joachims, 2003; Kwok and Tsang, 2003; Goldberger et al., 2004; Shalev-Shwartz et al., 2004; Globerson and Roweis, 2005). Within these contributions, two algorithms are particularly popular in applications: the Large Margin Nearest Neighbor (LMNN) approach described by Weinberger et al. (2006; 2008; 2009), and the Information-Theoretic Metric Learning (ITML) approach proposed by Davis et al. (2007).

Among such objects of interest, histograms – the normalized representation for bags of features – play a fundamental role in many applications, from computer vision (Julesz, 1981; Sivic and Zisserman, 2003; Perronnin et al., 2010; Vedaldi and Zisserman, 2012), natural language processing (Salton and McGill, 1983; Salton, 1989; Baeza-Yates and Ribeiro-Neto, 1999; Joachims, 2002; Blei et al., 2003; Blei and Lafferty, 2006, 2009), speech processing (Doddington, 2001; Campbell et al., 2003; Campbell and Richardson, 2007) to bioinformatics (Erhan et al., 1980; Burge et al., 1992; Leslie et al., 2002). Mahalanobis distances can be used as such on histograms or bags-of-features, but fail however to incorporate the geometrical constraints of the probability simplex (non-negativity – all elements in histograms are nonnegative – and normalization – sum of all elements in histograms is equal to 1) in their definition. Given this issue, Cuturi and Avis (2014) (2011) and Kedem et al. (2012) have very recently proposed to learn the parameters of distances specifically designed for histograms, namely the transportation distance and a generalized variant of the χ^2 distance respectively.

We propose in this chapter a new approach to compare histograms that builds upon older work by Aitchison (1982). In a series of influential papers and monographs, Aitchison (1980; 1982; 1985; 1986; 2003) proposed to study different maps from the probability simplex onto a Euclidean space of suitable dimension. These maps are constructed such that they preserve the geometric characteristics of the probability simplex, yet make subsequent analysis easier by relying only upon Euclidean tools, such as Euclidean distances, quadratic forms and ellipses. Our goal in this chapter is to follow this line of work and propose suitable maps from the probability simplex to a Euclidean space of suitable dimension. However, rather than relying on a few mappings defined a priori such as those proposed in (Aitchison, 1982), we propose to *learn* such maps directly in a supervised fashion using Mahalanobis metric learning.

This chapter is organized as follows: after providing some background on Aitchison embeddings in Section 2.2, we propose a generalization of Aitchison embeddings in Section 2.3. In Section 2.4, we propose algorithms to learn the parameters of such embeddings using training data. We also review related work in Section 2.5, before providing experimental evidence in Section 2.6 that our approach improves upon other adaptive metrics on the probability simplex. Finally, we provide some observations on the empirical behavior of our algorithms in Section 2.7 before giving a summarization for this chapter in Section 2.8.

2.2 Aitchison Embeddings

We consider the probability simplex of n coordinates,

$$\mathbb{P}_{n-1} \overset{\text{def}}{=} \left\{ \mathbf{x} \in \mathbb{R}^n \,\middle|\, \mathbf{x}^T \mathbf{1} = 1 \text{ and } \mathbf{x} \geq 0 \right\},$$

throughout this chapter. Aitchison (1982, 1986, 2003) claims that the information reflected in histograms lies in *the relative values of their coordinates* rather than on their absolute value. Therefore, Aitchison makes the point that comparing histograms directly with Euclidean distances is not appropriate, since the Euclidean distance can only measure the arithmetic difference between coordinates. Given two points $\mathbf{x}$ and $\mathbf{z}$ in the simplex, Aitchison proposes to focus explicitly on the log-ratio of $\mathbf{x}_{(i)}$ and $\mathbf{z}_{(i)}$ for each coordinate i, which can be expressed as the arithmetic difference of the logarithms of $\mathbf{x}_{(i)}$ and $\mathbf{z}_{(i)}$,

$$\log \frac{\mathbf{x}_{(i)}}{\mathbf{z}_{(i)}} = \log \mathbf{x}_{(i)} - \log \mathbf{z}_{(i)}.$$

2.2.1 Additive log-ratio Embedding

The first embedding proposed by Aitchison (1982, p.144, 2003, p.29) is the additive log-ratio map (**alr**) which maps a vector $\mathbf{x}$ from the probability simplex $\mathbb{P}_{n-1}$ onto $\mathbb{R}^{n-1}$,

$$\mathbf{alr}(\mathbf{x}) \overset{\text{def}}{=} \begin{bmatrix} \log \frac{\mathbf{x}_{(1)}+\varepsilon}{\mathbf{x}_{(n)}+\varepsilon} \\ \log \frac{\mathbf{x}_{(2)}+\varepsilon}{\mathbf{x}_{(n)}+\varepsilon} \\ \vdots \\ \log \frac{\mathbf{x}_{(n-1)}+\varepsilon}{\mathbf{x}_{(n)}+\varepsilon} \end{bmatrix} \in \mathbb{R}^{n-1},$$

where $\varepsilon > 0$ is small. The **alr** map for $\mathbf{x} \in \mathbb{P}_{n-1}$ can be reformulated as:

$$\mathbf{alr}(\mathbf{x}) = \mathbf{U} \log \left(\mathbf{x} + \varepsilon \mathbf{1}_n \right), \tag{2.1}$$

where

$$\mathbf{U} = \begin{bmatrix} 1 & \cdots & 0 & -1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & -1 \end{bmatrix} \in \mathbb{R}^{(n-1)\times n},$$

$\mathbf{1}_n \in \mathbb{R}^n$ is the vector of ones, and $\log \mathbf{x}$ is the element-wise logarithm of $\mathbf{x}$. The formula of the **alr** map is related to the definition of the logistic-normal distribution (Aitchison and Shen, 1980; Blei and Lafferty, 2006) on $\mathbb{P}_{n-1}$. The density of a logistic normal distribution at any point in the simplex is proportional to the density of the multivariate normal density on the image of that point under the **alr** map. The **alr** map is an isomorphism between $(\mathbb{P}_{n-1}, \oplus, \odot)$ and $\left(\mathbb{R}^{n-1}, +, \times\right)$ where $\oplus$ and $\odot$ are the perturbation (Aitchison, 2003, p.24) and power (Aitchison, 2003, p.26) operations in the probability simplex respectively, but not isometric since it does not preserve the distance between them.

2.2.2 Centered log-ratio Embedding

The second embedding proposed by Aitchison (2003, p.30) is the centered log-ratio embedding (**clr**), which considers the log-ratio of each coordinate of $\mathbf{x}$ with the geometric mean of all its coordinates,

$$\mathbf{clr}(\mathbf{x}) \stackrel{\text{def}}{=} \begin{bmatrix} \log \frac{\mathbf{x}_{(1)}+\varepsilon}{\sqrt[n]{\prod_{j=1}^{n}\left(\mathbf{x}_{(j)}+\varepsilon\right)}} \\ \log \frac{\mathbf{x}_{(2)}+\varepsilon}{\sqrt[n]{\prod_{j=1}^{n}\left(\mathbf{x}_{(j)}+\varepsilon\right)}} \\ \vdots \\ \log \frac{\mathbf{x}_{(n)}+\varepsilon}{\sqrt[n]{\prod_{j=1}^{n}\left(\mathbf{x}_{(j)}+\varepsilon\right)}} \end{bmatrix} \in \mathbb{R}^n. \tag{2.2}$$

The **clr** map can be also expressed with simpler notations in matrix form:

$$\mathbf{clr}(\mathbf{x}) = \left(\mathbf{I} - \frac{\mathbf{1}_{n\times n}}{n}\right) \log\left(\mathbf{x} + \varepsilon \mathbf{1}_n\right).$$

Here, $\mathbf{I}$ and $\mathbf{1}_{n\times n}$ stand for the identity matrix and the matrix of ones in $\mathbb{R}^{n\times n}$ respectively. The **clr** map is not only an isomorphism, but also an isometry between the probability simplex $\mathbb{P}_{n-1}$ and $\mathbb{R}^n$. Note that the **clr** map spans the orthogonal of $\mathbf{1}_n$ in $\mathbb{R}^n$.

2.2.3 Isometric log-ratio Embedding

Egozcue et al. (2003) proposed to project the images of the **clr** map onto $\mathbb{R}^{n-1}$, to define the *isometric log-ratio embedding* (**ilr**). The **ilr** map is defined as follows:

$$\mathbf{ilr}(\mathbf{x}) \overset{\text{def}}{=} \mathbf{V}\mathbf{clr}(\mathbf{x}) = \mathbf{V}\left(\mathbf{I} - \frac{\mathbf{1}_{n\times n}}{n}\right)\log\left(\mathbf{x} + \varepsilon \mathbf{1}_n\right), \tag{2.3}$$

where $\mathbf{V} \in \mathbb{R}^{(n-1)\times n}$ is a matrix whose row vectors describe a base of the null space of $\mathbf{1}_n^T$ in $\mathbb{R}^n$. The **ilr** map is also an isometric map between both spaces in Aitchison's sense.

Aitchison's original definitions do not consider explicitly the regularization coefficient ε (1982; 1986; 2003). In that literature, the histograms are either assumed to have strictly positive values or the problem is dismissed by stating that all values can be regularized by a very small constant (Aitchison, 1985, p.132; 1986, Section 11.5). We consider explicitly this constant ε here because it forms the basis of the embeddings we propose in the next section.

2.3 Generalized Aitchison Embeddings

Rather than settling for a particular weight matrix – such as those defined in Equations (2.1), (2.2) or (2.3) – and defining a regularization constant ε arbitrarily, we introduce in the definition below a family of mappings that leverage instead these parameters to define a flexible generalization of Aitchison's maps. In the following, S_n^+ is the cone of symmetric positive semidefinite matrices of size $n \times n$.

Definition 1. *Let* $\mathbf{P}$ *be a matrix in* $\mathbb{R}^{r\times n}$ *and* $\mathbf{b}$ *be a vector in the positive orthant* $\mathbb{R}_+^n$. *We define the generalized Aitchison embedding* $\mathfrak{a}(\mathbf{x})$ *of a point* $\mathbf{x}$ *in* $\mathbb{P}_{n-1}$ *parameterized by* $\mathbf{P}$ *and* $\mathbf{b}$ *as*

$$\mathfrak{a}(\mathbf{x}) \overset{\text{def}}{=} \mathbf{P}\log\left(\mathbf{x} + \mathbf{b}\right) \in \mathbb{R}^r. \tag{2.4}$$

Vector $\mathbf{b}$ in Equation (2.4), can be interpreted as a pseudo-count vector that weights the importance of each coordinate of $\mathbf{x}$. Figure 2.2 illustrates how larger pseudo-count values tend to smoothen the logarithm mapping. A large value for $\mathbf{b}_{(i)}$ directly implies that the map for the coordinate described in bin number i is nearly constant, thereby canceling the impact of that coordinate in subsequent analysis. Smaller values for $\mathbf{b}_{(i)}$ denote on the contrary influential coordinates.

We propose to *learn* $\mathbf{P}$ and $\mathbf{b}$ such that histograms mapped following $\mathfrak{a}$ can be efficiently discriminated using the Euclidean distance. The Euclidean distance between the images of

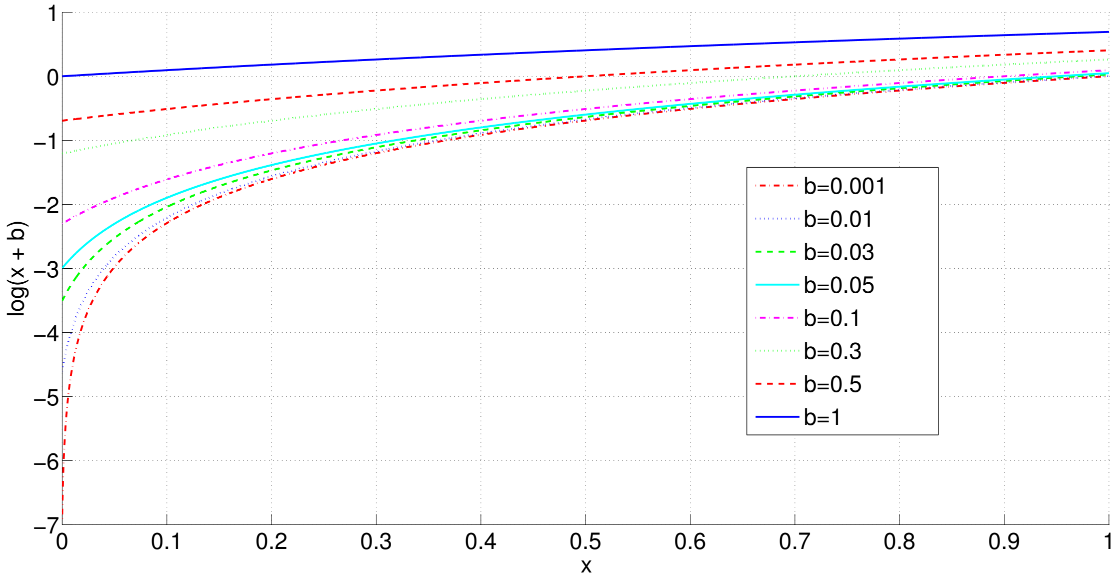


Fig. 2.2 Impact of variable pseudo-count values in the logarithm function.

two histograms $\mathbf{x}$ and $\mathbf{z}$ under the embedding $\mathfrak{a}$ is:

$$\begin{aligned} d_{\mathfrak{a}}(\mathbf{x},\mathbf{z}) &\overset{\text{def}}{=} d(\mathfrak{a}(\mathbf{x}),\mathfrak{a}(\mathbf{z})) \\ &= \|\mathbf{P}\log(\mathbf{x}+\mathbf{b}) - \mathbf{P}\log(\mathbf{z}+\mathbf{b})\|_2 \\ &= \left\|\log\left(\frac{\mathbf{x}+\mathbf{b}}{\mathbf{z}+\mathbf{b}}\right)\right\|_{\mathbf{Q}}, \end{aligned} \tag{2.5}$$

where the division between two vectors is here considered element-wise and we have introduced the positive semidefinite matrix $\mathbf{Q} = \mathbf{P}^T\mathbf{P}$, along with the Mahalanobis norm $\|\cdot\|_{\mathbf{Q}} \overset{\text{def}}{=} \sqrt{\cdot^T Q \cdot}$. Our goal is to learn both $\mathbf{Q} \in \mathrm{S}_d^+$ (we may also consider $\mathbf{P}$ directly) and the pseudo-count vector $\mathbf{b}$ to obtain an embedding that performs well with k-nearest neighbors. We also give an illustration for our general framework in Figure 2.1.

2.4 Learning Generalized Aitchison Embeddings

2.4.1 Criterion

Let $\mathscr{D} = \{(\mathbf{x}_i, y_i)_{1\le i\le m}\}$ be a dataset of labeled points in the simplex, where each $\mathbf{x}_i \in \mathbb{P}_{n-1}$ and each $y_i \in \{1,\cdots,L\}$ is a label. We follow Weinberger et al.'s approach to define a criterion to optimize the parameters $(\mathbf{Q},\mathbf{b})$ (2006; 2009). Weinberger et al. propose a large margin approach to nearest neighbor classification: given a training set $\mathscr{D}$, their criterion considers for a single reference point $\mathbf{x}_i$ the cumulated distance of its closest neighbors that belong to the same class, corrected by a coefficient which takes into account whether points from a different class are in the immediate neighborhood of $\mathbf{x}_i$. Taken together over the entire dataset, these two factors promote metric parameters which ensure that each point's immediate neighborhood is mostly composed of points that share its label.

These ideas can be formulated using the following notations. Let κ be an integer. Given a pair of parameters $(\mathbf{Q}, \mathbf{b})$, consider the geometry induced by $d_{\mathfrak{a}}$. For each point $\mathbf{x}_i$ in the dataset, there exists κ neighbors of $\mathbf{x}_i$ which share its label. We single out these indices by introducing the binary relationship $j \rightsquigarrow i$ for two indices $1 \leq i \neq j \leq m$. The notation $j \rightsquigarrow i$ means that the j-th point is among those close neighbors with the same class (namely $y_i = y_j$). The set of indices j such that $j \rightsquigarrow i$ is called the set of *target neighbors* of the i-th point. Note that $j \rightsquigarrow i$ does not imply $i \rightsquigarrow j$.

Next, we introduce the hinge loss of a real number t as $[t]_+ \overset{\text{def}}{=} \max(t, 0)$, to define the margin between three points: given a triplet (i, j, ℓ) of distinct indices, the margin $\mathtt{H}_{ij\ell}$ is derived as:

$$\mathtt{H}_{ij\ell} \overset{\text{def}}{=} \left[1 + d_{\mathfrak{a}}^2(\mathbf{x}_i, \mathbf{x}_j) - d_{\mathfrak{a}}^2(\mathbf{x}_i, \mathbf{x}_\ell)\right]_+ .$$

This margin is positive whenever the distance between the i-th and ℓ-th points is not larger than the distance between the i-th and j-th points plus an offset of 1. If, for instance, the i-th and j-th points share the same class but the ℓ-th point comes from a different class, $\mathtt{H}_{ij\ell}$ will be positive whenever the ℓ-th point is not far enough from the i-th point relative to where the j-th point stands. Figure 2.3 gives an illustration for the approach of large margin nearest neighbor framework (Weinberger et al., 2006; Weinberger and Saul, 2009).

Using these definitions, we can define the following metric learning problem:

$$\begin{aligned} \min_{\mathbf{Q}, \mathbf{b}} \quad & \mathtt{F} \overset{\text{def}}{=} \sum_{i, j \rightsquigarrow i} d_{\mathfrak{a}}^2(\mathbf{x}_i, \mathbf{x}_j) + \mu \sum_{i, j \rightsquigarrow i} \sum_{\ell} (1 - y_{i\ell}) \mathtt{H}_{ij\ell} + \lambda \|\mathbf{b}\|_2^2 \\ \text{s.t.} \quad & \mathbf{Q} \succeq 0, \\ & \mathbf{b} > \mathbf{0}_n, \end{aligned} \tag{2.6}$$

where $y_{i\ell}$ is equal to 1 if $y_i = y_\ell$ and 0 otherwise, and $\mu > 0, \lambda > 0$ are two regularization parameters. The first term in the objective favors small distances between neighboring points of the same class, while the second term ensures no points with a different label are in the neighborhood of each point, complemented by a regularization term.

2.4.2 Alternating Optimization

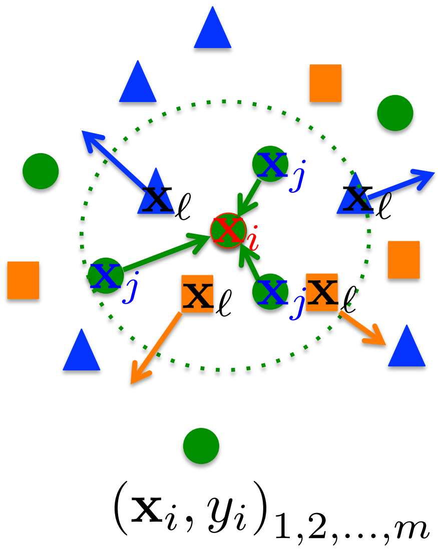


Fig. 2.3 Illustration for the approach of the large margin nearest neighbor framework. Considering a sample $\mathbf{x}_i$, $\mathbf{x}_j$ is target neighbor of the sample $\mathbf{x}_i$. The goal is to push $\mathbf{x}_\ell$, having different labels to $\mathbf{x}_i$, farther and simultaneously pull the target neighbor $\mathbf{x}_j$, having same label as $\mathbf{x}_i$, closer to $\mathbf{x}_i$.

Unlike the original LMNN formulation, optimization problem (2.6) is not convex because of the introduction of a pseudo count vector $\mathbf{b}$. Although the objective is still convex with respect to $\mathbf{Q}$, it is non-convex with respect to $\mathbf{b}$. We consider first a naive approach which updates alternatively $\mathbf{Q}$ and $\mathbf{b}$. This approach is summarized in Algorithm 1 and detailed below.

When $\mathbf{b}$ is fixed, optimization problem (2.6) is equivalent to the Mahalanobis metric learning problem: indeed, once each training vector $\mathbf{x}$ is mapped to $\log(\mathbf{x}+\mathbf{b})$, problem (2.6) can be solved with a LMNN solver.

When $\mathbf{Q}$ is fixed, we can use a projected subgradient descent to learn the pseudo-count vector $\mathbf{b}$. Defining

$$g_{ij}(\mathbf{b}) \overset{\text{def}}{=} d_{\mathfrak{a}}^2(\mathbf{x}_i, \mathbf{x}_j),$$

we can compute the gradient of g_{ij} as:

$$\nabla g_{ij}(\mathbf{b}) = 2\left(\mathbf{Q}\log\frac{\mathbf{x}_i+\mathbf{b}}{\mathbf{x}_j+\mathbf{b}}\right) \bullet \left(\frac{1}{\mathbf{x}_i+\mathbf{b}} - \frac{1}{\mathbf{x}_j+\mathbf{b}}\right),$$

where $\bullet$ is the Schur product – element-wise product – between vectors or matrices. Since only terms such that $\mathtt{H}_{ij\ell}$ is positive contribute to the gradient, a subgradient γ for the objective

Algorithm 1 Alternating optimization (AO) Approach for Problem (2.6)

```
Input: data $(\mathbf{x}_i, y_i)_{1 \leq i \leq m}$, neighborhood size $\kappa$, intialization $\mathbf{b}_0, \mathbf{Q}_0$.
Set $t \leftarrow 0$.
repeat
    Find $\kappa$ target neighbors for each point $\mathbf{x}_i$ with $d_{\mathfrak{a}}$ as in Equation (2.5) at $(\mathbf{Q}_t, \mathbf{b}_t)$.
    Compute $\mathbf{Q}_{t+1}$ using the LMNN algorithm initialized with $\mathbf{Q}_t$ and training data $\{(\log(\mathbf{x}_i + \mathbf{b}_t), y_i)_{1 \leq i \leq m}\}$.
    Update target neighbors for each vector $\mathbf{x}_i$ using parameters $(\mathbf{Q}_{t+1}, \mathbf{b}_t)$.
    Compute $\mathbf{b}_{t+1}$ using Algorithm 2 initialized with $\mathbf{b}_t$, on $\{(\mathbf{x}_i, y_i)_{1 \leq i \leq m}\}$ and $\mathbf{Q}_{t+1}$.
    Update the objective $\mathtt{F}_{t+1} \leftarrow \mathtt{F}(\mathbf{Q}_{t+1}, \mathbf{b}_{t+1})$.
    $t \leftarrow t + 1$.
until $t < t_{\max}$ or insufficient progress for $\mathtt{F}_t$.
Output: matrix $\mathbf{Q}_t$, pseudo-count vector $\mathbf{b}_t$.
```

function $\mathtt{F}$ at $\mathbf{b}_t$ can be expressed as

$$\gamma = \sum_{i, j \rightsquigarrow i} \left(\nabla g_{ij}(\mathbf{b}_t) + \mu \sum_{\ell | \mathtt{H}_{ij\ell} > 0} \left(\nabla g_{ij}(\mathbf{b}_t) - \nabla g_{i\ell}(\mathbf{b}_t) \right) \right) + 2\lambda \mathbf{b}_t. \tag{2.7}$$

This formula results in the following update for $\mathbf{b}_t$ using a preset step size $\frac{t_0}{\sqrt{t}}$:

$$\mathbf{b}_{t+1} = \Psi\left(\mathbf{b}_t - \frac{t_0}{\sqrt{t}} \gamma \right),$$

where $\Psi(\mathbf{x})$ is the projection of $\mathbf{x}$ on the positive orthant offset by a small minimum threshold $\varepsilon = 10^{-20}$, namely the set of all vectors whose coordinates are larger or equal to 10^{-20}. A pseudo-code of this approach is summarized in Algorithm 2. We can set the initial point $\mathbf{Q}_0$ to be equal to $\mathbf{P}^T \mathbf{P}$ where $\mathbf{P}$ can be selected among the linear embeddings originally considered by Aitchison presented in Section 2.2. We initialize the pseudo-count vector to the uniform smoothing term $\mathbf{1}_n / n$.

2.4.3 Projected Subgradient Descent with Nesterov Acceleration

We propose in this section a more straightforward approach to the problem of minimizing Problem (2.6) which bypasses the cost associated with running many iterations of the LMNN solver. We consider a projected subgradient descent using Nesterov acceleration scheme (Nesterov, 1983, 2004) to optimize the parameters $(\mathbf{Q}, \mathbf{b})$ in Problem (2.6) directly. Our experiments show that this approach is considerably faster and equally efficient in terms of classification accuracy.

Algorithm 2 Subgradient Descent Update of **b** when **Q** is fixed.

```
**Input:** data $(\mathbf{x}_i, y_i)_{1\leq i\leq m}$, a matrix $\mathbf{Q}$, a subgradient step size $t_0$, an initial vector $\mathbf{b}_0$.
Set $t \leftarrow 0$.
Set $\mathbf{b}_t \leftarrow \mathbf{b}_0$.
**repeat**
    Compute a subgradient $\gamma$ at $\mathbf{b}_t$ following Equation (2.7).
    Compute $\mathbf{b}_{t+1} \leftarrow \Pi\Big(\mathbf{b}_t - \frac{t_0}{\sqrt{t}}\gamma\Big)$.
    Update the objective $\mathtt{F}_{t+1} \leftarrow \mathtt{F}(\mathbf{Q}, \mathbf{b}_{t+1})$.
    Set $t \leftarrow t+1$.
**until** $t < t_{\max}$ or insufficient progress for $\mathtt{F}_t$.
**Output:** a pseudo-count vector $\mathbf{b}_t$.
```

Analogously to the previous section, we consider the distance $d_{\mathfrak{a}}^2(\mathbf{x}_i, \mathbf{x}_j)$ as a function $h_{ij}(\mathbf{Q}, \mathbf{b})$ of $\mathbf{Q}$ and $\mathbf{b}$. Gradients of h_{ij} with respect to $\mathbf{Q}$ and $\mathbf{b}$ are, introducing the notation $\mathbf{u}_{ij}$ below:

$$\nabla h_{ij}(\mathbf{Q}, \mathbf{b})|_{\mathbf{Q}} = \left(\log \frac{\mathbf{x}_i + \mathbf{b}}{\mathbf{x}_j + \mathbf{b}}\right)\left(\log \frac{\mathbf{x}_i + \mathbf{b}}{\mathbf{x}_j + \mathbf{b}}\right)^T = \mathbf{u}_{ij}\mathbf{u}_{ij}^T,$$

and

$$\nabla h_{ij}(\mathbf{Q}, \mathbf{b})|_{\mathbf{b}} = \nabla g_{ij}(\mathbf{b}).$$

At iteration $t+1$, a subgradient of the objective $\mathtt{F}$ with respect to $\mathbf{b}$ was given in Equation (2.7). We derive similarly a subgradient Γ with respect to $\mathbf{Q}$:

$$\Gamma = \sum_{i,j \rightsquigarrow i} \left(\mathbf{u}_{ij}\mathbf{u}_{ij}^T + \mu \sum_{\ell | \mathtt{H}_{ij\ell} > 0} \left(\mathbf{u}_{ij}\mathbf{u}_{ij}^T - \mathbf{u}_{i\ell}\mathbf{u}_{i\ell}^T\right) \right).$$

Nesterov acceleration scheme builds gradient updates using a momentum that involves two iterations. $\mathbf{b}_t$ and $\mathbf{Q}_t$ can be updated analogously as follows:

$$\begin{aligned} \mathbf{b}_{t-1}^{nes} &= \mathbf{b}_{t-1} + \frac{t-2}{t+1}(\mathbf{b}_{t-1} - \mathbf{b}_{t-2}), \\ \mathbf{b}_t &= \Psi\left(\mathbf{b}_{t-1}^{nes} - \frac{t_0}{\sqrt{t}} \frac{\partial \mathtt{F}}{\partial \mathbf{b}}(\mathbf{Q}_{t-1}, \mathbf{b}_{t-1}^{nes})\right). \end{aligned}$$

and

$$\begin{aligned} \mathbf{Q}_{t-1}^{nes} &= \mathbf{Q}_{t-1} + \frac{t-2}{t+1}(\mathbf{Q}_{t-1} - \mathbf{Q}_{t-2}), \\ \mathbf{Q}_t &= \pi_{\mathtt{S}_d^+}\left(\mathbf{Q}_{t-1}^{nes} - \frac{t_0}{\sqrt{t}} \frac{\partial \mathtt{F}}{\partial \mathbf{Q}}(\mathbf{Q}_{t-1}^{nes}, \mathbf{b}_{t-1})\right). \end{aligned}$$

The projection $\pi_{\mathbb{S}_d^+}$ of a matrix onto the cone of positive semidefinite matrices is carried out by thresholding its negative eigenvalues.

2.4.4 Low-Rank Approaches

Torresani and Lee (2006) have proposed to learn low-rank embeddings for LMNN. We include this variation here, which is beneficial in terms of computational speed, since it only involves storing a low-rank Cholesky factor $\mathbf{P} \in \mathbb{R}^{r \times n}$ of $\mathbf{Q}$ where $r < n$ is a predetermined parameter. This gain comes at the cost of losing convexity when the problem is parameterized by $\mathbf{Q}$. The subgradient of $\mathtt{F}$ with respect to $\mathbf{P}$ is:

$$\frac{\partial \mathtt{F}}{\partial \mathbf{P}} = 2\mathbf{P}\frac{\partial \mathtt{F}}{\partial \mathbf{Q}} \in \mathbb{R}^{r \times n}.$$

When using a descent expressed in terms of $\mathbf{P}$, we obtain the updates

$$\begin{aligned}\mathbf{P}_{t-1}^{nes} &= \mathbf{P}_{t-1} + \frac{t-2}{t+1}\left(\mathbf{P}_{t-1} - \mathbf{P}_{t-2}\right),\\ \mathbf{P}_t &= \mathbf{P}_{t-1}^{nes} - \frac{t_0}{\sqrt{t}}\frac{\partial \mathtt{F}}{\partial \mathbf{P}}\left(\mathbf{P}_{t-1}^{nes}\right).\end{aligned}$$

Since no constraints hold on $\mathbf{P}$, we do not need a projection step.

2.4.5 Adaptive Restart

The projected subgradient descent with Nesterov acceleration presented in Section 2.4.3 does not guarantee a monotone decrease of the objective value. Indeed, it has been observed that Nesterov acceleration scheme may create ripples in the objective value curve when plotted against iteration count. This phenomenon happens when the momentum built from Nesterov acceleration scheme becomes higher than a critical value (the optimal momentum value described by Nesterov (1983; 2004)), and thus damage convergence speed. To overcome this, we adopt the heuristic of O'Donoghue and Candès (2013), which sets the momentum back to zero whenever an increase in the objective is detected. Whenever $\mathtt{F}_t > \mathtt{F}_{t-1}$ at some point in time t, the idea of this heuristic is to erase the memory of previous iterations, reset the algorithm counter to 0 and use the current iteration as a warm start.

2.5 Related Work

Notwithstanding Aitchison's work, the logarithm mapping has been consistently applied in information retrieval to correct for the *burstiness* of feature counts (Salton, 1989; Baeza-Yates and Ribeiro-Neto, 1999; Rennie et al., 2003; Lewis et al., 2004; Madsen et al., 2005), using the mapping

$$\mathbf{x} \mapsto \log(\mathbf{x} + \alpha \mathbf{1}_d), \tag{2.8}$$

for an unnormalized histogram of feature counts $\mathbf{x}$, where $\alpha > 0$ is a constant in $\mathbb{R}_+$ typically set to $\alpha = 1$. This embedding can be directly applied to the original histograms or used on term-frequency inverse-document-frequency (TFIDF) and its variants (Aizawa, 2003; Madsen et al., 2005). These logarithmic maps can be interpreted as particular cases of the embeddings we propose here.

In addition to the logarithm, Hellinger's embedding, which considers the element-wise square-root vector of a histogram ($\mathbf{x} \mapsto \sqrt{\mathbf{x}}$) is particularly popular in computer vision (Perronnin et al., 2010; Vedaldi and Zisserman, 2012). This embedding was also considered as an adequate representation to learn Mahanlanobis metrics in the probability simplex as argued by Cuturi and Avis (2014, Section 6.2.1). Some other explicit feature maps such as χ^2, intersection and Jensen-Shannon are also benchmarked in Vedaldi and Zisserman (2012).

2.6 Experiments

2.6.1 Experimental Setting and Implementation Notes

Datasets

We evaluate our algorithms on 12 benchmark datasets of various sizes. Table 3.1 displays their properties and relevant parameters. These datasets include problems such as scene classification, image classification with a single label or multi labels, handwritten digit and text classification. We follow recommended configurations for these datasets. If they are not provided, we randomly generate 5 folds to evaluate in each run. Additionally, we also repeat the experiments at least 3 times to obtain averaged results, except for PASCAL VOC 2007 and MirFlickr datasets where we use a predefined train and test set.

Parameters of the Proposed Algorithms

We set the target neighborhood size $\kappa = 3$ as a default parameter setting of the `LMNN` solver[1]. We note that the number of target neighbor κ is not necessary to be equal to

[1] http://www.cse.wustl.edu/∼kilian/code/lmnn/lmnn.html

Table 2.1 Properties of datasets and their corresponding experimental parameters.

Dataset	#Train	#Test	#Class	Feature	Rep	#Dim	#Run
MIT Scene	800	800	8	SIFT	BoF	800	5
UIUC Scene	1500	1500	15	SIFT	BoF	800	5
DSLR	409	89	31	SURF	BoF	800	5
WEBCAM	646	149	31	SURF	BoF	800	5
AMAZON	2262	551	31	SURF	BoF	800	5
OXFORD Flower	680	680	17	SIFT	BoF	400	5
CALTECH-101	3060	2995	102	SIFT	BoF	400	3
Pascal Voc 2007	5011	4952	20	Dense Hue	BoF	100	1
MirFlickr	12500	12500	38	Dense Hue	BoF	100	1
MNIST	5000	5000	10	Normalized Intensity		784	5
20 News Group	600	19397	20	BoW	LDA	200	5
Reuters	500	9926	10	BoW	LDA	200	5

parameter k in k-nearest neighbor classification. In our experiments, κ is a fixed number while k varies. We also set the regularization $\mu = 1$ as in LMNN (Weinberger and Saul, 2009) while the regularization λ is set to κN (recall that N is the size of the training set), guided by preliminary experiments. For the step size t_0 in the subgradient descent update, we choose from the set $\frac{1}{\kappa N}\{0.01, 0.05, 0.1, 0.5\}$ via cross validation. For the alternating optimization (Algorithm 1), we set $t_{\max} = 20$ iterations (in our experiments, we observe that this number is generous, since usually 6 to 10 iterations suffice for most datasets, as shown in Figure 2.8 and Figure 2.9). For the projected subgradient descent with Nesterov acceleration (PSGD-NES), the algorithm takes less than 500 iterations for converge (usually about 300 iterations, illustrated in Figure 2.12 and Figure 2.13). So, we set $t_{\max} = 500$ for the PSGD-NES algorithm.

Dense SIFT Features for Images

Dense SIFT features are computed by operating a SIFT descriptor of 16×16 patches computed over each pixel of an image as in (Le et al., 2011) instead of key points (Lowe, 2004) or a grid of points (Lazebnik et al., 2006). Additionally, before computing the dense SIFT, we convert images into gray scale ones to improve robustness. We obtained dense SIFT features by using the LabelMe toolbox[2] (Russell et al., 2008).

[2] http://new-labelme.csail.mit.edu/Release3.0/browserTools/php/matlab_toolbox.php

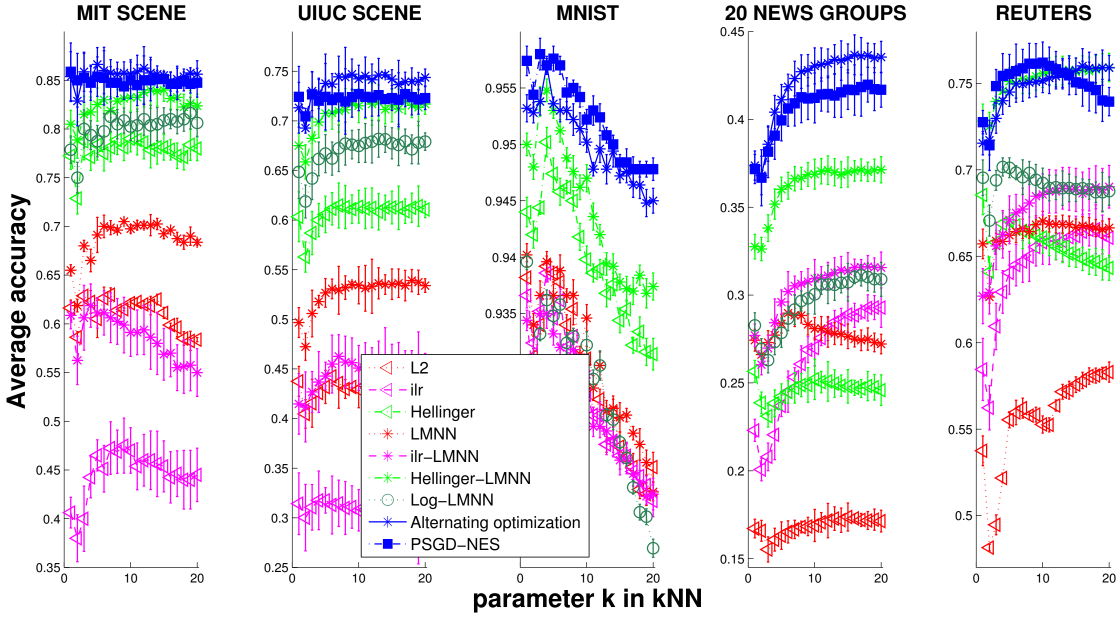


Fig. 2.4 Classification on scene (MIT Scene & UIUC Scene), handwritten digit (MNIST) and text (20 News Group & Reuters).

2.6.2 Metrics and Metric Learning Methods

We consider LMNN metric learning for histograms using: their original representation; the **ilr** representation (Section 2.2, Equation (2.3)); their Hellinger map. We also include the simple Euclidean distance in our benchmarks. To illustrate the fact that learning the pseudo-count vector **b** results in significant performance improvements, we also conduct experiments with an algorithm that learns **Q** through LMNN but only considers a uniform pseudo-count vector of α chosen in $\{0.0001, 0.001, 0.01, 0.1, 1\}$ by cross validation on the training fold. We call this approach Log−LMNN.

2.6.3 Scene Classification

We conduct experiments on the MIT Scene[3] and UIUC Scene[4] datasets. In these datasets, we select randomly 100 train and 100 test points from each class. Histograms are obtained by using dense SIFT features with bag-of-feature representation (BoF) where the number of

[3] http://people.csail.mit.edu/torralba/code/spatialenve-lope/

[4] http://www.cs.illinois.edu/homes/slazebni/research/

Table 2.2 Averaged percentage of zero-elements in a histogram (sparseness) of single-label datasets.

Dataset	Sparseness
MIT Scene	20.04%
UIUC Scene	20.33%
DSLR	39.58%
WEBCAM	64.44%
AMAZON	83.20%
OXFORD Flower	1.12%
CALTECH-101	13.15%
MNIST	80.68%
20 News Group	98.01%
Reuters	98.00%

visual words is set to 800. We repeat experiments 5 times on each dataset and split randomly onto train and test sets.

The two leftmost graphs in Figure 2.4 shows averaged results with error bars on these datasets. The performance of the proposed embedding improves upon that of `LMNN` on the original histograms by more than 15% and is slightly better than `LMNN` combined with the Hellinger map. These graphs also illustrates that Hellinger is an efficient embedding for histograms. The performance of *k*-NN seeded with the Hellinger distance is even better than that of `LMNN` in these datasets. The performances of all alternative embeddings with `LMNN` are better than those with Euclidean distance respectively.

2.6.4 Handwritten Digits Classification

We also perform experiments for handwritten digits classification on the MNIST[5] dataset. A feature vector for each point is constructed from a normalized intensity level of each pixel. We randomly choose 500 points disjointly from each class for train and test sets, repeat 5 times for averaged results. The middle graph in Figure 2.4 illustrates that the generalized Aitchison embedding also outperforms other alternative embeddings.

[5] http://yann.lecun.com/exdb/mnist/

2.6.5 Text Classification

We also carry out experiments for text classification on 20 News Groups[6] and Reuters[7] (the 10 largest classes) datasets. In these datasets, we calculate bag of words (BoW) for each document, and then we use topic modelling (LDA) to reduce the dimension of histograms using the *gensim* toolbox[8]. We obtain a histogram of topics for each document (Blei et al., 2003; Blei and Lafferty, 2009). We randomly choose 30 points and 50 points from each class in 20 News Groups and Reuters datasets for training, and use the remaining points for testing respectively. We randomly generate 5 different train and test sets for each dataset and average results.

The two rightmost graphs in Figure 2.4 show that the proposed embedding improves the performance of `LMNN` by more than 10% on each dataset. It also outperforms the **ilr** and Hellinger representations on these datasets, except for the Reuters dataset where their performances are comparable. Moreover, as in Table 2.2 which illustrates averaged percentages of zero-elements in a histogram (sparseness), these datasets are very sparse. There are averaged more than 98% zero-elements in a histogram in these datasets. Therefore, the proposed algorithm may have advantages for very sparse datasets.

2.6.6 Single-label Object Classification

DSLR, AMAZON and WEBCAM

These datasets[9] are split into 5 folds. Each point is a histogram of visual words obtained by BoF representation on SURF features (Bay et al., 2006) where the code-book size is set to 800. We repeat experiments 5 times on each dataset with different random splits and average results.

The three leftmost graphs in Figure 2.5 illustrate that the performance of the proposed embedding outperforms that of `LMNN` on these datasets and even improves about 30%, 25% and 10% on DSLR, WEBCAM and AMAZON dataset respectively. Our proposed algorithm also improves the performances of `Log-LMNN` about 7%.

OXFORD FLOWER

We randomly choose 40 flower images of each class for training and use the rest for testing

[6] http://qwone.com/∼jason/20Newsgroups/

[7] http://archive.ics.uci.edu/ml/datasets/Reuters–21578+Text+Categorization+Collection

[8] http://radimrehurek.com/gensim/

[9] http://www1.icsi.berkeley.edu/∼saenko/projects.html

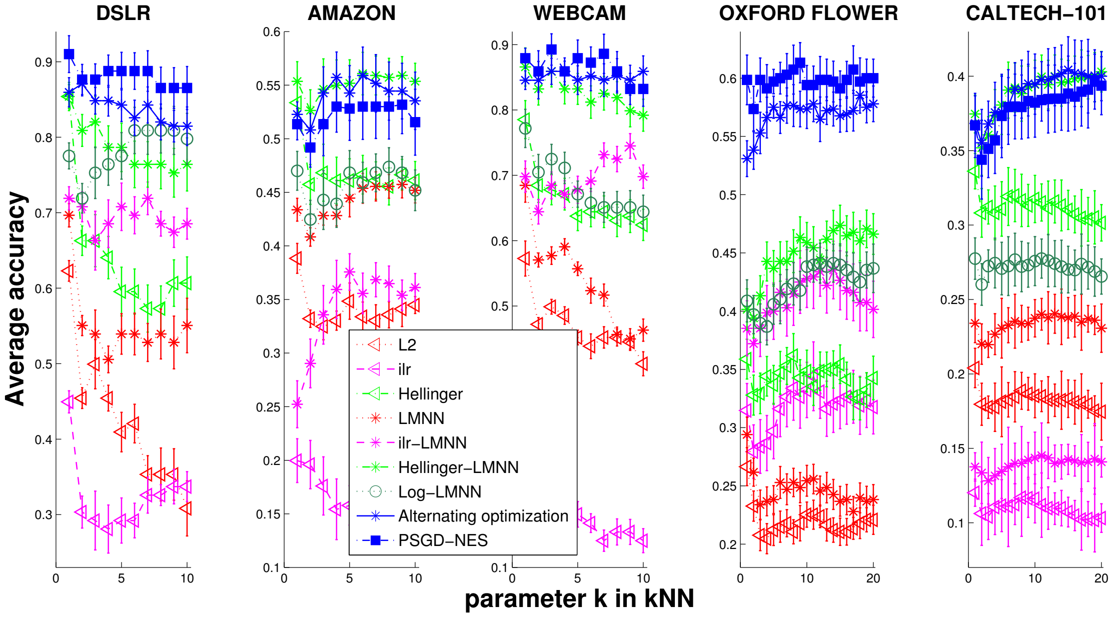


Fig. 2.5 Single-label object classification on DSLR, AMAZON, WEBCAM, OXFORD FLOWER and CALTECH-101.

for Oxford Flower dataset[10]. We construct histograms using a BoF representation with 400 visual words on a dense SIFT feature and repeat experiments 5 times on different random splits to obtain averaged results. The fourth graph in Figure 2.5 shows that the proposed embedding outperforms that of histograms more than 30%, and also improves about 15% comparing to the **ilr** embedding as well as the Hellinger representation with `LMNN`. As showed in Table 2.2, this dataset is highly dense since there are only about 1% zero-elements in a histogram. This suggests that our approaches might work better with dense datasets.

CALTECH-101

We randomly choose 30 images for training and up to 50 other images for testing. We use BoF representation with 400 visual words on a dense SIFT feature to construct histograms for each image. The rightmost graph in Figure 2.5 shows averaged results on 3 different random splits of the CALTECH-101[11] dataset, illustrating again the interest of our approach.

[10] http://www.robots.ox.ac.uk/∼vgg/data/flowers/17/

[11] http://www.vision.caltech.edu/Image_Datasets/Cal-tech101/

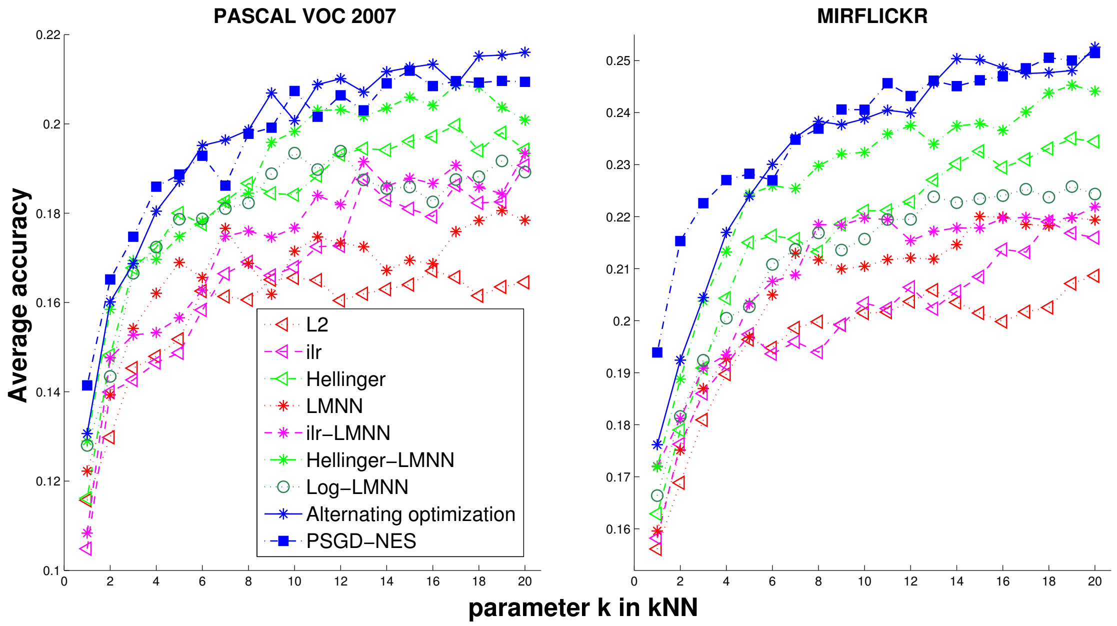


Fig. 2.6 Multi-label object classification on PASCAL VOC 2007 & MirFlickr.

2.6.7 Multi-label Object Classification

We evaluate the proposed method on multi-label image categorization using the PASCAL VOC 2007[12] and MirFlickr[13] datasets. We follow the guidelines to define the train and test sets. Histograms for each image are built in these datasets based on BoF representation with 100 visual words on a dense hue feature. Then, we employ a one-versus-all strategy for k-NN classification and calculate averaged precisions for each dataset. Figure 2.6 illustrates that the proposed embedding outperforms original, **ilr**, and Hellinger representation with `LMNN` again. Additionally, the performance of Hellinger distance is better than that of `LMNN` and comparative with that of `Log-LMNN` in these datasets.

2.6.8 Low-Rank Embeddings

We conduct experiments for the low-rank version of our algorithm, where the dimension is set to $\{80\%, 60\%, 40\%, 20\%\}$ of the original dimension of the single-label datasets. Figure 2.7 indicates that reducing rank can be carried out to accelerate computations, but this speed up can come, depending on the dataset, at the expense of a degradation in performance.

[12] http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2007/
[13] http://press.liacs.nl/mirflickr/

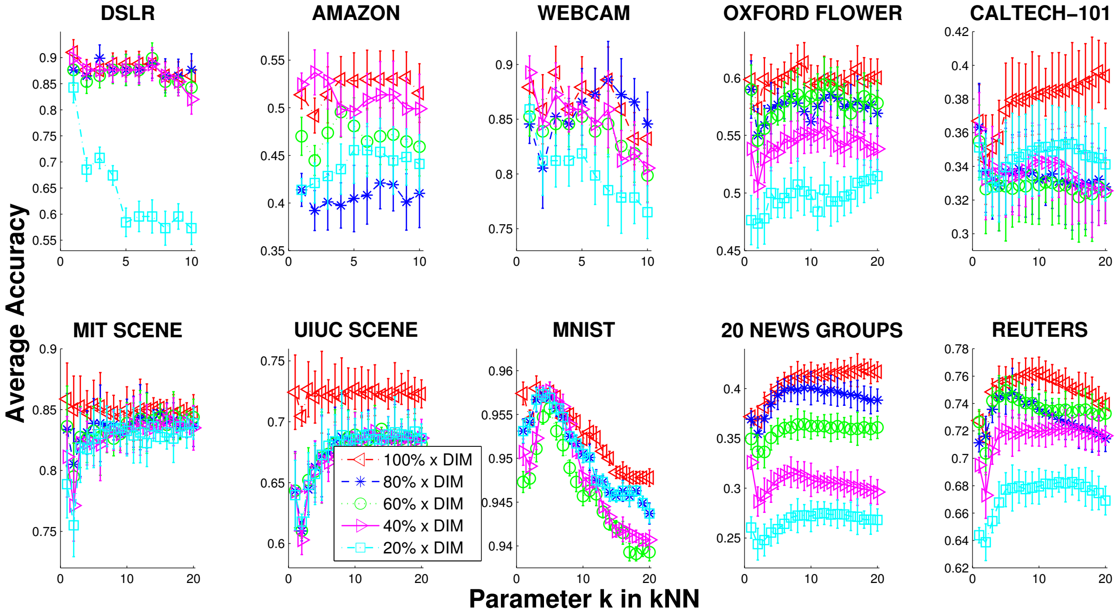


Fig. 2.7 Classification results for low-rank generalized Aitchison embedding.

2.7 Experimental Behavior of the Algorithm

2.7.1 Convergence Speed

Figure 2.8 and Figure 2.9 illustrate the convergence of the objective with respect to computational time on a log-log scale. We consider the naive alternating optimization approach (Section 2.4.2), a standard projected subgradient descent, projected subgradient descent with Nesterov acceleration (Section 2.4.3), and a version with adaptive restart (PSGD-NES-AR in Section 2.4.5). We use the LMNN solver directly and measure raw time using a single core. Gaps in that curve indicate the value of the objective before and after running the LMNN solver.

The naive alternating optimization has computational cost that is about one order of magnitude larger than that of a direct application of LMNN. This factor appears because we run the LMNN solver multiple times. The burden of optimizing the pseudo-count vector is small due to the fact that the gradient has a closed-form solution for each pair in the objective function. We only need to run a few iterations of the LMNN algorithm using a warm start when alternating. Our experiments show that we only need to run 6 to 10 alternating iterations for these datasets, but each iteration is costly. These results show the interest of

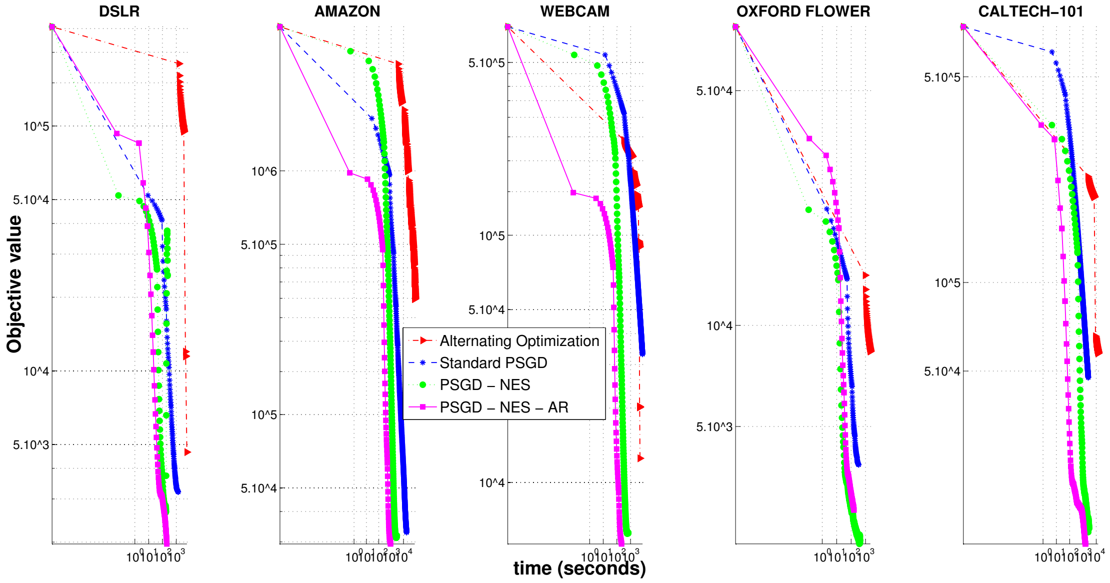


Fig. 2.8 Log-log plot illustration for the relation between behavior of the objective function and computational time in the proposed algorithms on DSLR, AMAZON, WEBCAM, OXFORD FLOWER and CALTECH-101.

using Nesterov acceleration scheme here, and even suggest adopting the adaptive restart heuristic of O'Donoghue and Candès (2013).

2.7.2 Sensitivity to Parameters

Target Neighbors

Figure 2.10 and Figure 2.11 illustrate the effect of the number of target neighbors κ on the results of our algorithms. We evaluate for $\kappa = \{1,3,5,7,9\}$ for single-label datasets, except $\kappa = \{7,9\}$ for DSLR and $\kappa = 9$ for WEBCAM due to the size of the smallest class in these datasets. These results suggest that the number of target neighbors has a large impact and should remain low, both from a computational viewpoint and performances of the algorithm. Figure 2.10 and Figure 2.11 also show that 3-target-neighbor setup is an appropriate choice for those evaluated datasets.

Average Test Accuracy over Iteration Count

Figure 2.12 and Figure 2.13 show the average test accuracy over iteration count. The

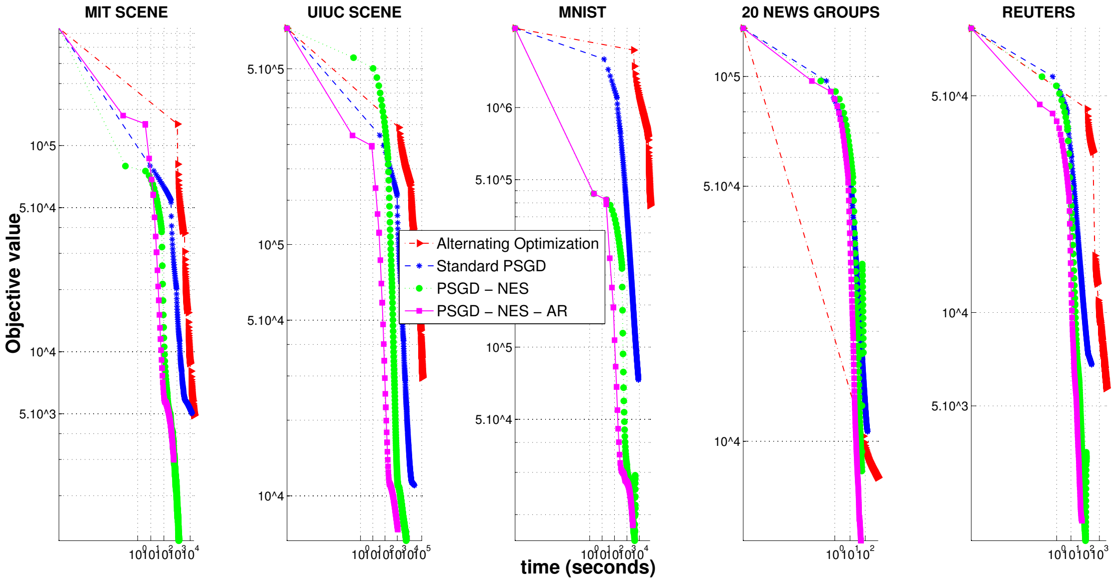


Fig. 2.9 Log-log plot illustration for the relation between behavior of the objective function and computational time in the proposed algorithms on scene (MIT Scene & UIUC Scene), handwritten digit (MNIST) and text (20 News Group & Reuters).

curves of average test accuracy value seem to increase monotonically with the iteration count, therefore suggesting that our algorithms do not overfit training data in these evaluations.

2.8 Summary

We have shown that a generalized family of embeddings for histograms coupled with different procedures to estimate its parameters can be effective to represent histograms in Euclidean spaces. Our variations outperform other common approaches such as the Hellinger map or Aitchison's original embeddings. Rather than using an alternative optimization scheme and use `LMNN` solvers, our results indicated that a simple accelerated subgradient method provides the best results both in performance and computational time. Other variations, such as learning a low-rank embedding or using adaptive restart heuristic for PSGD-NES, can also prove beneficial, depending on the datasets.

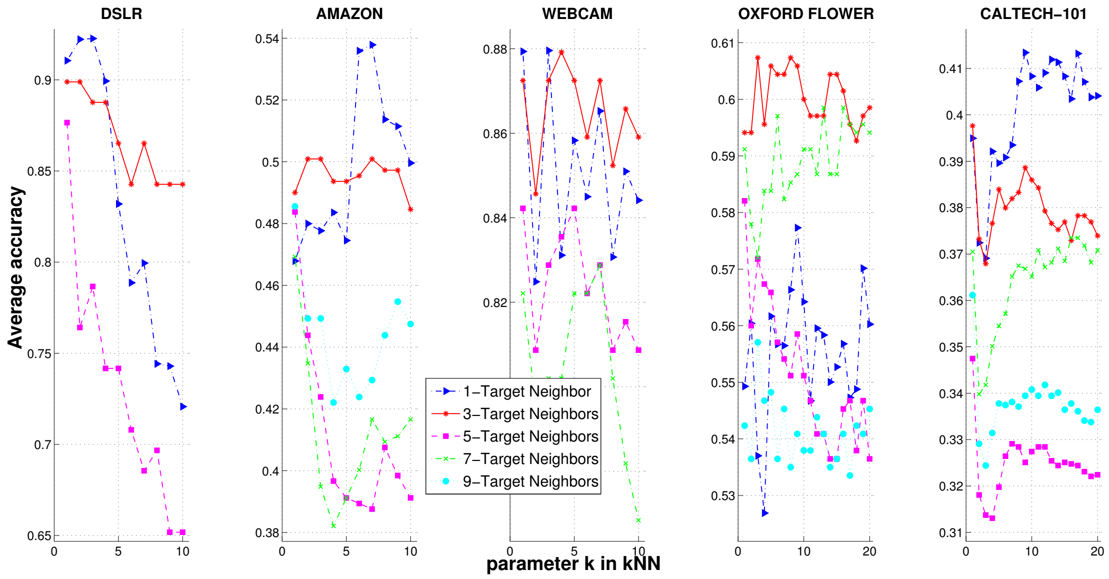


Fig. 2.10 Illustration for the effect of target neighbors in the PSGD-NES on DSLR, AMAZON, WEBCAM, OXFORD FLOWER and CALTECH-101.

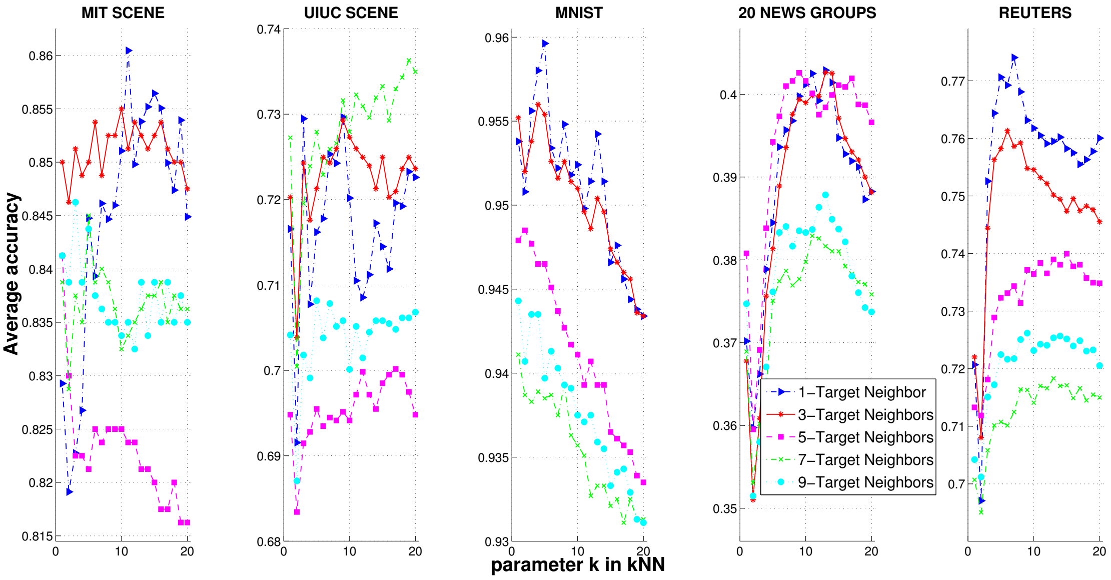


Fig. 2.11 Illustration for the effect of target neighbors in the PSGD-NES scene (MIT Scene & UIUC Scene), handwritten digit (MNIST) and text (20 News Group & Reuters).

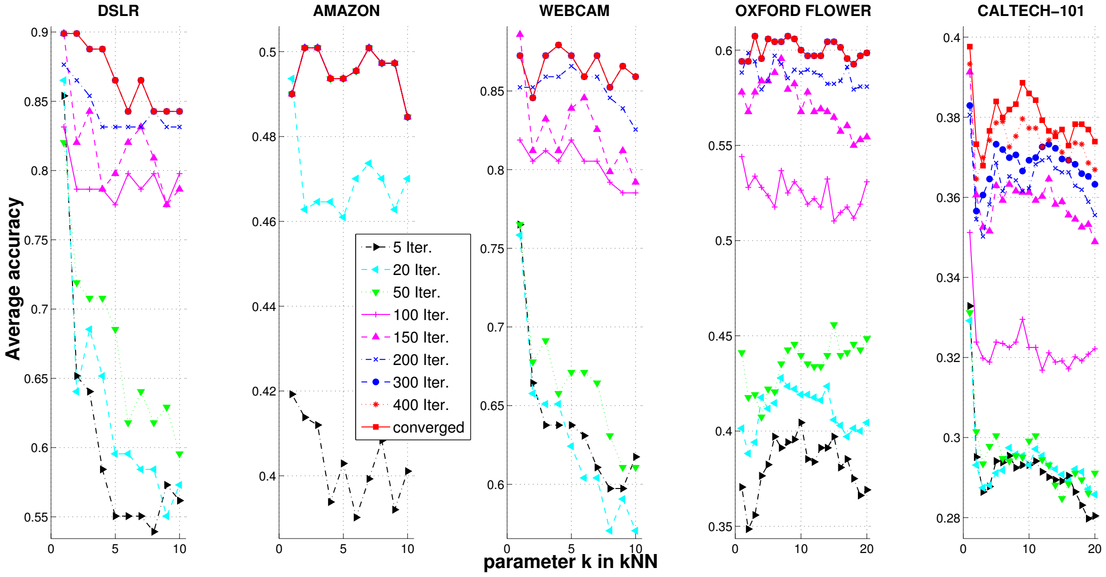


Fig. 2.12 Illustration for the average test accuracy over iteration count of the PSGD-NES to indicate that the algorithm do not overfit to training data on DSLR, AMAZON, WEBCAM, OXFORD FLOWER and CALTECH-101.

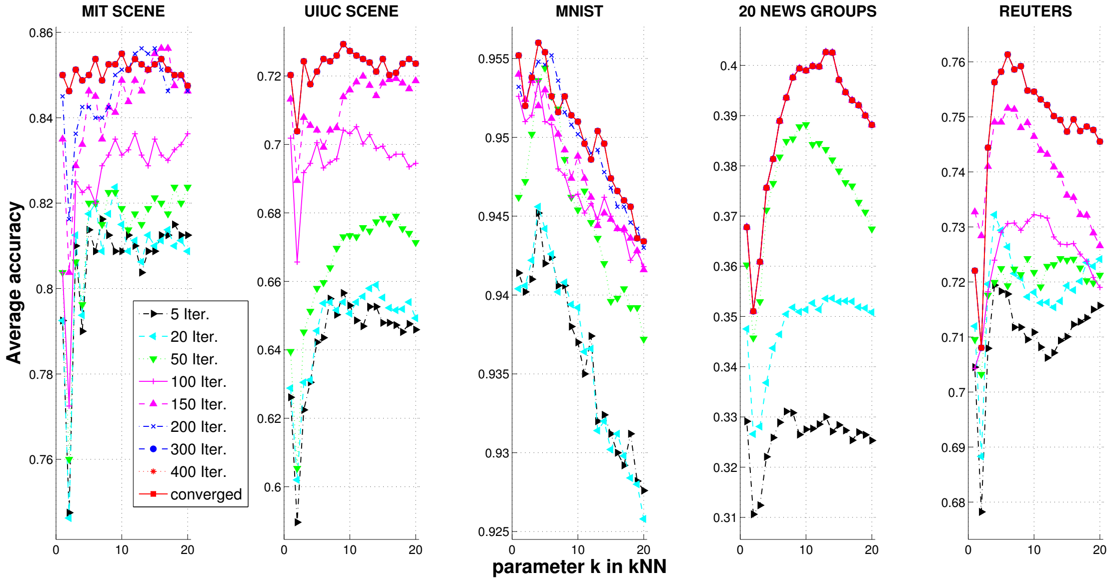


Fig. 2.13 Illustration for the average test accuracy over iteration count of the PSGD-NES to indicate that the algorithm do not overfit to training data scene (MIT Scene & UIUC Scene), handwritten digit (MNIST) and text (20 News Group & Reuters).

Chapter 3

Unsupervised Riemannian Metric Learning for Histograms

In this chapter, we propose a new approach to learn a metric for histograms from *unlabeled* data.

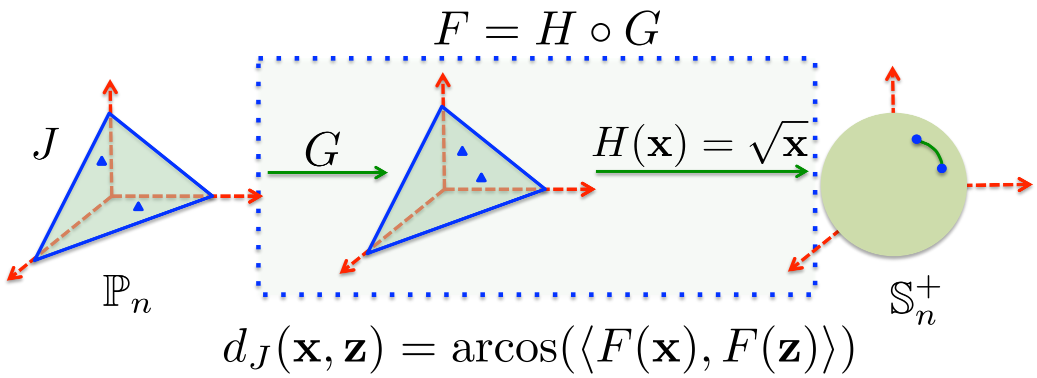

$F = H \circ G$
J
G
$H(\mathbf{x}) = \sqrt{\mathbf{x}}$
$\mathbb{P}_n$
$\mathbb{S}_n^+$
$d_J(\mathbf{x},\mathbf{z}) = \text{arcos}(\langle F(\mathbf{x}), F(\mathbf{z})\rangle)$

Fig. 3.1 Illustration for the Riemannian metric in the simplex considered in this chapter – the Fisher information metric J under the transformation G on the simplex $\mathbb{P}_n$. It is also known as a pull-back metric of the Euclidean metric on the positive sphere $\mathbb{S}_n^+$ through a transformation $F = H \circ G$.

3.1 Overview

Learning distances to compare objects is an important topic in machine learning. Many approaches have been proposed to tackle this problem, notably by making the most of Mahalanobis distances in a supervised setting (Xing et al., 2002; Schultz and Joachims, 2003; Goldberger et al., 2004; Shalev-Shwartz et al., 2004; Globerson and Roweis, 2005; Weinberger et al., 2006; Davis et al., 2007; Weinberger and Saul, 2008, 2009).

Among such objects of interest, histograms – the normalized representation for bags of features – are popular in many applications, notably computer vision (Julesz, 1981; Sivic and Zisserman, 2003; Vedaldi and Zisserman, 2012), natural language processing (Salton and McGill, 1983; Salton, 1989; Joachims, 2002; Blei et al., 2003; Blei and Lafferty, 2009) and speech processing (Doddington, 2001; Campbell and Richardson, 2007). Mahalanobis distances can be used as such on histograms, but are known to perform poorly because they do not take into account the inherent constraints that histograms have (non-negativity and normalization). Cuturi and Avis (2014) and Kedem et al. (2012) proposed recently two supervised metric learning approaches in the simplex. Kedem et al.'s contribution is particularly relevant to this chapter's approach: they proposed to compare two histograms $\mathbf{r}$ and $\mathbf{c}$ by using the χ^2 distance, $\chi^2(\mathbf{Lr},\mathbf{Lc})$ between $\mathbf{Lr}$ and $\mathbf{Lc}$, where $\mathbf{L}$ is a linear map from and onto the simplex. This map $\mathbf{L}$ is learned by using labeled data and the large margin nearest neighbor framework (Weinberger et al., 2006; Weinberger and Saul, 2008, 2009). Our approaches also build on the idea of learning a map from and onto the simplex to parameterize a family of distances.

An even stronger influence on this chapter lies in the work of Lebanon (2002, 2006) who proposed to learn a Riemannian metric for histograms using unlabeled data. The family of Riemannian metrics considered in these works can be seen as the standard Fisher information metric (instead of the χ_2 distance) using a particular family of transformations in the simplex. Cuturi and Avis (2014, Section 5.3) noticed that these transformations were defined in earlier references by Aitchison (1982, 1986, 2003) who called them simplicial perturbations.

Our contribution in this chapter is two-fold: (1) we extend Lebanon (2006)'s original approach to more general Aitchison transformations in the simplex; (2) we propose a new approach to solve a key step in Lebanon's procedure, namely the maximization of the inverse volume of a Riemannian metric.

This chapter is organized as follows: after providing short reminders of Aitchison's tools and Riemannian geometry in Section 3.2, we proceed with the description of Fisher's information metric for histograms and show how all these elements can be used to form a parameterized family of Riemannian metrics in the simplex in Section 3.3. In Section 3.4, we propose a new algorithm to learn such metrics in an unsupervised way. In Section 3.5, we propose to use locally sensitive hashing to approximate k-nearest neighbors for our metrics to apply for large datasets. We study connections of this work with related approaches in Section 3.6, before providing experimental evidence in Section 3.7, and giving a summarization of this chapter in Section 3.8.

3.2 Preliminary

We provide in this section a self-contained review of Aitchison's geometry as well as elements of Riemannian geometry that will be useful to define our methods.

3.2.1 Aitchison Geometry

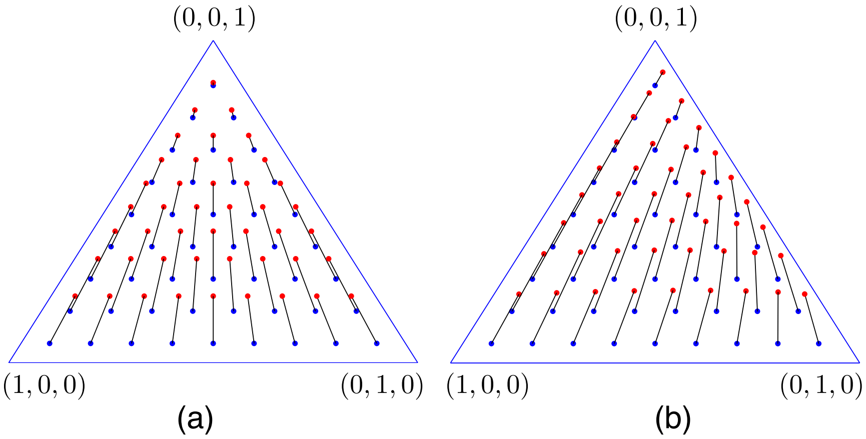


Fig. 3.2 Illustration for the perturbation operator in the simplex $\mathbb{P}_2$ where blue dots are input histograms and red dots are images of the perturbation operator on blue dots. (a) for $\lambda = [0.3, 0.3, 0.4]$ and (b) for $\lambda = [0.28, 0.34, 0.38]$.

We consider the n-simplex $\mathbb{P}_n$, defined by

$$\mathbb{P}_n \overset{\text{def}}{=} \left\{ \mathbf{x} \in \mathbb{R}^{n+1} \;\middle|\; \mathbf{x} \geq 0 \text{ and } \mathbf{x}^T \mathbf{1} = 1 \right\},$$

where $\mathbf{1}$ is a vector of 1 and write $\text{int}\mathbb{P}_n$ for its interior. Aitchison (1982, 1986, 2003) claims that the information reflected in histograms lies in the relative values of their coordinates rather than on their absolute value. Therefore, Aitchison proposes dedicated binary operations to combine two elements $\mathbf{x}$ and $\mathbf{z}$ in the interior of the simplex. Given $\gamma \in \mathbb{R}$, the perturbation and powering operations, denoted by $\oplus$ and $\odot$, are respectively defined as

$$\mathbf{x} \oplus \mathbf{z} \overset{\text{def}}{=} C(\mathbf{x} \bullet \mathbf{z}) \in \text{int}\mathbb{P}_n,$$

and

$$\gamma \odot \mathbf{z} \overset{\text{def}}{=} C(\mathbf{z}^{\gamma}) \in \text{int}\mathbb{P}_n,$$

where we denote • for element-wise product – Hadamard product – between 2 vectors and $C(\mathbf{x}) = \frac{\mathbf{x}}{\mathbf{x}^T\mathbf{1}}$ is the closure or normalization operator. Figure 3.2 and Figure 3.3 illustrate the action of the perturbation and powering operators respectively for histograms in the simplex $\mathbb{P}_2$. Additionally, a definition for the difference between $\mathbf{x}$ and $\mathbf{z}$ is naturally defined as:

$$\mathbf{x} \ominus \mathbf{z} = \mathbf{x} \oplus (-1 \odot \mathbf{z}) = C\left(\frac{\mathbf{x}}{\mathbf{z}}\right) \in \text{int}\mathbb{P}_n,$$

where $\frac{\mathbf{x}}{\mathbf{z}}$ is element-wise division for 2 vectors $\mathbf{x}$ and $\mathbf{z}$. Note that the difference of two elements in the simplex with these operations remains in the simplex, unlike the results obtained in with the usual Euclidean geometry.

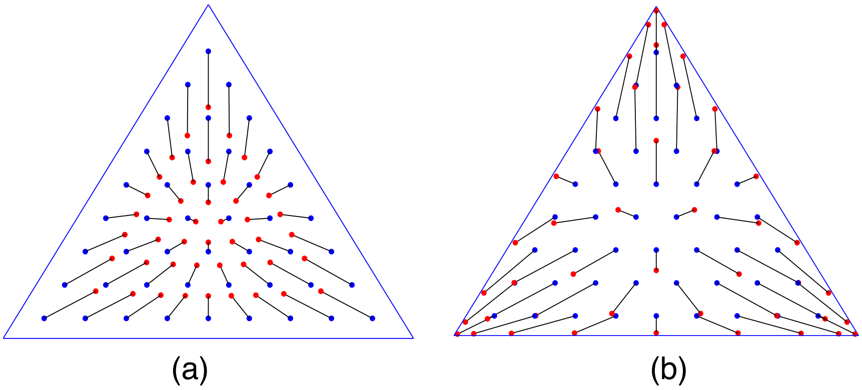


Fig. 3.3 Illustration for the powering operator in the simplex $\mathbb{P}_2$ where blue dots are input histograms and red dots are images of the powering operator on blue dots. (a) for $t = 0.6$ and (b) for $t = 2$.

3.2.2 Riemannian Manifold

A Riemannian metric g on a manifold M is a function which assigns to each point $\mathbf{x} \in M$ an inner product $g_\mathbf{x}$ on the corresponding tangent space $T_\mathbf{x}M$. Consequently, we can measure the length of a tangent vector $\mathbf{v} \in T_\mathbf{x}M$ as

$$\|\mathbf{v}\|_\mathbf{x} = \sqrt{g_\mathbf{x}(\mathbf{v}, \mathbf{v})}.$$

Let $c : [a,b] \mapsto M$ be a curve in M. Its length is defined as

$$L(c) = \int_a^b \sqrt{g_{c(t)}(c'(t), c'(t))}\mathrm{d}t,$$

where $c'(t)$ belongs to $T_{c(t)}M$. The geodesic distance $d(\mathbf{x},\mathbf{z})$ between two points $\mathbf{x}$ and $\mathbf{z}$ in the manifold M is defined as the length of the shortest curve connecting $\mathbf{x}$ and $\mathbf{z}$,

$$d(\mathbf{x},\mathbf{z}) = \inf_{c \in \mho(\mathbf{x},\mathbf{z})} L(c),$$

where $\mho(\mathbf{x},\mathbf{z})$ is a set of differential curves which connect $\mathbf{x}$ and $\mathbf{z}$ on M.

One way to specify a Riemannian metric on M is by using pull-back metrics. Let $F : M \mapsto N$ be a diffeomorphism that maps the manifold M onto the manifold N, and write h for a Riemannian metric on N. Let $T_{\mathbf{x}}M$, $T_{\mathbf{z}}N$ be the tangent spaces on the manifold M and N at $\mathbf{x}$ and $\mathbf{z}$ respectively. We can define a pull-back metric F^*h on M as follows:

$$F^*h_{\mathbf{x}}(\mathbf{u},\mathbf{v}) = h_{F(\mathbf{x})}(F_*\mathbf{u}, F_*\mathbf{v}),$$

where F_* is the push-forward map which transforms a tangent vector $\mathbf{v} \in T_xM$ to a tangent vector $F_*\mathbf{v} \in T_{F(\mathbf{x})}N$. Thus, F is an isometric mapping between the manifold M and N:

$$d_{F^*h}(\mathbf{x},\mathbf{z}) = d_h(F(\mathbf{x}), F(\mathbf{z})).$$

3.3 Fisher Information Metric for Histograms

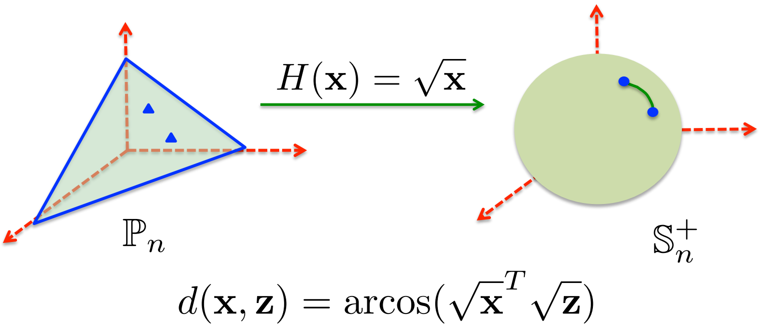

$H(\mathbf{x}) = \sqrt{\mathbf{x}}$
$\mathbb{P}_n$
$\mathbb{S}_n^+$
$d(\mathbf{x},\mathbf{z}) = \arccos(\sqrt{\mathbf{x}}^T\sqrt{\mathbf{z}})$

Fig. 3.4 Illustration for the Fisher information metric in the simplex.

In information geometry, the Fisher information metric is a particular Riemannian metric, defined on the simplex. It is well-known that the Fisher information metric can be described as a pull-back metric from the positive orthant of the sphere $\mathbb{S}_n^+$,

$$\mathbb{S}_n^+ \overset{\text{def}}{=} \left\{ \mathbf{x} \in \mathbb{R}^{n+1} \;\middle|\; \mathbf{x} \geq 0 \text{ and } \mathbf{x}^T\mathbf{x} = 1 \right\}.$$

The diffeomorphism mapping $H : \mathbb{P}_n \mapsto \mathbb{S}_n^+$ is defined as the Hellinger mapping,

$$H(\mathbf{x}) \overset{\text{def}}{=} \sqrt{\mathbf{x}},$$

where the square root is an element-wise function. The mapping H pulls-back the Euclidean metric on the positive sphere $\mathbb{S}_n^+$ to the Fisher information metric on the simplex $\mathbb{P}_n$. Thus, the geodesic distance $d(\mathbf{x}, \mathbf{z})$ between two histograms $\mathbf{x}, \mathbf{z}$ in the simplex $\mathbb{P}_n$ under the Fisher information metric is equivalent to the length of the shortest curve on the positive sphere $\mathbb{S}_n^+$ between $H(\mathbf{x})$ and $H(\mathbf{z})$,

$$d(\mathbf{x}, \mathbf{z}) = \operatorname{arcos}\left(\langle H(\mathbf{x}), H(\mathbf{z})\rangle\right) = \operatorname{arcos}\left(\left\langle \sqrt{\mathbf{x}}, \sqrt{\mathbf{z}} \right\rangle\right), \tag{3.1}$$

where $\langle \cdot, \cdot \rangle$ is the Euclidean inner product. We also give an illustration for the Fisher information metric in the simplex in Figure 3.4.

Let $G : \operatorname{int}\mathbb{P}_n \mapsto \operatorname{int}\mathbb{P}_n$ be a transformation inside the simplex. The Fisher information metric under the transformation G on the simplex $\mathbb{P}_n$, denoted as J, is a pull-back metric of the Euclidean metric on the positive sphere $\mathbb{S}_n^+$ through a transformation $F = H \circ G$. The geodesic distance that results by using J between $\mathbf{x}, \mathbf{z} \in \mathbb{P}_n$ is thus

$$d_J(\mathbf{x}, \mathbf{z}) = \operatorname{arcos}\left(\langle F(\mathbf{x}), F(\mathbf{z})\rangle\right).$$

We illustrate for this metric in Figure 3.1. Therefore, we have a family of pull-back metrics J on the simplex $\mathbb{P}_n$, parameterized by the transformation G inside the simplex $\mathbb{P}_n$. In the next section, we will present a way to learn a suitable pull-back metric J based on a family of transformations G using only unlabeled data.

3.4 Unsupervised Riemannian Metric Learning for Histograms

3.4.1 Aitchison Transformation

We consider a family of transformations G on the simplex that can be defined using Aitchison elementary perturbation and powering operations presented in Section 3.2.1. Firstly, we generalize the powering operation for a histogram and a vector, denoted as $\otimes$, so that we can have exponents that can vary for each coordinate:

$$\alpha \otimes \mathbf{x} \overset{\text{def}}{=} C(\mathbf{x}^{\alpha}) \in \operatorname{int}\mathbb{P}_n,$$

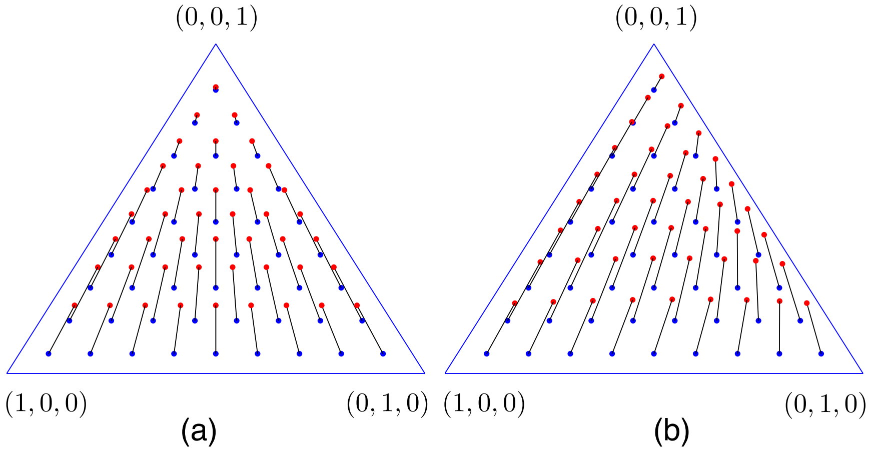


Fig. 3.5 Illustration for the generalized powering operator in the simplex $\mathbb{P}_2$ where blue dots are input histograms and red dots are images of the generalized powering operator on blue dots. (a) for $\alpha = [1, 1, 0.5]$ and (b) for $\alpha = [1.3, 1, 0.5]$.

where the power function is element-wise between vector $\mathbf{x}$ and vector α. Figure 3.5 shows an action of the generalized powering operator for histograms in the simplex $\mathbb{P}_2$.

The transformation we consider is parameterized by a vector α in the strictly positive orthant $\mathbb{R}_+^{n+1}$, and by $\lambda \in \text{int}\mathbb{P}_n$:

$$G(\mathbf{x}) = \alpha \otimes \mathbf{x} \oplus \lambda \in \text{int}\mathbb{P}_n. \tag{3.2}$$

Or, we have

$$G(\mathbf{x}) = C(\mathbf{x}^{\alpha} \bullet \lambda) \in \text{int}\mathbb{P}_n.$$

We note that $\alpha \otimes (\mathbf{x} \oplus \lambda) = (\alpha \otimes \mathbf{x}) \oplus (\alpha \otimes \lambda)$. So, for the transformation $G(\mathbf{x})$, we can interpret that vector λ under operator $\oplus$ may be considered as a translation, and vector α under operator $\otimes$ has a role as a linear mapping for a histogram $\mathbf{x}$ in the simplex. We also give an illustration for the generalized Aitchison transformation for histograms in the simplex $\mathbb{P}_2$ in Figure 3.6.

Additionally, we can express the transformation $F(\mathbf{x})$ as the element-wise square root for $G(\mathbf{x})$:

$$F(\mathbf{x}) = H \circ G(\mathbf{x}) = \sqrt{C(\mathbf{x}^{\alpha} \bullet \lambda)} \in \mathbb{S}_n^+.$$

Hence, we have a closed form for the geodesic distance under Riemannian metric J – the pull-back metric of the Euclidean metric on the positive sphere $\mathbb{S}_n^+$ through a transformation

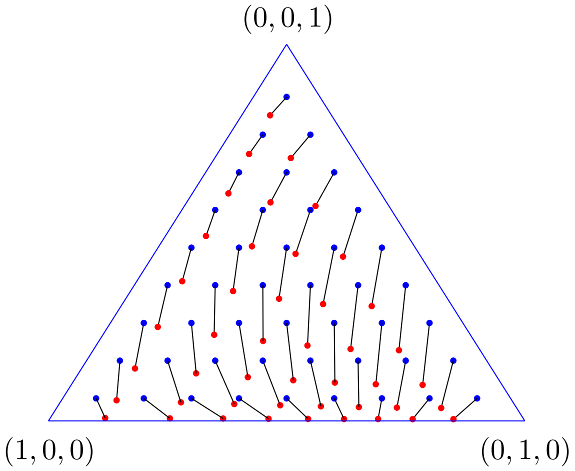


Fig. 3.6 Illustration for the generalized Aitchison transformation in the simplex $\mathbb{P}_2$ where blue dots are input histograms and red dots are images of the generalized powering operator on blue dots with $\alpha = [0.5, 1, 2]$ and $\lambda = [0.2, 0.35, 0.45]$.

$F = H \circ G$,

$$d_J(\mathbf{x}, \mathbf{z}) = \arccos\left(\langle F(\mathbf{x}), F(\mathbf{z})\rangle\right) = \arccos\left(\left\langle \sqrt{C(\mathbf{x}^{\alpha} \bullet \lambda)}, \sqrt{C(\mathbf{z}^{\alpha} \bullet \lambda)} \right\rangle\right). \tag{3.3}$$

3.4.2 Criterion

Let $\mathscr{D} = \{\mathbf{x}_i, 1 \leq i \leq m\}$ be a dataset of unlabeled histograms in the interior of the simplex. We will learn a Riemannian metric from a family of pull-back metrics J on the simplex as described in Section 3.3. Since J is parameterized by Aitchison transformation G, defined in Equation (3.2), we equivalently learn an Aitchison transformation on the simplex.

The volume element of the Riemanian metric J at point $\mathbf{x}$ is defined as:

$$\mathrm{dvol} J(\mathbf{x}) \overset{\text{def}}{=} \sqrt{\det \mathscr{G}(\mathbf{x})},$$

where $\mathscr{G}(\mathbf{x})$ is the Gram matrix, whose components

$$[\mathscr{G}]_{ij} = J(\partial_i, \partial_j),$$

where $\{\partial_i\}_{1 \leq i \leq n}$ is a basis of a tangent space $T_{\mathbf{x}}\mathbb{P}_n$ of the simplex $\mathbb{P}_n$ at point $\mathbf{x}$. Intuitively, the volume element $\mathrm{dvol}J(\mathbf{x})$ summaries the size of metric J at $\mathbf{x}$ in a scalar. Paths over areas with smaller volume will tend to be shorter than similar paths over areas with higher volume. Lebanon (2002, 2006) propose to maximize inverse volume to obtain shorter curves across

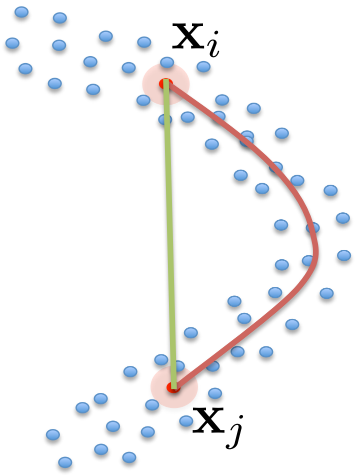


Fig. 3.7 Illustration for the intuition of a geodesic distance which should go through dense data points. The geodesic distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ should go like the red curve instead of the green line to capture the structure of the data.

densely populated regions of the simplex $\mathbb{P}_n$. Therefore, the geodesic distances will also tend to pass densely populated regions. It matches with an intuition about distance which should be measured on the lower dimensional data submanifold to capture intrinsic geometrical structure of data as illustrated in Figure 3.7. We note that volume element $\mathrm{dvol}J(\mathbf{x})$ is a homogeneous function, normalization for inverse volume is necessary to bound its quantity in optimization.

Following these intuitions, we consider a metric learning problem:

$$\begin{aligned} \max_{\alpha,\lambda} \quad & \mathscr{F} \overset{\text{def}}{=} \frac{1}{m}\sum_{i=1}^{m} \log \frac{\mathrm{dvol}J^{-1}(\mathbf{x}_i)}{\int_{\mathbb{P}_n} \mathrm{dvol}J^{-1}(\mathbf{x})\mathrm{d}\mathbf{x}} - \frac{\mu}{2}\|\log\alpha\|_2^2 \\ \text{s.t.} \quad & \lambda \in \mathrm{int}\mathbb{P}_n, \quad \alpha \in \mathbb{R}_+^{n+1}, \end{aligned} \tag{3.4}$$

where $\log\alpha$ is an element-wise function and $\mu > 0$ is a regularization parameter. We apply the logarithm function to the normalized inverse volume element in the criterion to simplify our learning procedure. We regularize this objective by the ℓ_2-norm of the element-wise logarithm α, that tends to avoid 0 values for our exponents. We do not regularize λ since $\lambda \in \mathrm{int}\mathbb{P}_n$ or $\|\lambda\|_1 = 1$.

3.4.3 Volume Element

We recall that the volume element of the Riemannian metric J at a point $\mathbf{x}$ is defined as

$$\text{dvol}J(\mathbf{x}) = \sqrt{\det\mathscr{G}(\mathbf{x})},$$

and $[\mathscr{G}]_{ij} = J(\partial_i, \partial_j)$ where $\{\partial_i\}_{1\leq i\leq n}$ is a basis of a tangent space of the simplex $T_{\mathbf{x}}\mathbb{P}_n$, described as rows of the matrix

$$U = \begin{bmatrix} 1 & \cdots & 0 & -1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & -1 \end{bmatrix} \in \mathbb{R}^{n\times(n+1)}.$$

The Gram matrix $\mathscr{G}$ is provided by Proposition 1 while its determinant is studied in Proposition 2. The proofs for these two propositions are given in the Appendix A.

Proposition 1. *Let T be a $n\times(n+1)$ matrix whose rows are $\{F_*\partial_i\}_{1\leq i\leq n}$, I is an identity matrix in $\mathbb{R}^{(n+1)\times(n+1)}$, D is a diagonal matrix in $\mathbb{R}^{(n+1)\times(n+1)}$ where*

$$\text{diag}\,(D) = \frac{\mathbf{x}^{\frac{\alpha}{2}-1} \bullet \alpha \bullet \sqrt{\lambda}}{2\sqrt{(\mathbf{x}^{\alpha} \bullet \lambda)^T \mathbf{1}}},$$

β and η are column vectors in $\mathbb{R}^{n+1}$ where

$$\beta = \mathbf{x}^{\alpha-1} \bullet \alpha \bullet \lambda$$

and

$$\eta = \frac{\mathbf{x}}{\alpha}\frac{1}{(\mathbf{x}^{\alpha} \bullet \lambda)^T \mathbf{1}}.$$

We have $T = U(I-\beta\eta^T)D$, and the Gram matrix is given by

$$\mathscr{G} = TT^T = U(I-\beta\eta^T)D^2(I-\beta\eta^T)^T U^T.$$

Proposition 2. *Let $\prod(\mathbf{x})$ be a function of product of all elements in vector $\mathbf{x}$. The determinant of the Gram matrix $\mathscr{G}$ is*

$$det\mathscr{G} \propto \frac{\left(\left(\frac{\mathbf{x}}{\alpha}\right)^T \mathbf{1}\right)^2 \prod\left(\mathbf{x}^{\alpha-2}\right)}{\left((\mathbf{x}^{\alpha} \bullet \lambda)^T \mathbf{1}\right)^{n+1}}$$

Algorithm 3 Gradient Ascent using Contrastive Divergence

```
Input: data $(\mathbf{x}_i)_{1\leq i\leq m}$, gradient step size $t_0^\alpha$ and $t_0^\lambda$, initial vectors $\alpha_0$, $\lambda_0$ and a tolerance $\varepsilon$.
Set $t \leftarrow 1$.
Set $\alpha_\mathbf{t} \leftarrow \alpha_\mathbf{0}$.
Set $\lambda_\mathbf{t} \leftarrow \lambda_\mathbf{0}$.
repeat
    Use Metropolis-Hasting sampling algorithm where its proposal distribution is logistic normal distribution to transform training data $(\mathbf{x}_i)_{1\leq i\leq m}$ into data drawn from $p(\mathbf{x})$.
    Compute gradient of the objective function with respect to $\alpha$, $\lambda$ using Proposition 3.
    Update $\alpha_{\mathbf{t+1}} \leftarrow \Psi\Big(\alpha_\mathbf{t} + \frac{t_0^\alpha}{\sqrt{t}}\frac{\partial\mathscr{F}}{\partial\alpha}\Big)$.
    Update $\lambda_{\mathbf{t+1}} \leftarrow C\left[\lambda_\mathbf{t} \bullet \exp\left(\frac{t_0^\lambda}{\sqrt{t}}\frac{\partial\mathscr{F}}{\partial\lambda}\right)\right]$.
    Set $t \leftarrow t+1$.
until $(t > t_{\max})$ or $(\|\alpha_\mathbf{t} - \alpha_\mathbf{t-1}\| < \varepsilon)$ or $(\|\lambda_\mathbf{t} - \lambda_\mathbf{t-1}\| < \varepsilon)$.
Output: vectors $\alpha_\mathbf{t}$ and $\lambda_\mathbf{t}$.
```

3.4.4 Gradient Ascent using Contrastive Divergences

The main obstacle of our optimization problem is the normalization term of the inverse volume element since it is not known in closed form. However, we can bypass this factor to compute a partial derivative of the objective function $\mathscr{F}$ with respect to α and λ as given in Proposition 3. Its proof is given in the Appendix A.

Proposition 3. *Let $E(\cdot)_\mathbf{X}$ denote the expectation of $\cdot$ given the data distribution $\mathbf{X}$, and a distribution,*

$$p(\mathbf{x}) = \frac{dvolJ^{-1}(\mathbf{x})}{\int_{\mathbb{P}_n} dvolJ^{-1}(\mathbf{z})d\mathbf{z}}. \tag{3.5}$$

The partial derivative of the objective function $\mathscr{F}$ with respect to α, λ in the optimization problem are:

$$\frac{\partial\mathscr{F}}{\partial\alpha} = \frac{1}{m}\sum_{i=1}^{m}\frac{\partial\log dvolJ^{-1}(\mathbf{x}_i)}{\partial\alpha} - E\left(\frac{\partial\log dvolJ^{-1}(\mathbf{x})}{\partial\alpha}\right)_{p(\mathbf{x})} - \mu\frac{\log\alpha}{\alpha}$$

where

$$\frac{\partial\log dvolJ^{-1}(\mathbf{x})}{\partial\alpha} = \frac{n+1}{2\left(\mathbf{x}^\alpha\bullet\lambda\right)^T\mathbf{1}}(\mathbf{x}^\alpha\bullet\lambda\bullet\log\mathbf{x}) + \frac{1}{\left(\frac{\mathbf{x}}{\alpha}\right)^T\mathbf{1}}\left(\frac{\mathbf{x}}{\alpha\bullet\alpha}\right) - \frac{1}{2}\log\mathbf{x}$$

and

$$\frac{\partial \mathscr{F}}{\partial \lambda} = \frac{1}{m}\sum_{i=1}^{m} \frac{\partial \log dvolJ^{-1}(\mathbf{x_i})}{\partial \lambda} - E\left(\frac{\partial \log dvolJ^{-1}(\mathbf{x})}{\partial \lambda}\right)_{p(\mathbf{x})}$$

where

$$\frac{\partial \log dvolJ^{-1}(\mathbf{x})}{\partial \lambda} = \frac{n+1}{2\,(\mathbf{x}^{\alpha} \bullet \lambda)^{T}\,\mathbf{1}}\mathbf{x}^{\alpha},$$

where $\log \mathbf{x}$, $\mathbf{x}^{\mathbf{z}}$ *and* $\frac{\mathbf{x}}{\mathbf{z}}$ *are element-wise logarithm, power and division functions respectively.*

We propose to approximate the expectation $E(\cdot)_{p(\mathbf{x})}$ that appears in Proposition 3 by drawing samples from the distribution $p(\mathbf{x})$. Since the partition function $\int_{\mathbb{P}_n} \mathrm{dvol}J^{-1}(\mathbf{z})\mathrm{d}\mathbf{z}$ is not known in closed form, we can not draw samples directly from $p(\mathbf{x})$. However, we can use Markov Chain Monte Carlo (MCMC) sampling methods to draw such samples. Because we only need to compute the ratio of two probabilities, ${}^{p(\mathbf{x})}/_{p(\mathbf{z})}$ an approximation for the partition function itself is not required. Moreover, Hinton (2002) suggests that only a few cycles of MCMC can provide in certain settings a useful approximation. The intuition is that the data have moved from the target distribution – training data – towards the proposed distribution $p(\mathbf{x})$ after a few iterations.

We propose to use a Metropolis-Hasting sampling method with a logistic normal distribution (Aitchison and Shen, 1980) proposal. We note that the logistic normal distribution is also a by-product of Aitchison's simplicial geometry. We apply contrastive divergences (Hinton, 2002) to compute approximations of the partial derivative of $\mathscr{F}$ as shown in the proof of the Proposition 3.

We propose to use a gradient ascent to optimize for the metric learning problem following the results in the Proposition 3. At iteration t, we can update α, λ using preset step size $\frac{t_0^{\alpha}}{\sqrt{t}}$ and $\frac{t_0^{\lambda}}{\sqrt{t}}$ respectively, as follow

$$\alpha_{\mathbf{t+1}} = \Psi\left(\alpha_{\mathbf{t}} + \frac{t_0^{\alpha}}{\sqrt{t}}\frac{\partial \mathscr{F}}{\partial \alpha}\right),$$

and

$$\lambda_{\mathbf{t+1}} = C\left(\lambda_{\mathbf{t}} \bullet \exp\left(\frac{t_0^{\lambda}}{\sqrt{t}}\frac{\partial \mathscr{F}}{\partial \lambda}\right)\right),$$

where $\Psi(\mathbf{x})$ is the projection of $\mathbf{x}$ on the positive orthant offset by a small minimum threshold $\varepsilon = 10^{-20}$, namely the set of all vectors whose coordinates are larger or equal to 10^{-20}, and $\bullet$ is the Schur product – element-wise product – between vectors or matrices, and the exp operator is here applied element-wise. Since we have a constraint $\lambda \in \mathrm{int}\mathbb{P}_n$ in the optimization problem (3.4), we use an exponentiated gradient update for λ (Kivinen and Warmuth, 1997).

We recall that computing the normalization term for a specific transformation in (Lebanon, 2002, 2006) takes $O(n^2 \log n)$ by careful dynamic programming. So, our proposal is more efficient and general than Lebanon's approach. A pseudo-code for the projected gradient ascent algorithm is summarized in Algorithm 3.

We also note that the optimization problem (3.4) can be interpreted as maximizing log-likelihood for the probabilistic model on the simplex (Equation (3.5) and Proposition 2) which assigns probabilities propositional to the inverse Riemannian volume element, with a regularization.

3.5 Locally Sensitive Hashing to Approximate k-Nearest Neighbors Search

We recall that our proposed family of distances (Equation (3.3) in Section 3.4.1) is the pull-back metric of the Euclidean metric on the positive sphere through a composition transformation of Hellinger mapping and Aitchison transformation. Equivalently, it can be considered as measuring the angle between two mapped vectors from the composition transformation. So, we can apply the Locally Sensitive Hashing family proposed by Charikar (2002) to approximate k-nearest neighbors search.

For two histogram vectors $\mathbf{x}, \mathbf{z} \in \text{int}\mathbb{P}_n$, we have the corresponding mapped vector

$$\bar{\mathbf{x}} = F(\mathbf{x}), \bar{\mathbf{z}} = F(\mathbf{z}) \in \mathbb{S}_n^+$$

via the composition transformation F. Charikar (2002) defines a hash function

$$h_{\mathbf{r}}(\bar{\mathbf{x}}) = \text{sign}(\mathbf{r}^T \bar{\mathbf{x}}),$$

where $\mathbf{r}$ is a random unit-length vector in $\mathbb{R}^{n+1}$. The hash function can be considered as a randomly chosen hyperplane to partition the space into two half-spaces. The probability of collision is as follow

$$\Pr\left[h_{\mathbf{r}}(\bar{\mathbf{x}}) = h_{\mathbf{r}}(\bar{\mathbf{z}})\right] = 1 - \frac{d_J(\mathbf{x}, \mathbf{z})}{\pi}.$$

For a random vector $\mathbf{r}$, we have a hash-bit $h_{\mathbf{r}}(\cdot)$ for each histogram $\mathbf{x}$ in a database. We use b random vectors for a total b hash functions to obtain hash keys (b hash bits) for each histogram. For a query histogram $\mathbf{z}$, we apply the same b hash functions, and then use

Table 3.1 Properties of datasets and their corresponding experimental parameters.

Dataset	#Samples	#Class	Feature	Rep	#Dim	#Run
MIT Scene	1600	8	SIFT	BoF	200	100
UIUC Scene	3000	15	SIFT	BoF	200	100
OXFORD Flower	1360	17	SIFT	BoF	200	100
CALTECH-101	3060	102	SIFT	BoF	200	100
20 News Group	10000	20	BoW	LDA	200	100
Reuters	2500	10	BoW	LDA	200	100
MNIST-60K	60000	10	Normalized Intensity		784	4
CIFAR-10	60000	10	BoW	SIFT	200	4

the approximated similarity search method in (Charikar, 2002) which requires to search $O(m^{1/(1+\varepsilon)})$ histograms for $k = 1$ approximated nearest neighbor.

3.6 Related Work

Lebanon's use of Aitchison's perturbation operator provided the main inspiration for the metric learning approach advocated in this work (2002; 2006). We propose to extend this idea to other operations in the simplex. We also propose to adapt the contrastive divergence method for the purpose of computing a gradient to maximize inverse volumes, whereas Lebanon uses an approximation for the partition function which only applies to the perturbation transformation. We also show in the experimental section that our approach can also be used in Lebanon's original setting.

In chapter 2, this thesis proposes the generalized Aitchison embeddings to learn metrics for histograms. Rather than using Aitchison transformations, in chapter 2, we focus on a different family of tools, Aitchison maps, that can map points in the simplex onto a Euclidean space $\mathbb{R}^d$. We recall that we propose, in chapter 2, to learn simultaneously the parameters of such maps and the metric (a Mahalanobis metric) on $\mathbb{R}^d$ that will be used on such representations. This is related, although very different, from the approach we propose here that learns in an unsupervised way a map from and onto the simplex, to be used with Fisher's information metric.

3.7 Experiments

3.7.1 Clustering application with k-medoids

Datasets and Experimental Setting

We use the k-medoids clustering algorithm seeded with different metrics and compute their clusters. We set the number of clusters k equal to the number of classes in corresponding datasets. To evaluate the adequacy of a metric for given data, we check that these clusters agree with a class typology provided for these points[1]. We test our method on 6 benchmark datasets. Table 3.1 displays their properties and parameters. These datasets include different kinds of data such as scene images in MIT Scene[2] and UIUC Scene[3] datasets, flower images in Oxford Flower[4] dataset, object images in CALTECH-101[5] dataset and texts in Reuters[6] and 20 News Group[7] datasets.

Implementation Notes

For image datasets, we compute dense SIFT features by operating a SIFT descriptor of 16×16 patches computed over each pixel of an image. We also convert images into gray scale ones before computing dense SIFT to improve robustness. We use the LabelMe toolbox[8] for computing dense SIFT features. Then, we use bag-of-features (BoF) to represent for each image as a histogram, the size of dictionary for visual words is set 200.

For text datasets, we calculate bag of words (BoW) for each document, and then compute topic modelling to reduce the dimension of histograms using the *gensim* toolbox[9]. Each document can be thus described as a histogram of topics (Blei et al., 2003; Blei and Lafferty, 2009).

[1] In this setting for clustering application, we process with unlabelled data (for both learning the distance and applying to k-medoids clustering method). Labels are only used to evaluate the clustering results. We use k-medoids clustering algorithm instead of a traditional k-means since it is not trivial to compute a mean with respect to a specific distance (i.e our proposed distance).

[2] http://people.csail.mit.edu/torralba/code/spatialenve-lope/

[3] http://www.cs.illinois.edu/homes/slazebni/research/

[4] http://www.robots.ox.ac.uk/∼vgg/data/flowers/17/

[5] http://www.vision.caltech.edu/Image_Datasets/Cal-tech101/

[6] http://archive.ics.uci.edu/ml/datasets/Reuters–21578+Text +Categorization+Collection

[7] http://qwone.com/∼jason/20Newsgroups/

[8] http://new-labelme.csail.mit.edu/Release3.0/

[9] http://radimrehurek.com/gensim/

We use the PMTK3 toolbox[10] implementation of the k-medoids algorithm. For each metric, we performs k-medoids algorithm 100 times with different random initializations, resulting in box-plots for our error statistics.

We may use $\alpha_{\mathbf{0}} = [1,1,\cdots,1]$ and $\lambda_{\mathbf{0}} = C[\alpha_{\mathbf{0}}]$ for initialization since our proposed distance (Equation (3.3) in Section 3.4.1) is equivalent to the Fisher Information Metric (Equation (3.1) in Section 3.3) at these values for α and λ. We also propose to use an internal criterion - Davies-Bouldin index (Davies and Bouldin, 1979) to select parameters via applying k-medoids clustering algorithm. We choose gradient step size t_0^{α} and t_0^{λ} from the sets $\frac{1}{\|\frac{\partial \mathscr{F}}{\partial \alpha}(\alpha_{\mathbf{0}},\lambda_{\mathbf{0}})\|_2}\{.001,.005,.01,.05,.1,.5\}$ and $\frac{1}{\|\frac{\partial \mathscr{F}}{\partial \lambda}(\alpha_{\mathbf{0}},\lambda_{\mathbf{0}})\|_2}\{.001,.005,.01,.05,.1,.5\}$ respectively and μ from $\{0.1,1,10\}$. We set maximum iterations t_{max} = 10000 and a tolerance $\varepsilon = 10^{-5}$. We also set 5 cycles for Metropolis-Hasting sampling algorithm to transform training data into data drawn from $p(\mathbf{x})$. The logistic normal distribution is used as the proposal distribution for Metropolis-Hasting algorithm where its mean is training data point, and its covariance is set $0.01I$ where I is an identity matrix.

Metrics and Metric Learning Baseline Method

We use usual metrics on the simplex such as the Euclidean, the total variation, χ^2 and Hellinger distances. We recall that the Hellinger distance between two histograms $\mathbf{x}$ and $\mathbf{z}$ in the simplex $\mathbb{P}_n$ is

$$d_{\text{Hellinger}}(\mathbf{x},\mathbf{z}) = \left(\sqrt{\mathbf{x}} - \sqrt{\mathbf{z}}\right)^T \left(\sqrt{\mathbf{x}} - \sqrt{\mathbf{z}}\right),$$

where $\sqrt{\mathbf{x}}$ is an element-wise square root for vector $\mathbf{x}$. We also consider the cosine similarity as suggested in (Lebanon, 2002, 2006) and the most popular of Aitchison mappings, known as isometric log-ratio (**ilr**) (Egozcue et al., 2003; Le and Cuturi, 2013, 2014). Additionally, we compare our proposal with the work of (Lebanon, 2002, 2006) implemented using our algorithm to maximize inverse volumes, denoted as `pFIM`.

$\mathbf{F}_{\beta}$ measure

We use the $\mathbf{F}_{\beta}$ measure to compare results of k-medoids clustering with different metrics (Manning et al., 2008). The intuition is that a pair of histograms is assigned to the same cluster if and only if they are in the same class and otherwise[11]. So, a true positive (TP) decision assigns a pair of histograms in the same class to the same cluster while a true negative (TN) one assigns a pair of histograms in the different classes to the different clusters.

[10] https://github.com/probml/pmtk3

[11] We note that the class label y_i corresponding for a histogram $\mathbf{x_i}$, for all $1 \leq i \leq m$, is only used for evaluation. In training procedure, only histograms (without labels) are available.

We have two types of errors. A false positive (FP) decision assigns a pair of histograms of different classes to the same cluster, and a false negative (FN) one assigns a pair of histograms of the same class to different clusters. Therefore, we can measure the precision

$$\mathbf{P} = \frac{TP}{TP+FP}$$

and recall

$$\mathbf{R} = \frac{TP}{TP+FN}.$$

Since we have more pairs of histograms in different classes than in the same class, we need to penalize false negative more strongly than false positives. $\mathbf{F}_\beta$ measure can take into account of that idea through a scalar $\beta > 1$ as

$$\mathbf{F}_\beta = \frac{\left(\beta^2+1\right)\mathbf{PR}}{\beta^2\mathbf{P}+\mathbf{R}}.$$

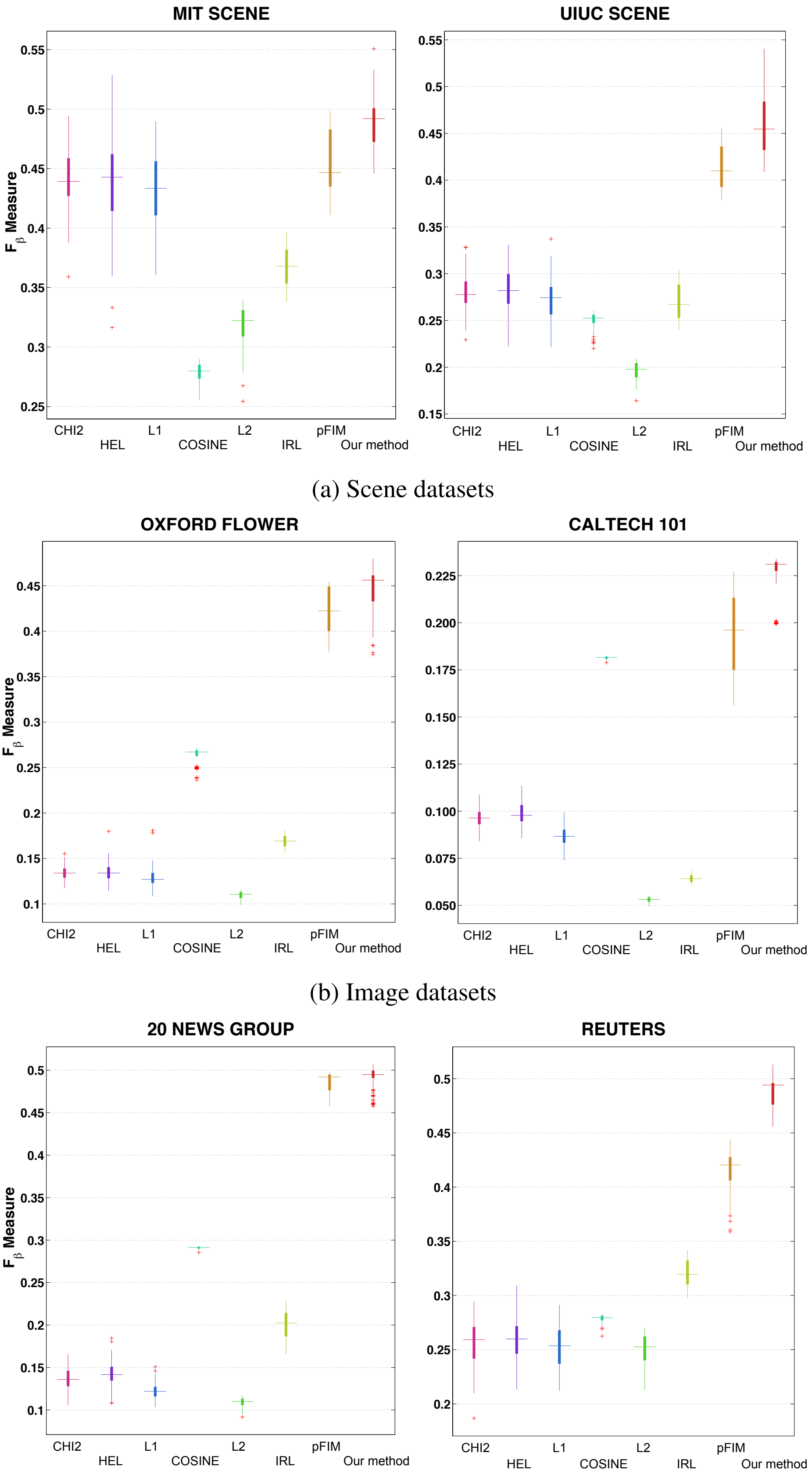
By replacing $\mathbf{P}$ and $\mathbf{R}$ into $\mathbf{F}_\beta$, we note that $\mathbf{F}_\beta$ penalizes false negative β^2 times more than false positives. So, let $\mathtt{D}$ and $\mathtt{S}$ be sets of pairs of histograms in different and same classes of a dataset respectively, we can set

$$\beta = \sqrt{\frac{|\mathtt{D}|}{|\mathtt{S}|}}$$

where $|\cdot|$ denotes a cardinality of a set.

Results

Figure 3.8 illustrates $\mathbf{F}_\beta$ measure for k-medoids clustering on 6 benchmark datasets where we denote CHI2 for χ^2 distance, HEL for Hellinger distance, L1 for total variation distance, COSINE for cosine similarity, L2 for Euclidean distance, ILR for isometric log-ratio mapping - the most popular Aitchison mapping and pFIM for Fisher information metric pameterized by a perturbation transformation (Lebanon, 2002, 2006). It shows that the Euclidean distance, which fails to incorporate the geometrical constraints in the simplex, does not work well for histogram data. Some popular distances for histograms such as total variation distance, χ^2 distance and Hellinger distance as well as the Aitchison mapping - **ilr** give better results than the simple Euclidean distance. Cosine similarity (or angular distance) has a better or comparative performance to these popular distances for histograms, except on MIT Scene and UIUC Scene datasets. The performances of Riemannian metric learning using Aitchison transformations is significantly better, notably on the UIUC Scene, Oxford Flower, 20 News Group and Reuters datasets.



(a) Scene datasets

(b) Image datasets

(c) Text datasets

Fig. 3.8 $\mathbf{F}_\beta$ measure for K-medoids clustering.

3.7.2 *k*-Nearest Neighbors Classification with Locally Sensitive Hashing

We also carry out *k*-nearest neighbors classification with locally sensitive hashing. We use 2 large datasets MNIST-60K[12] and CIFAR-10[13]. Each dataset consists of 60000 images, we randomly choose 50000 images as a database and use the rest 10000 images for queries. Table 3.1 displays their properties and parameters.

To handle large datasets, we propose a variance of Algorithm 3 by using a mini-batch stochastic gradient (Bengio, 2007). Instead of using the whole samples at each iteration to compute gradients, we randomly choose a small subset of the order of 10 samples as suggested in (Bengio, 2007) to speed up the learning procedure.

As baselines, we consider the Euclidean, a Mahalanobis distance learned by using Large Margin Nearest Neighbors (`LMNN`) Weinberger et al. (2006); Weinberger and Saul (2009) algorithm. We also consider Hellinger distance and Hellinger mapping with a Mahalanobis distance learned by using `LMNN`, denoted as `HELLINGER-LMNN`, as well as the approach of (Lebanon, 2002, 2006) as mentioned in Section 3.7.1.

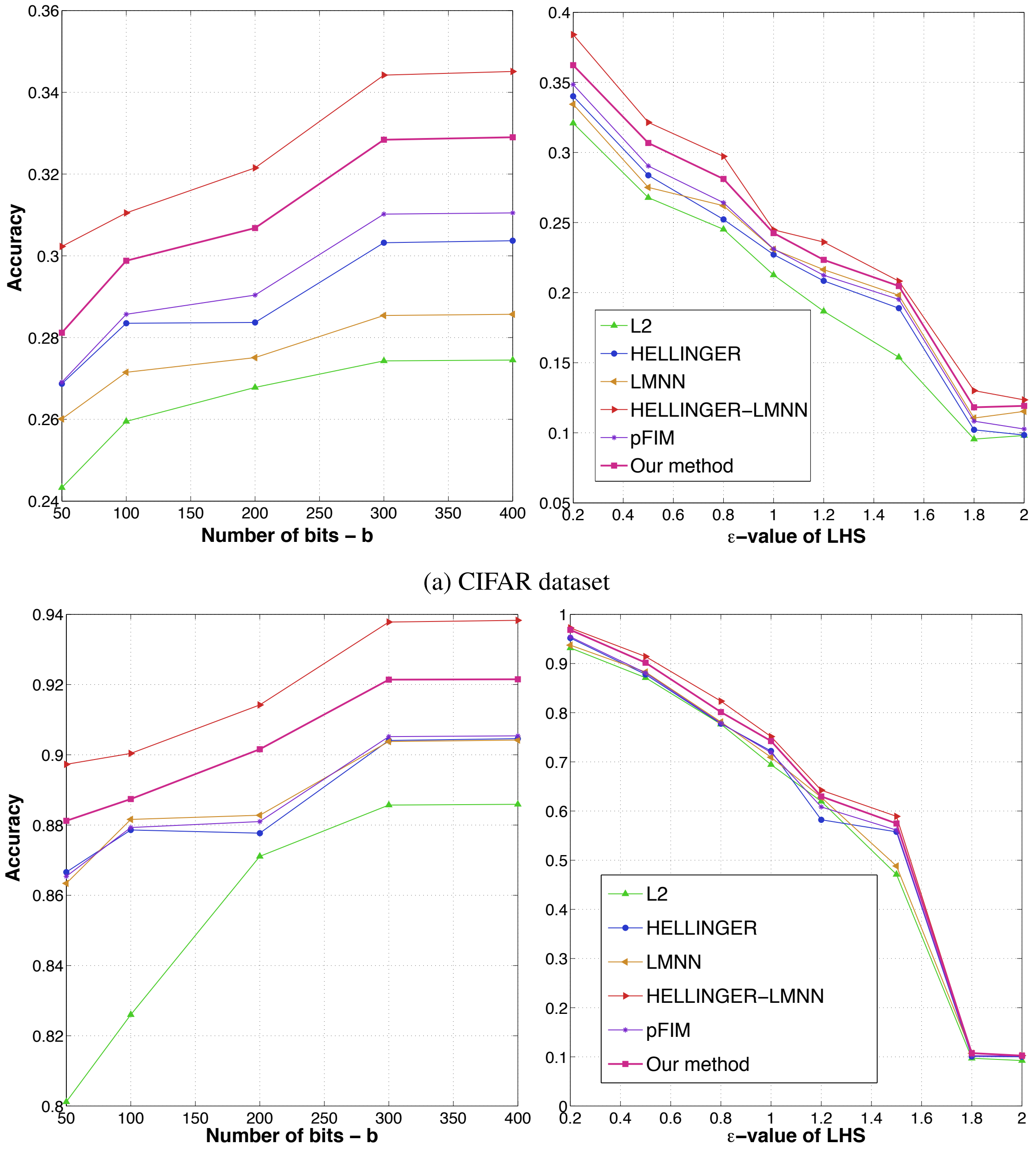
Figure 3.9 illustrates our results on MNIST-60K and CIFAR-10 datasets. Our approach outperforms other alternative distances except `HELLINGER-LMNN` which should be expected, given that it is a state of the art *supervised* metric learning approach for histograms. Figure 3.9 also shows that Euclidean distance and a straightforward application of `LMNN` do not work well for histogram data. We insist that `HELLINGER-LMNN` uses labels to learn a Mahalanobis matrix while our approach do *not* consider them.

3.8 Summary

We propose a new unsupervised metric learning approach for histograms that leverages Aitchison transformations for histograms in the simplex. These transformations are learned with the maximum inverse volume framework of Lebanon (2006). We provide a new algorithm to carry out such a maximization using contrastive divergences which solves the key obstacle - the partition function for a general case. We show empirically that our proposal can learn effectively histogram metrics for unlabeled data. It outperforms alternative popular metrics for histograms such as χ^2, Hellinger, total variation, Euclidean distance, cosine similarity and an Aitchison map (**ilr**) in clustering problem on many benchmark datasets.

[12] http://yann.lecun.com/exdb/mnist/

[13] http://www.cs.toronto.edu/∼kriz/cifar.html



(a) CIFAR dataset

(b) MNIST dataset

Fig. 3.9 Performances of k-Nearest neighbors with locally sensitive hashing on CIFAR-10 and MNIST-60K datasets, averaged over 4 repetitions where we denote L2 for Euclidean distance, HELLINGER for Hellinger distance, LMNN for Mahalanobis distance learned by Large Margin Nearest Neighbor Weinberger et al. (2006); Weinberger and Saul (2009) algorithm, HELLINGER-LMNN for LMNN learned from data mapped by Hellinger transformation and pFIM for Fisher information metric pameterized by a perturbation transformation (Lebanon, 2002, 2006). For figure Accuracy vs Number of bits - b, we set ε=0.5. For figure Accuracy vs ε-value, we set b=200. All figures are reported with k=7, since in our experiments, the relative performance of these classifiers does not vary with k.

Additionally, it also improves the performances of *k*-nearest neighbors classification with locally sensitive hashing for large datasets.

Chapter 4

Hierarchical Spatial Matching Kernel

In this chapter, we propose a new kernel, namely hierarchical spatial matching kernel for images which use the bag of features approach for representation.

4.1 Overview

Image categorization is the task of classifying a given image into a suitable semantic category. The semantic category can be defined as the depicting of a whole image such as a forest, a mountain or a beach, or of the presence of an interesting object such as an airplane, a chair or a strawberry. Among existing methods for image categorization, the bag of features (BoF) model is one of the most popular and efficient. It considers an image as a set of unordered features extracted from local patches. The features are quantized into discrete visual words, with sets of all visual words referred to as a dictionary. A histogram of visual words is then computed to represent an image. One of the main weaknesses in this model is that it discards the spatial information of local features in the image. To overcome it, spatial pyramid matching (SPM) proposed by Lazebnik et al. (2006), an extension of the BoF model, uses the aggregated statistics of the local features on fixed sub-regions. It uses a sequence of grids at several different levels of resolution to partition the image into sub-regions, and then computes a BoF histogram for each sub-region at each level of resolution. Thus, the representation of the whole image is the concatenation vector of all the histograms.

Empirically, it is realized that to obtain good performances, the BoF model and SPM have to be applied together with specific nonlinear Mercer kernels (Boughhorbel et al., 2004) such as the intersection kernel or χ^2 kernel. When the kernel function is proved to be positive definite, Mercer kernels guarantee the optimal solutions in learning algorithms. Intersection kernel for BoF histogram can be used in Support Vector Machine (SVM) based image categorization and object recognition tasks. The Pyramid Match Kernel proposed

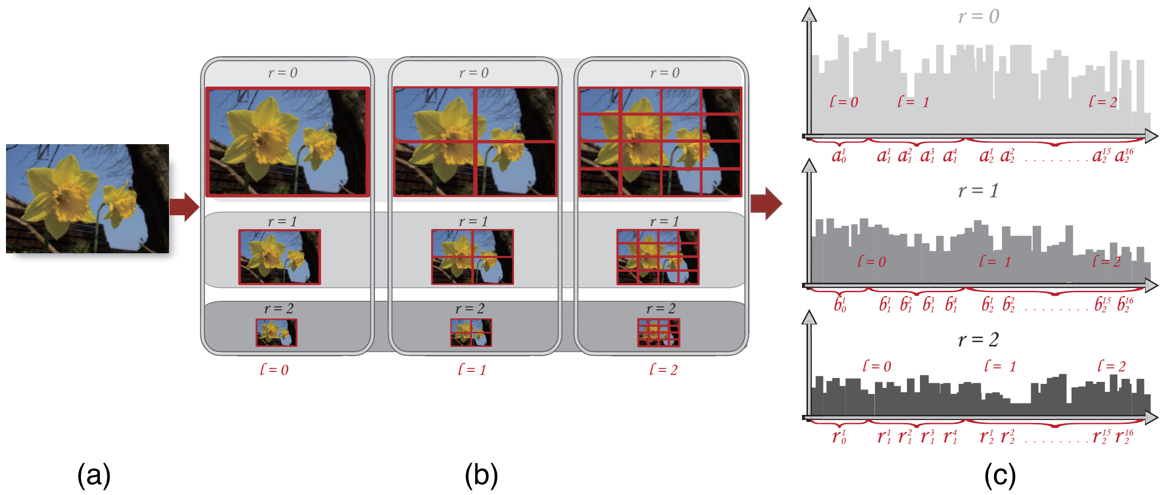

r = 0
r = 1
r = 2
ℓ = 0
ℓ = 1
ℓ = 2
a_0^1 a_1^1 a_1^2 a_1^3 a_1^4 a_2^1 a_2^2 .... a_2^{15} a_2^{16}
b_0^1 b_1^1 b_1^2 b_1^3 b_1^4 b_2^1 b_2^2 .... b_2^{15} b_2^{16}
r_0^1 r_1^1 r_1^2 r_1^3 r_1^4 r_2^1 r_2^2 .... r_2^{15} r_2^{16}
(a)
(b)
(c)

Fig. 4.1 An illustration for hierarchical spatial matching kernel (HSMK) applied to images X and Y with $L = 2$ and $R = 2$ (a). HSMK first partitions the images into $2^\ell \times 2^\ell$ sub-regions with $\ell = 0, 1, 2$ as spatial pyramid matching kernel (SPMK) (b). However, HSMK applies the coarse-to-fine model for each sub-region by considering it on a sequence of different resolutions $2^{-r} \times 2^{-r}$ with $r = 0, 1, 2$ (c). The weight set notes $\left(a_i^j, b_i^j, r_i^j\right)$, where $i = 0, 1, 2$ and $j = 1, 2, \ldots, 16$. The Equation (4.3) with the weight vector achieved from the uniform combination of basic kernels – a special case of multiple kernel learning where the weights are uniform – is applied to obtain better similarity measurement between sub-regions instead of using the bag of words model as in SPMK.

by Grauman and Darrell (2005) is suitable for discriminative classification with unordered sets of local features. This means that a kernel-based discriminative classifier is trained by calculating the similarity between each pair of sets of unordered features in the whole images or in the sub-regions. It is also well known that numerous problems exist in image categorization such as the presence of heavy clutter, occlusion, different viewpoints, and intra-class variety. In addition, the sets of features have various cardinalities and are lacking in the concept of spatial order. SPM embeds a part of the spatial information over the whole image by partitioning an image into a sequence of sub-regions, but in order to measure the similarity between corresponding sub-regions, it still applies the BoF model, which is known to be confined when dealing with sets of unordered features.

In this chapter, we propose a new kernel function based on a *coarse to fine* approach and we call it a hierarchical spatial matching kernel (HSMK). HSMK allows not only capturing spatial order of local features, but also accurately measuring the similarity between sets of unordered local features in sub-regions. In HSMK, the coarse to fine model on sub-regions is realized by using multi-resolutions, and thus our feature descriptors capture not only the local details from fine resolution sub-regions, but also global information from coarse resolution ones. In addition, matching based on our coarse-to-fine model involves a hierarchical process. This indicates that a feature that does not find its correspondence in a fine resolution still has a possibility of having its correspondence in a coarse resolution. Accordingly, our proposed kernel can achieve a better similarity measurement between sub-regions than SPM.

4.2 Related work

The BoF models are popular and powerful method for image categorization and generic object recognition. This framework works by extracting local image features, quantizing them according to typical clustering method such as k-means vector quantization, accumulating histograms of the *visual word* over the input image, and then, classifying the histograms with simple classifiers such as an SVM and Boosting. However, the traditional BoF model discards the context information for spatial layout of an image. Many recent methods have been proposed to improve the problems of traditional BoF model.

Boiman et al. (2008); Fei-Fei et al. (2004) used generative methods for quantization of local image descriptors, and modeled the co-occurrence of visual words. While Moosmann et al. (2008); Yang et al. (2008) introduced extremely randomized clustering forests to generate discriminative visual words using clustering decision trees. Mairal et al. (2009); Yang et al. (2009) modeled data vectors as sparse linear combinations that is called sparse coding methods. They improved the visual dictionary in terms of discriminative ability or lower reconstruction error instead of using the quantization by k-means clustering.

On the other hand, Lazebnik et al. (2006) proposed SPM method that can capture the spatial layout of features ignored in the BoF model. The SPM is particularly effective as well as being easy and simple to construct. It is used as a major part in many state-of-the-art frameworks in image categorization (Gehler and Nowozin, 2009). Grauman and Darrell (2005) proposed a fast kernel function called the pyramid match using multi-resolution histograms. The pyramid match hierarchically measures similarity between histograms which consist of sets of features extracted from the finest resolution to the coarsest one. The proposed kernel approximates the optimal partial matching by computing a weighted intersection over multi-resolution histograms for classification and regression tasks. Wang and Wang (2010) proposed a multiple scale learning (MSL) framework in which multiple kernel learning (MKL) is employed to learn the optimal weights instead of using predefined weights of SPM. The multiple scale learning method can determine the optimal combination of base kernels constructed in different image scales for visual categorization.

SPM is often applied with a nonlinear kernel such as the intersection kernel or χ^2 kernel. This requires high computation and large storage. Maji et al. (2008) proposed an approximation to improve efficiency in building the histogram intersection kernel, but efficiency can be attained merely by using pre-computed auxiliary tables which are considered as a type of pre-trained nonlinear support vector machines (SVM). To give SPM the linearity needed to deal with large datasets, Yang et al. (2009) proposed a linear SPM with spare coding (ScSPM), in which a linear kernel is chosen instead of a nonlinear kernel due to the more linearly separable property of sparse features.

Our proposed kernel concentrates on improvement of the similarity measurement between sub-regions by using a *coarse to fine* model instead of the BoF model used in SPM. We consider the sub-regions on a sequence of different resolutions as the pyramid matching kernel (PMK) (Grauman and Darrell, 2005). Furthermore, instead of using the priori weight vector for basic intersection kernels to penalize across different resolutions, we reformulate the problem into a uniform combination of basic kernels – a special case of multiple kernel learning where the weights are uniform – to obtain it more effectively. In addition, our proposed kernel can deal with different cardinalities of sets of unordered features by applying the square root diagonal normalization (Schölkopf and Smola, 2001) for each intersection kernel, which is not considered in PMK.

4.3 Hierarchical Spatial Matching Kernel

In this section, we first describe the original formulation of SPM and then introduce our proposed HSMK, which uses a *coarse to fine* model as a basic for improving SPM.

4.3.1 Spatial Pyramid Matching

Each image is represented by a set of vectors in the n-dimensional feature space. Features are quantized into discrete types called visual words by using k-means clustering or sparse coding. The matching between features turns into a comparison between discrete corresponding types. This means that they are matched if they are in the same type and unmatched otherwise.

SPM constructs a sequence of different scales with $\ell = 0, 1, 2, \ldots, L$ on an image. In each scale, it partitions the image into $2^\ell \times 2^\ell$ sub-regions and applies the BoF model to measure the similarity between sub-regions. Let X and Y be two sets of vectors in the n-dimensional feature space. The similarity between two sets at scale ℓ is the sum of the similarity between all corresponding sub-regions:

$$\mathtt{k}_\ell(X,Y) = \sum_{i=1}^{2^{2\ell}} \mathtt{I}(X_i^\ell, Y_i^\ell),$$

where X_i^ℓ is the set of feature descriptors in the i^{th} sub-region at scale ℓ of the image vector set X, and $\mathtt{I}$ is the intersection kernel between X_i^ℓ and Y_i^ℓ, formulated as:

$$\mathtt{I}(X_i^\ell, Y_i^\ell) = \sum_{j=1}^{V} \min(\mathrm{H}_{X_i^\ell}(j), \mathrm{H}_{Y_i^\ell}(j)),$$

where V is the total number of visual words and $\mathrm{H}_\alpha(j)$ is the number of occurrences of the j^{th} visual word which is obtained by quantizing feature descriptors in the set α. Finally, the SPM kernel (SPMK) is the sum of weighted similarity over the scale sequence:

$$\mathtt{k}(X,Y) = \frac{1}{2^L}\mathtt{k}_0(X,Y) + \sum_{\ell=1}^{L} \frac{1}{2^{L-\ell+1}}\mathtt{k}_\ell(X,Y).$$

The weight $\frac{1}{2^{L-\ell+1}}$ associated with scale ℓ is inversely proportional to the sub-region width at that scale. This weight is used to penalize the matching since it is easier to find the matches in the larger regions. We remark that all the matches found at scale ℓ are also included in a finer scale $\ell - \zeta$ with $\zeta > 0$.

4.3.2 The proposed kernel: Hierarchical Spatial Matching Kernel

To improve efficiency in achieving the similarity measurement between sub-regions, we use a *coarse to fine* model on sub-regions by mapping them into a sequence of different resolutions $2^{-r} \times 2^{-r}$ with $r = 0, 1, 2, \ldots, R$ as in (Grauman and Darrell, 2005).

X_i^ℓ and Y_i^ℓ are the sets of feature descriptors in the i^{th} sub-regions at scale ℓ of image vector sets X, Y respectively. At each resolution r, we apply the normalized intersection kernel k^r using the square root diagonal normalization method to measure the similarity as follows:

$$\mathsf{k}^r(X_i^\ell, Y_i^\ell) = \frac{\mathtt{I}(X_\ell^\ell(r), Y_i^\ell(r))}{\sqrt{\mathtt{I}(X_i^\ell(r), X_i^\ell(r))\mathtt{I}(Y_i^\ell(r), Y_i^\ell(r))}}, \tag{4.1}$$

where $X_i^\ell(r)$, $Y_i^\ell(r)$ are the sets X_i^ℓ, Y_i^ℓ at the resolution r respectively. Note that the histogram intersection between X and itself is equivalent with its cardinality. Thus, letting $\mathscr{N}_{X_i^\ell(r)}$ and

$\mathscr{N}_{Y_i^\ell(r)}$ be the cardinality of sets $X_i^\ell(r)$ and $Y_i^\ell(r)$, the Equation (4.1) is rewritten as:

$$\mathsf{k}^r(X_i^\ell, Y_i^\ell) = \frac{\mathtt{I}(X_i^\ell(r), Y_i^\ell(r))}{\sqrt{\mathscr{N}_{X_i^\ell(r)} \mathscr{N}_{Y_i^\ell(r)}}}. \tag{4.2}$$

The square root diagonal normalization of the intersection kernel not only satisfies Mercer's conditions (Schölkopf and Smola, 2001), but also penalizes the difference in cardinality between sets as in Equation (4.2).

To obtain the synthetic similarity measurement of the coarse-to-fine model, we define the linear combination over a sequence of local kernels, each term of which is calculated using Equation (4.2) at each resolution. Accordingly, the kernel function k between two sets X_i^ℓ and Y_i^ℓ in the coarse-to-fine model is formulated as:

$$\begin{aligned} \mathsf{k}(X_i^\ell, Y_i^\ell) &= \sum_{r=0}^{R} \theta_r \mathsf{k}^r(X_i^\ell, Y_i^\ell) \\ \text{where} \quad &\sum_{r=0}^{R} \theta_r = 1, \theta_r \geq 0, \forall r = 0, 1, 2, ..., R. \end{aligned}$$

Moreover, when the linear combination of local kernels is integrated with SVM, it can be reformulated as a MKL problem where basic local kernels are defined as Equation (4.2) across the resolutions of the sub-region as:

$$\begin{aligned} \min_{\mathbf{w}_\alpha, w_0, \xi, \theta} \quad & \frac{1}{2}\left(\sum_{\alpha=1}^{N} \theta_\alpha \|\mathbf{w}_\alpha\|_2\right)^2 + C\sum_{i=1}^{m} \xi_i \\ \text{s.t.} \quad & y_i\left(\sum_{\alpha=1}^{N} \theta_\alpha \langle \mathbf{w}_\alpha, \Phi_\alpha(\mathbf{x}_i)\rangle + w_0\right) \geq 1 - \xi_i, \quad \forall i = 1, 2, \ldots, m, \\ & \textstyle\sum_{\alpha=1}^{N} \theta_\alpha = 1, \theta \geq 0, \xi \geq 0, \end{aligned}$$

where $\mathbf{x}_i$ is an image sample, y_i is the category label for $\mathbf{x}_i$, m is the number of training samples, ($\mathbf{w}_\alpha$, w_0, ξ) are parameters of SVM, C is a soft margin parameter defined by users to penalize training errors in SVM, θ is a weight vector for basic local kernels, N is the number of the basic local kernels of the sub-region over the sequence of resolutions, $\theta \geq 0$ means that all elements of vector θ is nonnegative, $\Phi(\mathbf{x})$ is the function that maps the vector $\mathbf{x}$ into the reproducing Hilbert space and $\langle \cdot, \cdot \rangle$ denotes the inner product. MKL solves the parameters of SVM and the weight vector for basic local kernels simultaneously.

These basic local kernels are analogously defined across resolutions of the sub-region. Therefore, the redundant information among them is high. The experiments in Gehler and Nowozin (2009) and especially Kloft et al. (2008) have shown that the uniform combination of basic kernels – a special case of MKL where the weights are uniform – which is an approximation of MKL into traditional nonlinear kernel SVM, is the most efficient for this case in terms of both performance and complexity. Thus, Equation (4.3) with linear combination coefficients obtained from the uniform combination of basic kernels method becomes:

$$\mathsf{k}(X_i^\ell, Y_i^\ell) = \frac{1}{R+1} \sum_{r=0}^{R} \mathsf{k}^r(X_i^\ell, Y_i^\ell). \tag{4.3}$$

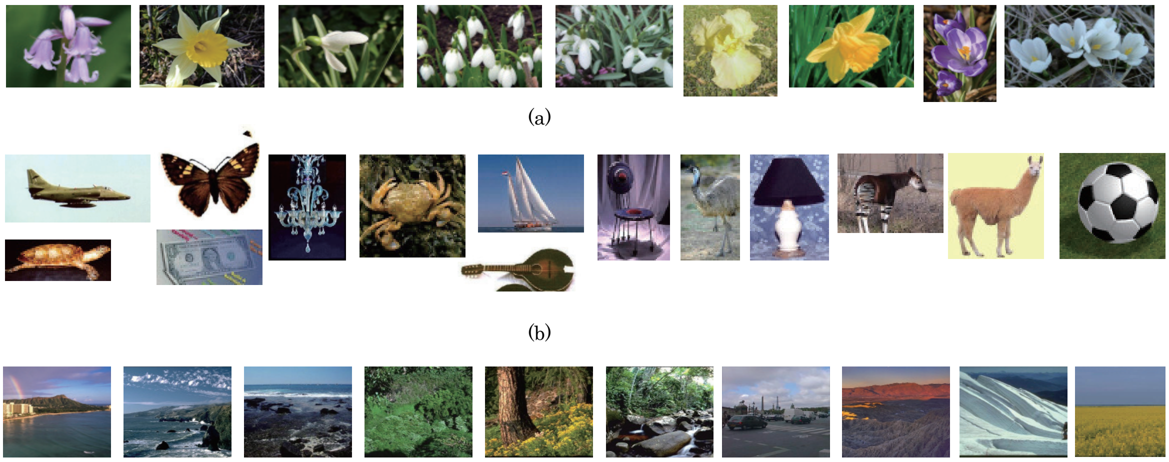
Figure 4.1 illustrates an application of HSMK with $L = 2$ and $R = 2$. HSMK also maps the sub-regions into a sequence of different resolutions for PMK to obtain better measurement of similarity between them. Additionally, the weight vector is achieved from the uniform combination of basic kernels – a special case of MKL where the weights are uniform. Thus, it is more efficient than prior one in PMK. Furthermore, applying the square root diagonal normalization allows it to robustly deal with differences in cardinality that are not considered in PMK. HSMK is formulated based on SPM in the coarse-to-fine model, which is efficient with sets of unordered feature descriptors, even in the presence of differences in cardinality. Mathematically, the formulation of HSMK is as follows:

$$\begin{aligned} \mathsf{k}(X,Y) &= \frac{1}{2^L}\mathsf{k}_0(X,Y) + \sum_{\ell=1}^{L} \frac{1}{2^{L-\ell+1}} \mathsf{k}_\ell(X,Y) \\ \text{with} \quad \mathsf{k}_\ell(X,Y) &= \sum_{i=1}^{2^{2\ell}} \mathsf{k}(X_i^\ell, Y_i^\ell) \\ &= \frac{1}{R+1} \sum_{i=1}^{2^{2\ell}} \sum_{r=0}^{R} \mathsf{k}^r(X_i^\ell, Y_i^\ell). \end{aligned}$$

Briefly, HSMK uses the *kd*-tree algorithm to map each feature descriptor into a discrete visual word, and then the normalized intersection kernel by the square root diagonal method is applied to the histogram of V bins to measure the similarity. We have $\mathcal{N}$ feature descriptors in the n-dimension space, and the *kd*-tree algorithm costs $O(\log V)$ steps to map feature descriptors. Therefore, the complexity of HSMK is $O(nM \log V)$ with $M = \max(\mathcal{N}_X, \mathcal{N}_Y)$. We note that the complexity of the optimal matching approach (Kondor and Jebara, 2003) is $O(nM^3)$.

4.4 Experimental results

Most recent approaches use local invariant features as an effective means of representing images, because they can well describe and match instances of objects or scenes under a wide variety of viewpoints, illuminations, or even background clutter. Among them, SIFT



(a)

(b)

(c)

Fig. 4.2 Example images of various datasets used in experiments: (a) Oxford flower dataset. (b) Caltech 101 dataset. (c) Scene Categorization dataset.

(Lowe, 2004) is one of the most robust and efficient features. To achieve better discriminative ability, we use the dense SIFT by operating a SIFT descriptor of 16×16 patches computed over each pixel of an image instead of key points (Lowe, 2004) or a grid of points (Lazebnik et al., 2006). In addition, to improve robustness, we convert images into gray scale ones before computing the dense SIFT. Dense features have the capability of capturing uniform regions such as sky, water or grass where key points usually do not exist. Moreover, the combination of dense features and the coarse-to-fine model allows images to be represented more exactly since feature descriptors achieves more neighbor information across many levels in resolution. We performed unsupervised k-means clustering on a random subset of SIFT descriptors to build visual words. Typically, we used two different dictionary sizes V in our experiment: $V = 400$ and $V = 800$.

We conducted experiments for two types of image categorization: object categorization and scene categorization. For object categorization, we used the Oxford Flower dataset (Nilsback and Zisserman, 2006). To show the efficiency and scalability of our proposed kernel, we also used the large scale object datasets such as CALTECH-101 (Fei-Fei et al., 2004) and CALTECH-256 (Griffin et al., 2007). For scene categorization, we evaluated the proposed kernel on the MIT scene (Oliva and Torralba, 2001) and UIUC scene (Lazebnik et al., 2006) datasets. Example images of datasets used in experiments are shown in Fig. 4.2.

Table 4.1 Classification rate (%) with a single feature comparison on Oxford Flower dataset (with NN that denotes the nearest neighbor algorithm)

Method	Accuracy (%)
HSV (NN) (Nilsback and Zisserman, 2008)	43.0
SIFT-Internal (NN) (Nilsback and Zisserman, 2008)	55.1
SIFT-Boundary (NN) (Nilsback and Zisserman, 2008)	32.0
HOG (NN) (Nilsback and Zisserman, 2008)	49.6
HSV (SVM) (Gehler and Nowozin, 2009)	61.3
SIFT-Internal (SVM) (Gehler and Nowozin, 2009)	70.6
SIFT-Boundary (SVM) (Gehler and Nowozin, 2009)	59.4
HOG (SVM) (Gehler and Nowozin, 2009)	58.5
SIFT (MSL) (Wang and Wang, 2010)	65.3
Dense SIFT (HSMK)	**72.9**

Table 4.2 Classification rate (%) comparison between SPMK and HSMK on Oxford Flower dataset

Kernel	$V = 400$	$V = 800$
SPMK	68.09	69.12
HSMK	**71.76**	**72.94**

4.4.1 Object categorization

To access the efficiency of the proposed HSMK for object categorization, we compared the classification accuracy with that of conventional SPM in Oxford Flowers dataset and Caltech datasets.

Oxford Flowers dataset

This dataset contains 17 classes of common flowers in the United Kingdom, collected by Nilsback and Zisserman (2006). Each class has 80 images with large scale, pose and light variations. Moreover, intra-class flowers such as irises, fritillaries and pansies are also widely diverse in their colors and shapes. There are some cases of close similarity between flowers of different classes such as that between dandelion and Colts'Foot. In our experiments, we followed the set-up of Gehler and Nowozin (2009), randomly choosing 40 samples from each class for training and using the rest for testing. Note that we did not use a validation set as in (Nilsback and Zisserman, 2006, 2008) for choosing the optimal parameters.

Table 4.1 shows that our proposed kernel achieved a state-of-the-art results using single image feature. We use various classifier for comparison results such as Nearest Neighbour (NN), SVM, and Multi-scale learning (MSL) (Wang and Wang, 2010) method. The HSMK

Table 4.3 Classification rate (%) comparison on CALTECH-101 dataset

# training samples (each class)	5	10	15	20	25	30
Grauman and Darrell (2005)	34.8	44	50.0	53.5	55.5	58.2
Wang and Wang (2010)	-	-	61.4	-	-	-
Lazebnik et al. (2006)	-	-	56.4	-	-	64.6
Yang et al. (2009)	-	-	67.0	-	-	73.2
Boiman et al. (2008)	56.9	-	72.8	-	-	79.1
Gehler and Nowozin (2009) (MKL)	42.1	55.1	62.3	67.1	70.5	73.7
Gehler and Nowozin (2009) (LP-β)	54.2	65.0	70.4	73.6	75.7	77.8
Gehler and Nowozin (2009) (LP-B)	46.5	59.7	66.7	71.1	73.8	77.2
Our method (HSMK)	50.5	62.2	69.0	72.3	74.4	77.3

Table 4.4 Classification rate (%) comparison between SPMK and HSMK on CALTECH-101 dataset

# training samples (each class)	5	10	15	20	25	30
SPMK ($V = 400$)	48.18	58.86	65.34	69.35	71.95	73.46
HSMK(V=400)	**50.68**	**61.97**	**67.91**	**71.35**	**73.92**	**75.59**
SPMK ($V = 800$)	48.11	59.70	66.84	69.98	72.62	75.13
HSMK(V=800)	**50.48**	**62.17**	**68.95**	**72.32**	**74.36**	**77.33**

using dense SIFT gives 72.9% that outperformed not only SIFT-Internal (Nilsback and Zisserman, 2008) of 70.6%, the best feature for this dataset computed on a segmented image, but also the same feature on SPMK with the optimal weights by MSL of 65.3%. Table 4.2 shows that the performance of our HSMK outperformed that of conventional SPMK when using a single SIFT feature.

Caltech datasets

To show the efficiency and robustness of HSMK, we also evaluated its performance on large scale object datasets, i.e., the CALTECH-101 and CALTECH-256 datasets. These datasets feature high intra-class variability, poses, and viewpoints. On CALTECH-101,

Table 4.5 Classification rate (%) comparison on CALTECH-256 dataset

# training samples (each class)	15	30
Griffin et al. (2007) (SPMK)	28.4	34.2
Yang et al. (2009) (ScSPM)	27.7	34.0
Gehler and Nowozin (2009) (MKL)	30.6	35.6
SPMK	25.3	31.3
Our method (HSMK)	27.2	34.1

we carried out experiments with 5, 10, 15, 20, 25, and 30 training samples for each class, including the background class, and used up to 50 samples per class for testing. Table 4.3 compares the classification rate results of our approach with other ones. As shown, our approach obtained the comparable result with that of state-of-the-art approaches even using only a single feature while others used many types of features and complex learning algorithms such as MKL and linear programming boosting (LP-B) (Gehler and Nowozin, 2009). Table 4.4 shows that the result of HSMK outperformed that of SPMK in this case as well. It should be noted that when the experiment was conducted without the background class, our approach achieved a classification rate of 78.4% for 30 training samples. This shows that our approach is efficient in spite of its simplicity.

On the UIUC Scene dataset, we followed the experiment setup described in (Lazebnik et al., 2006). We randomly chose 100 training samples per class and the rest were used for testing. As shown in Table 4.7, the result of our proposed kernel also outperformed that of SPMK (Lazebnik et al., 2006) as well as SPM based on sparse coding (Yang et al., 2009) for this dataset.

On CALTECH-256, we performed experiments with HSMK using 15 and 30 training samples per class, including the clutter class, and 25 samples of each class for testing. We also re-implemented SPMK (Griffin et al., 2007) but used our dense SIFT to enable a fair comparation of SPMK and HSMK. As shown in Table 4.5, the HSMK classification rate was about 3 percent higher than that of SPMK.

4.4.2 Scene categorization

We also performed experiments using HSMK on the MIT Scene (8 classes) and UIUC Scene (15 classes) dataset. In these datasets, we set $V = 400$ as the dictionary size. On the MIT Scene dataset, we randomly chose 100 samples per class for training and 100 other samples per class for testing. As shown in Table 4.6, the classification rate for HSMK was 2.5 percent higher than that of SPMK. Our approach also outperformed other local feature approaches (Johnson, 2008) as well as local feature combinations (Johnson, 2008) by more than 10 percent, and was better than the global feature GIST (Oliva and Torralba, 2001), an efficient feature in scene categorization.

4.5 HSMK with Sparse Coding

As in section 4.4, hierarchical spatial matching kernel is proved as an efficient and effective kernel. However, it is still a nonlinear kernel due to the fact that an intersection kernel is used

Table 4.6 Classification rate (%) comparison on MIT Scene (8 classes) dataset

Method	Accuracy (%)
GIST (Oliva and Torralba, 2001)	83.7
Local features (Johnson, 2008)	77.2
Dense SIFT (SPMK)	85.8
Dense SIFT (HSMK)	**88.3**

Table 4.7 Classification rate (%) comparison on UIUC Scene (15 classes) dataset

Method	Accuracy (%)
Lazebnik et al. (2006) (SPMK)	81.4
Yang et al. (2009) (ScSPM)	80.3
SPMK	79.9
Our method (HSMK)	**82.2**

as a basic kernel to build it. So, it is difficult to apply HSMK to deal with large-scale datasets effectively in term of time consuming. To help HSMK overcome this issue, we exploit a sparse coding approach and max spooling strategy to make data linear instead of using a vector quantization method by k-means. Therefore, we can replace the intersection kernel by a linear kernel as a basic kernel to construct HSMK based on the linear property of this kind of data. It is worthwhile noting that the performance will become much worse when we apply the linear kernel as a basic kernel in HSMK in case of utilizing the vector quantization method.

We conducted the same configuration as in section 4.4 for the experiments of HSMK with sparse coding, but we set dictionary sizes $V = 800$. For sparse coding, we apply ℓ_1 regularization instead of other regularization constraint like ℓ_0 or ℓ_2 norm, because ℓ_1 norm regularization is known as the best choice for image categorization problem (Raina et al., 2007; Yang et al., 2009). After that, we follow an efficient algorithm proposed by Lee et al. (2006) to achieve the solution for sparse coding problem.

Table 4.8 and table 4.9 show the comparison of applying between vector quantization and sparse coding with HSMK on Oxford Flower and CALTECH-101 dataset respectively. They are proved that sparse coding is an efficient method to make HSMK linear, it can keep the performance of HSMK as in case of utilizing an intersection kernel as basic kernels. The performance of HSMK with linear kernel just decreases about 1.62% and 1.07% on Oxford Flower and CALTECH-101 dataset respectively in comparison with HSMK with intersection kernel while it is about 10% in case of using vector quantization.

We can explore from the results in table 4.8 and table 4.9 that the performance of HSMK with intersection kernel is better than one of HSMK with linear kernel for both vector

Table 4.8 Classification rate (%) comparison between HSMK with vector quantization and HSMK with sparse coding on Oxford Flower dataset

HSMK	Vector Quantization	Sparse Coding
Linear kernel	63.53	73.38
Intersection kernel	**72.94**	**75.00**

Table 4.9 Classification rate (%) comparison between HSMK with vector quantization and HSMK with sparse coding on CALTECH-101 dataset with 30 training samples

HSMK	Vector Quantization	Sparse Coding
Linear kernel	65.28	78.93
Intersection kernel	**77.33**	**80.60**

quantization and sparse coding in Oxford Flower and CALTECH-101 dataset. Additionally, it is different with the case of spatial pyramid kernel which in (Yang et al., 2009) whose authors claimed that SPK with linear kernel was also better than SPK with nonlinear kernel when we used sparse coding.

Note that the results of sparse coding for HSMK with intersection kernel in table 4.8 and table 4.9 are state of the art results for Oxford Flower and CALTECH-101 dataset respectively. Therefore, HSMK with sparse coding is an effective approach for image categorization and especially the performance of HSMK with linear kernel can achieve comparative results with HSMK with nonlinear kernel in case of utilizing sparse coding.

4.6 Summary

In this chapter, we proposed an efficient and robust kernel that we call the hierarchical spatial matching kernel (HSMK). It uses a coarse-to-fine model for sub-regions to improve spatial pyramid matching kernel (SPMK) and thus obtains more neighbor information through a sequence of different resolutions. In addition, the kernel efficiently and robustly handles sets of unordered features as SPMK and pyramid matching kernel as well as sets having different cardinalities.

Combining the proposed kernel with a dense feature approach was found to be sufficiently effective and efficient. It enabled us to obtain at least comparable results with those by existing methods for many kinds of datasets. Moreover, our approach is simple since it is based on only a single feature with nonlinear support vector machines, in contrast to other more complicated recent approaches based on multiple kernel learning or feature combinations. In addition, it is more effective when we combine HSMK with sparse coding.

In most well-known datasets of object and scene categorization, the proposed kernel was also found to outperform SPMK which is an important component such as a basic kernel in multiple kernel learning. This means that we can replace SPMK with HSMK to improve the performance of frameworks based on basic kernels.

Chapter 5

Conclusion

5.1 Summary of the Contributions

In this thesis, we contribute to metric learning for histograms in both supervised and unsupervised settings. We leverage the Aitchison geometry in the simplex to do so. Additionally, we propose the hierarchical spatial matching kernel for images using the bag of features approach for representation. We use a coarse to fine model, realized by multi-resolutions to achieve better similarity measure.

In chapter 2, we propose the generalized Aitchison embeddings which map histograms from the simplex into a suitable Euclidean space. These maps not only preserve the geometric properties of the simplex, but they also make the following analysis easier since we can rely on enormous Euclidean tools such as Euclidean distance, quadratic forms and ellipses. Instead of using a few predefined maps such as those proposed by Aitchison (1982), we provide algorithms to learn such maps from labeled data by adapting the large margin nearest neighbor framework, which is one of the most popular Mahalanobis metric learning frameworks. We illustrate that our proposal outperforms alternative approaches on a variety of contexts from image, handwritten digit, flower to text classification. Furthermore, we give an empirical analysis about the behaviour of our proposed algorithms such as convergence speed and parameter sensitivity.

In chapter 3, we propose to learn a Riemannian metric on the simplex from *unlabeled* histogram data. We follow the maximizing inverse volume framework proposed by Lebanon (2006). This framework provides us a way to estimate such the metric within a parametric family. The metrics we consider on the simplex are pull-back metrics from the Euclidean metric on a positive sphere through a composition transformation of the Hellinger mapping and Aitchison transformation (Aitchison, 1982) within the simplex. This approach can be also interpreted to learn the Aitchison transformation within the simplex for the Fisher's information metric. We propose a new algorithmic approach to maximize inverse volumes of a given dataset of points under the metric in a general case by using sampling and contrastive divergences. Empirical evidence shows that the metric obtained under our proposal outperforms alternative approaches for clustering and classification tasks on many benchmark datasets.

In chapter 4, we propose a kernel for images represented by the bag of features in computer vision. We build upon the spatial pyramid matching kernel (Lazebnik et al., 2006) which plays an important role to embed spatial information into the bag of features representation for images. Lazebnik et al. use a sequence of grids to partition an image into fixed sub-regions. However, the authors still applies bag of features for each subregion which is limited in its capacity to measure similarity between sets of unordered features. In the chapter, we leverage a coarse to fine model for each subregion to improve optimal matching approximation. Empirical evidence shows that our proposed hierarchical spatial matching kernel outperforms spatial pyramid matching kernel on several benchmark datasets in image categorization.

5.2 Future work

Similarity learning is one of the most popular problems in machine learning. However, when input data are a bag of features (histogram), the traditional metric learning approaches based on the Mahalanobis distance or a linear mapping do not work well in practice. Additionally, the bag of features are widely used in many research fields such as computer vision, natural language processing and speech processing. Therefore, metric learning for histograms is essential.

Some traditional distances for histograms have been recently used to replace the Mahalanobis distance in metric learning for histogram data such as the χ^2 distance, the Fisher's information metric and the transportation distance (also known as the earth mover's distance). Therefore, one of further research directions is to explore other distances or divergences for histograms such as the Hellinger distance, the Kullback-Leibler divergence, the Jensen-Shannon divergence (a symmetrized version of the Kullback-Leibler divergence) or general divergences such as the f-divergence or the Bregman divergence. Additionally, traditional frameworks for the Mahalanobis distance may be not appropriate for those distances and divergences for histograms. For example, metric learning based on Mahalanobis distance can be formulated as a convex problem. However, when we replace Mahalanobis distance by those traditional distances for histograms, the optimization problem usually turns into non-convex. Seeking an appropriate framework for those histogram distances and divergences is also necessary.

It is easy to collect unlabelled data, but it takes much time to build a labeled dataset. Additionally, most of metric learning approaches use the supervised setting. There are only a few unsupervised metric learning approaches for unlabeled data. So, metric learning for histograms in the unsupervised setting are indeed needed more explorations.

We can also explore metric learning for other structured data types such as trees, graphs, strings, time series, molecules or materials. Since each data type has its own geometry and traditional metric learning may not work well in practice, geometry-aware metric learning for those structured data is also a promising direction for future work.

References

J. Aitchison. The statistical analysis of compositional data. *Journal of the Royal Statistical Society*, 44:139–177, 1982.

J. Aitchison. *The statistical analysis of compositional data*. Chapman and Hall, Ltd., 1986.

J. Aitchison. A concise guide to compositional data analysis. In *Compositional Data Analysis Workshop*, 2003.

J. Aitchison and I. J. Lauder. Kernel density estimation for compositional data. *Applied statistics*, pages 129–137, 1985.

J. Aitchison and S. M. Shen. Logistic-normal distributions: Some properties and uses. *Biometrika*, pages 261–272, 1980.

A. Aizawa. An information-theoretic perspective of tf–idf measures. *Information Processing and Management*, 39(1):45–65, 2003.

R. Baeza-Yates and B. Ribeiro-Neto. *Modern information retrieval*, volume 463. ACM press New York, 1999.

H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *European Conference on Computer Vision*, pages 404–417, 2006.

Y. Bengio. Speeding up stochastic gradient descent. In *Workshop on Efficient Machine Learning, Neural Information Processing Systems*, 2007.

D. Blei and J. Lafferty. Correlated topic models. In *Advances in Neural Information Processing Systems*, pages 147–154, 2006.

D. Blei and J. Lafferty. *Topic models*. Text Mining: Classification, Clustering, and Applications, 2009.

D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

O. Boiman, E. Shechtman, and M. Irani. In defense of nearest-neighbor based image classification. In *Computer Vision and Pattern Recognition*, 2008.

S. Boughhorbel, J. P. Tarel, and F. Fleuret. Non-mercer kernels for support vector machine object recognition. *In British Machine Vision Conference*, 2004.

C. Burge, A. M. Campbell, and S. Karlin. Over-and under-representation of short oligonucleotides in dna sequences. *National Academy of Sciences*, 89(4):1358–1362, 1992.

W. M. Campbell and F. S. Richardson. Discriminative keyword selection using support vector machines. In *Advances in Neural Information Processing Systems*, 2007.

W. M. Campbell, J. P. Campbell, D. A. Reynolds, D. A. Jones, and T. R. Leek. Phonetic speaker recognition with support vector machines. In *Advances in Neural Information Processing Systems*, 2003.

M. S. Charikar. Similarity estimation techniques from rounding algorithms. In *ACM symposium on Theory of computing*, pages 380–388. ACM, 2002.

R. Chatpatanasiri, T. Korsrilabutr, P. Tangchanachaianan, and B. Kijsirikul. On kernelizing mahalanobis distance learning algorithms. *Arxiv preprint*, 804, 2008.

S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 539–546. IEEE, 2005.

M. Cuturi and D. Avis. Ground metric learning. *arXiv preprint arXiv:1110.2306*, 2011.

M. Cuturi and D. Avis. Ground metric learning. *The Journal of Machine Learning Research*, 15(1):533–564, 2014.

D. L. Davies and D. W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 224–227, 1979.

J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *International Conference on Machine Learning*, pages 209–216, 2007.

G. Doddington. Speaker recognition based on idiolectal differences between speakers. In *Eurospeech*, pages 2521–2524, 2001.

J. J. Egozcue, V. Pawlowsky-Glahn, G. Mateu-Figueras, and C. Barcelo-Vidal. Isometric logratio transformations for compositional data analysis. *Mathematical Geology*, 35(3): 279–300, 2003.

S. Erhan, T. Marzolf, and L. Cohen. Amino-acid neighborhood relationships in proteins. breakdown of amino-acid sequences into overlapping doublets, triplets and quadruplets. *International Journal of Bio-Medical Computing*, 11(1):67–75, 1980.

L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. In *Workshop on Generative-Model Based Vision*, 2004.

A. Frome, Y. Singer, F. Sha, and J. Malik. Learning globally-consistent local distance functions for shape-based image retrieval and classification. In *International Conference on Computer Vision*, pages 1–8. IEEE, 2007.

P. Gehler and S. Nowozin. On feature combination for multiclass object classification. In *International Conference on Computer Vision*, pages 221 –228, 2009.

A. Globerson and S. T. Roweis. Metric learning by collapsing classes. In *Advances in Neural Information Processing Systems*, pages 451–458, 2005.

J. Goldberger, S. T. Roweis, G. E. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In *Advances in Neural Information Processing Systems*, 2004.

K. Grauman and T. Darrell. The pyramid match kernel: discriminative classification with sets of image features. In *International Conference on Computer Vision*, volume 2, pages 1458 –1465 Vol. 2, 2005.

G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.

G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.

M. Hoffman, F. R. Bach, and D. M. Blei. Online learning for latent dirichlet allocation. In *Advances in neural information processing systems*, pages 856–864, 2010.

J. Hu, J. Lu, and Y.-P. Tan. Discriminative deep metric learning for face verification in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1875–1882. IEEE, 2014.

P. Jain, B. Kulis, I. S. Dhillon, and K. Grauman. Online metric learning and fast similarity search. In *Advances in neural information processing systems*, pages 761–768, 2009.

P. Jain, B. Kulis, J. V. Davis, and I. S. Dhillon. Metric and kernel learning using a linear transformation. *The Journal of Machine Learning Research*, 13(1):519–547, 2012.

T. Joachims. *Learning to Classify Text Using Support Vector Machines: Methods, Theory and Algorithms*. Springer, 2002.

M. Johnson. *Semantic Segmentation and Image Search*. PhD thesis, University of Cambridge, 2008.

B. Julesz. Textons, the elements of texture perception, and their interactions. *Nature*, 1981.

D. Kedem, S. Tyree, K. Q. Weinberger, F. Sha, and G. Lanckriet. Nonlinear metric learning. In *Advances in Neural Information Processing Systems*, pages 2582–2590, 2012.

J. Kivinen and M. K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–63, 1997.

M. Kloft, U. Brefeld, P. Laskov, and S. Sonnenburg. Non-sparse multiple kernel learning. In *NIPS Workshop on Kernel Learning: Automatic Selection of Kernels*, 2008.

R. I. Kondor and T. Jebara. A kernel between sets of vectors. In *International Conference on Machine Learning*, pages 361–368, 2003.

G. Kunapuli and J. Shavlik. Mirror descent for metric learning: A unified approach. In *European on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pages 859–874. Springer, 2012.

J. T. Kwok and I. W. Tsang. Learning with idealized kernels. In *International Conference on Machine Learning*, pages 400–407, 2003.

S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition*, volume 2, pages 2169–2178, 2006.

T. Le and M. Cuturi. Generalized aitchison embeddings for histograms. In *Asian Conference on Machine Learning*, pages 293–308, 2013.

T. Le and M. Cuturi. Adaptive euclidean maps for histograms: generalized aitchison embeddings. *Machine Learning*, pages 1–19, 2014.

T. Le, Y. Kang, A. Sugimoto, S. Tran, and T. Nguyen. Hierarchical spatial matching kernel for image categorization. In *International Conference on Image Analysis and Recognition*, pages 141–151, 2011.

G. Lebanon. Learning riemannian metrics. In *Uncertainty in Artificial Intelligence*, pages 362–369. Morgan Kaufmann Publishers Inc., 2002.

G. Lebanon. *Riemannian Geometry and Statistical Machine Learning*. PhD thesis, Carnegie Mellon University, 2005.

G. Lebanon. Metric learning for text documents. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):497–508, 2006.

H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems*, pages 801–808, 2006.

C. S. Leslie, E. Eskin, and W. S. Noble. The spectrum kernel: A string kernel for svm protein classification. In *Pacific Symposium on Biocomputing*, volume 7, pages 566–575, 2002.

D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004.

D. Lim, G. Lanckriet, and B. McFee. Robust structural metric learning. In *International Conference on Machine Learning*, pages 615–623, 2013.

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

R. E. Madsen, D. Kauchak, and C. Elkan. Modeling word burstiness using the dirichlet distribution. In *International Conference on Machine Learning*, pages 545–552, 2005.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *International Conference on Machine Learning*, pages 689–696, 2009. ISBN 978-1-60558-516-1.

S. Maji, A. C. Berg, and J. Malik. Classification using intersection kernel support vector machines is efficient. In *Computer Vision and Pattern Recognition*, pages 1 –8, 2008.

C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

B. McFee and G. R. Lanckriet. Metric learning to rank. In *International Conference on Machine Learning*, pages 775–782, 2010.

F. Moosmann, B. Triggs, and F. Jurie. Randomized clustering forests for building fast and discriminative visual vocabularies. In *NIPS Workshop on Kernel Learning: Automatic Selection of Kernels*, 2008.

Y. Nesterov. A method of solving a convex programming problem with convergence rate o (1/k2). In *Soviet Mathematics Doklady*, volume 27, pages 372–376, 1983.

Y. Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer, 2004.

D. Newman, P. Smyth, M. Welling, and A. U. Asuncion. Distributed inference for latent dirichlet allocation. In *Advances in neural information processing systems*, pages 1081–1088, 2007.

M. E. Nilsback and A. Zisserman. A visual vocabulary for flower classification. In *Computer Vision and Pattern Recognition*, volume 2, pages 1447–1454, 2006.

M. E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*, 2008.

Y.-K. Noh, B.-T. Zhang, and D. D. Lee. Generative local metric learning for nearest neighbor classification. In *Advances in Neural Information Processing Systems*, pages 1822–1830, 2010.

M. Norouzi, D. M. Blei, and R. R. Salakhutdinov. Hamming distance metric learning. In *Advances in neural information processing systems*, pages 1061–1069, 2012.

B. O'Donoghue and E. Candès. Adaptive restart for accelerated gradient schemes. *Foundations of Computational Mathematics*, 2013.

A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42:145–175, 2001. ISSN 0920-5691.

S. Parameswaran and K. Q. Weinberger. Large margin multi-task metric learning. In *Advances in neural information processing systems*, pages 1867–1875, 2010.

F. Perronnin, J. Sánchez, and Y. Liu. Large-scale image categorization with explicit data embedding. In *Computer Vision and Pattern Recogition*, pages 2297–2304, 2010.

R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng. Self-taught learning: transfer learning from unlabeled data. In *International Conference on Machine Learning*, pages 759–766. ACM, 2007.

J. D. Rennie, L. Shih, J. Teevan, and D. Karger. Tackling the poor assumptions of naive bayes text classifiers. In *International Conference on Machine Learning*, volume 3, pages 616–623, 2003.

B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: a database and web-based tool for image annotation. *International Journal of Computer Vision*, 77(1-3): 157–173, 2008.

G. Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information and Computer*. Addison-Wesley, 1989.

G. Salton and M. J. McGill. *Introduction to Moderm Information Retrieval*. McGraw-Hill, 1983.

B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001. ISBN 0262194759.

M. Schultz and T. Joachims. Learning a distance metric from relative comparisons. In *Advances in Neural Information Processing Systems*, volume 16, page 41, 2003.

S. Shalev-Shwartz, Y. Singer, and A. Y. Ng. Online and batch learning of pseudo-metrics. In *International Conference on Machine Learning*, page 94, 2004.

J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *International Conference on Computer Vision*, 2003.

Y. W. Teh, D. Newman, and M. Welling. A collapsed variational bayesian inference algorithm for latent dirichlet allocation. In *Advances in neural information processing systems*, pages 1353–1360, 2006.

L. Torresani and K. Lee. Large margin component analysis. In *Advances in Neural Information Processing Systems*, pages 1385–1392, 2006.

S. Trivedi, D. McAllester, and G. Shakhnarovich. Discriminative metric learning by neighborhood gerrymandering. In *Advances in Neural Information Processing Systems*, pages 3392–3400, 2014.

A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. *IEEE Pattern Analysis and Machine Intelligence*, 34(3):480–492, 2012.

F. Wang and L. J. Guibas. Supervised earth mover's distance learning and its computer vision applications. In *European Conference on Computer Vision*, pages 442–455. Springer, 2012.

H. Y. Wang, H. Zha, and H. Qin. Dirichlet aggregation: unsupervised learning towards an optimal metric for proportional data. In *International conference on Machine learning*, pages 959–966. ACM, 2007.

J. Wang, A. Kalousis, and A. Woznica. Parametric local metric learning for nearest neighbor classification. In *Advances in Neural Information Processing Systems*, pages 1601–1609, 2012.

S. C. Wang and Y. C. F. Wang. A multi-scale learning framework for visual categorization. In *Asian Conference on Computer Vision*, 2010.

K. Q. Weinberger and L. K. Saul. Fast solvers and efficient implementations for distance metric learning. In *International Conference on Machine Learning*, pages 1160–1167, 2008.

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

K. Q. Weinberger, J. Blitzer, and L. Saul. Distance metric learning for large margin nearest neighbor classification. In *Advances in Neural Information Processing Systems*, pages 1473–1480, 2006.

P. Xie and E. P. Xing. Multi-modal distance metric learning. In *International joint conference on artificial intelligence*, pages 1806–1812. AAAI Press, 2013.

E. P. Xing, A. Y. Ng, M. I. Jordan, and S. J. Russell. Distance metric learning with application to clustering with side-information. In *Advances in Neural Information Processing Systems*, pages 1473–1480, 2002.

J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition*, pages 1794 –1801, 2009.

L. Yang, R. Jin, R. Sukthankar, and F. Jurie. Unifying discriminative visual codebook generation with classifier training for object category recognition. In *Computer Vision and Pattern Recognition*, volume 0, pages 1–8, Los Alamitos, CA, USA, 2008. ISBN 978-1-4244-2242-5.

P. Yang, K. Huang, and C.-L. Liu. Geometry preserving multi-task metric learning. *Machine learning*, 92(1):133–175, 2013.

Y. Zhang and D.-Y. Yeung. Transfer metric learning by learning task relationships. In *ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1199–1208. ACM, 2010.

Publications by the Author

Journal Articles

(J1) Tam Le, Marco Cuturi. Adaptive Euclidean maps for histograms: generalized Aitchison embeddings. *Machine Learning Journal (MLJ)*, 99(2):169–187, 2014.

(J2) Tam Le, Yousun Kang, Akihiro Sugimoto. Image categorization using hierarchical spatial matching kernel. *Journal of the Institute of Image Electronics Engineers of Japan (IIEEJ)*, 42(2):214–221, 2013.

Reviewed Conference Papers

(P1) Tam Le, Marco Cuturi. Unsupervised Riemannian metric learning for histograms using Aitchison transformations. In *International Conference on Machine Learning (ICML)*, pages 2002–2011, 2015.

(P2) Tam Le, Marco Cuturi. Generalized Aitchison embeddings for histograms. In *Asian Conference on Machine Learning (ACML)*, pages 293–308, 2013.

Related Talk

(T1) Tam Le. Adaptive Euclidean maps for histograms: generalized Aitchison embeddings. In *Lear, INRIA Rhone Alpes*, Grenoble, France, 2014.

Other Activities

(A1) Tam Le, Yousun Kang, Akihiro Sugimoto, Son Tran, Thuc Nguyen. Hierarchical spatial matching kernel for image categorization. In *International Conference on Image Processing and Recognition (ICIAR)*, pages 141–151, 2011.

(A2) Tam Le, Son Tran, Seiichi Mita, Thuc Nguyen. Real time traffic sign detection using color and shape-based features. In *Asian Conference on Intelligent Information and Database Systems (ACIIDS)*, pages 268–278, 2010.

Appendix A

Proof for the Propostions in Chaper 3

Proof for the Proposition 1

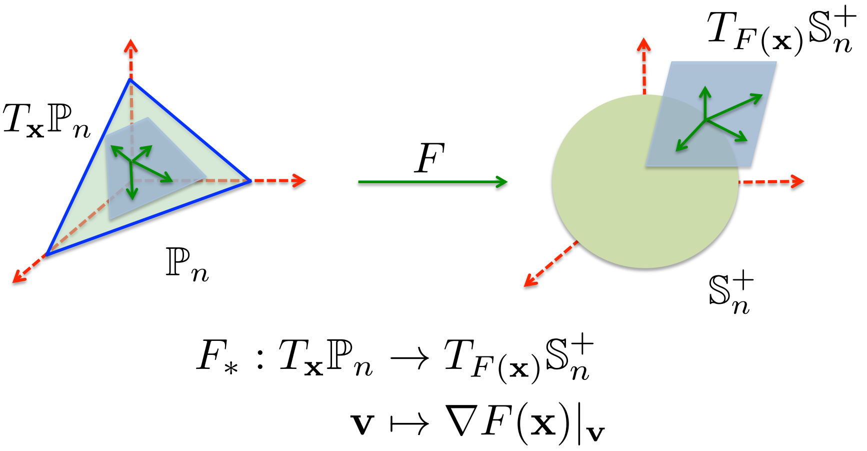

$T_{F(\mathbf{x})}\mathbb{S}_n^+$
$T_{\mathbf{x}}\mathbb{P}_n$
F
$\mathbb{P}_n$
$\mathbb{S}_n^+$
$F_* : T_{\mathbf{x}}\mathbb{P}_n \to T_{F(\mathbf{x})}\mathbb{S}_n^+$
$\mathbf{v} \mapsto \nabla F(\mathbf{x})|_{\mathbf{v}}$

Fig. A.1 The push forward map a tangent vector $\mathbf{v}$ on the tangent space of the simplex $T_{\mathbf{x}}\mathbb{P}_n$ into a tangent vector $F_*\mathbf{v}$ on the tangent space of the positive sphere $T_{F(\mathbf{x})}\mathbb{S}_n^+$.

The j^{th} component of the tangent vector of the positive sphere, mapped by a push-forward map F_* on a tangent vector $\mathbf{v} \in T_{\mathbf{x}}\mathbb{P}_n$ as illustrated in Figure A.1

$$[F_*\mathbf{v}]_j = \frac{\mathrm{d}}{\mathrm{d}t}\sqrt{\frac{\left(\mathbf{x}_j + t\mathbf{v}_j\right)^{\alpha_j}\lambda_j}{\sum_{i=1}^{n+1}\left(\mathbf{x}_i + t\mathbf{v}_i\right)^{\alpha_i}\lambda_i}}\Bigg|_{t=0},$$

For simplify, let denote:

$$f_i(t) = \left(\mathbf{x}_i + t\mathbf{v}_i\right)^{\alpha_i}\lambda_i,$$

and

$$h_i(t) = \frac{\mathrm{d}f_i(t)}{\mathrm{d}t} = (\mathbf{x}_i + t\mathbf{v}_i)^{\alpha_i - 1}\lambda_i\alpha_i\mathbf{v}_i.$$

So, we have:

$$[F_*\mathbf{v}]_j = \left.\frac{h_j(t)\sum_{i=1}^{n+1} f_i(t) - f_j(t)\sum_{i=1}^{n+1} h_i(t)}{2\sqrt{\frac{f_j(t)}{\sum_{i=1}^{n+1} f_i(t)}}\left(\sum_{i=1}^{n+1} f_i(t)\right)^2}\right|_{t=0}$$

Since at $t = 0$,

$$f_i(0) = \mathbf{x}_i^{\alpha_i}\lambda_i$$

and

$$h_i(0) = \mathbf{x}_i^{\alpha_i - 1}\lambda_i\alpha_i\mathbf{v}_i,$$

we have

$$[F_*\mathbf{v}]_j = \frac{1}{2}\mathbf{x}_j^{\frac{\alpha_j}{2}-1}\alpha_j\mathbf{v}_j\lambda_j^{\frac{1}{2}}\left(\sum_{\ell=1}^{n+1}\mathbf{x}_\ell^{\alpha_\ell}\lambda_\ell\right)^{-\frac{1}{2}} - \frac{1}{2}\mathbf{x}_j^{\frac{\alpha_j}{2}}\lambda_j^{\frac{1}{2}}\frac{\sum_{\ell=1}^{n+1}\mathbf{x}_\ell^{\alpha_\ell - 1}\alpha_\ell\mathbf{v}_\ell\lambda_\ell}{\left(\sum_{\ell=1}^{n+1}\mathbf{x}_\ell^{\alpha_\ell}\lambda_\ell\right)^{\frac{3}{2}}}.$$

Let apply $\mathbf{v} = \partial_i$, $1 \le i \le n$, the basis of the tangent space of the simplex $T_{\mathbf{x}}\mathbb{P}_n$

$$\begin{aligned}[F_*\partial_i]_j = &\frac{1}{2}\mathbf{x}_j^{\frac{\alpha_j}{2}-1}\alpha_j\lambda_j^{\frac{1}{2}}\left(\sum_{\ell=1}^{n+1}\mathbf{x}_\ell^{\alpha_\ell}\lambda_\ell\right)^{-\frac{1}{2}}\left(\delta_{j,i} - \delta_{j,n+1}\right)\\ &-\frac{1}{2}\mathbf{x}_j^{\frac{\alpha_j}{2}}\lambda_j^{\frac{1}{2}}\frac{\mathbf{x}_i^{\alpha_i - 1}\alpha_i\lambda_i - \mathbf{x}_{n+1}^{\alpha_{n+1}-1}\alpha_{n+1}\lambda_{n+1}}{\left(\sum_{\ell=1}^{n+1}\mathbf{x}_\ell^{\alpha_\ell}\lambda_\ell\right)^{\frac{3}{2}}},\end{aligned}$$

where $\delta_{j,i} = 1$ if $j = i$ and $\delta_{j,i} = 0$, otherwise.

Hence, we have

$$T = U(I - \beta\eta^T)D.$$

Moreover, the metric on the positive sphere $\mathbb{S}_n^+$ is Euclidean. So, we have

$$J(\partial_i, \partial_j) = \left\langle F_*\partial_i, F_*\partial_j\right\rangle.$$

Consequently, we have the Gram matrix:

$$\mathscr{G} = TT^T = U(I - \beta\eta^T)D^2(I - \beta\eta^T)^T U^T.$$

So, we have the proof for the Proposition 1.

Proof for the Proposition 2

Let consider matrix $\beta\eta^T \in \mathbb{R}^{(n+1)\times(n+1)}$, and vector $\mathbf{v}$ such that

$$\eta^T\mathbf{v} = 0.$$

So, $\mathbf{v}$ is a eigenvector of $\beta\eta^T$ with eigenvalue 0. There are n independent vectors $\{\mathbf{v_i}\}_{1\le i\le n}$ such that

$$\eta^T\mathbf{v_i} = 0.$$

Moreover,

$$\operatorname{tr}\left(\beta\eta^T\right) = \sum_{i=1}^{n+1} \beta_i\eta_i = 1,$$

or sum of the eigenvalues of $\beta\eta^T$ is 1.

So, the last of $(n+1)$ eigenvalues is 1. On the other hand,

$$\left(\beta\eta^T\right)\beta = \beta\left(\eta^T\beta\right) = \beta,$$

or β is a eigenvector of $\beta\eta^T$ with eigenvalue 1.

In summary, we have $\{(\mathbf{v_i},0)_{1\le i\le n},(\beta,1)\}$ are eigenvectors and corresponding eigenvalues of $\beta\eta^T$. Let V be a matrix in $\mathbb{R}^{(n+1)\times(n+1)}$ whose columns are $\{\mathbf{v_1},\mathbf{v_2},\cdots,\mathbf{v_n},\beta\}$.

So, we may express V as follow:

$$V = \begin{bmatrix} -\frac{\mathbf{x}_2\alpha_1}{\alpha_2\mathbf{x}_1} & \cdots & -\frac{\mathbf{x}_{n+1}\alpha_1}{\alpha_{n+1}\mathbf{x}_1} & \mathbf{x}_1^{\alpha_1-1}\alpha_1\lambda_1 \\ 1 & \cdots & 0 & \mathbf{x}_1^{\alpha_2-1}\alpha_2\lambda_2 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & \mathbf{x}_{n+1}^{\alpha_{n+1}-1}\alpha_{n+1}\lambda_{n+1} \end{bmatrix}.$$

Let Λ is a diagonal matrix in $\mathbb{R}^{(n+1)\times(n+1)}$ where $\Lambda_{ii} = 0$, for all $1 \le i \le n$, and $\Lambda_{(n+1)(n+1)} = 1$. We have

$$\beta\eta^T = V\Lambda V^{-1}.$$

Consequently, we have

$$I - \beta\eta^T = V(I-\Lambda)V^{-1}.$$

Since

$$I - \Lambda = \operatorname{diag}(1,1,\cdots,1,0)$$

and

$$I - \Lambda = (I - \Lambda)^2,$$

we may express

$$I - \beta \eta^T = \widetilde{V}\widetilde{V^{-1}},$$

where $\widetilde{V} \in \mathbb{R}^{(n+1)\times n}$ is the matrix V whose last column is removed, and $\widetilde{V^{-1}} \in \mathbb{R}^{n\times(n+1)}$ is the matrix V^{-1} whose last row is removed.

Thus, we can express the Gram matrix $\mathscr{G}$ as follow:

$$\begin{aligned}\mathscr{G} &= U\widetilde{V}\widetilde{V^{-1}}D^2(\widetilde{V}\widetilde{V^{-1}})^T U^T \\ &= (U\widetilde{V})(\widetilde{V^{-1}}D^2\widetilde{V^{-1}}^T)(U\widetilde{V})^T\end{aligned}$$

We also note that $U\widetilde{V}$ and $\widetilde{V^{-1}}D^2\widetilde{V^{-1}}^T$ are matrices in $\mathbb{R}^{n\times n}$.

So, we have

$$\det\mathscr{G} = \det{}^2(U\widetilde{V})\det(\widetilde{V^{-1}}D^2\widetilde{V^{-1}}^T).$$

Compute det$(U\widetilde{V})$: Since, we have

$$\begin{aligned}U\widetilde{V} &= \begin{pmatrix} -\frac{\mathbf{x}_2\alpha_1}{\alpha_2\mathbf{x}_1} & \cdots & -\frac{\mathbf{x}_n\alpha_1}{\alpha_n\mathbf{x}_1} & -\frac{\mathbf{x}_{n+1}\alpha_1}{\alpha_{n+1}\mathbf{x}_1} - 1 \\ 1 & \cdots & 0 & -1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & -1 \end{pmatrix} \\ &= \frac{\alpha_1}{\mathbf{x}_1}\begin{pmatrix} -\sum\limits_{i=1}^{n+1}\frac{\mathbf{x}_i}{\alpha_i} & -\frac{\mathbf{x}_3}{\alpha_3} & \cdots & -\frac{\mathbf{x}_n}{\alpha_n} & -\frac{\mathbf{x}_{n+1}}{\alpha_{n+1}} - \frac{\mathbf{x}_1}{\alpha_1} \\ 0 & 0 & \cdots & 0 & -1 \\ 0 & 1 & \cdots & 0 & -1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & -1 \end{pmatrix}.\end{aligned}$$

Therefore,

$$\det(U\widetilde{V}) = (-1)^n\frac{\alpha_1}{\mathbf{x}_1}\sum_{i=1}^{n+1}\frac{\mathbf{x}_i}{\alpha_i}.$$

Compute det$(\widetilde{V^{-1}}D^2\widetilde{V^{-1}}^T)$**:** Let consider a $(n+1)\times(n+1)$ matrix

$$W=\begin{pmatrix} -\frac{\mathbf{r}_2}{\mathbf{r}_1} & \cdots & -\frac{\mathbf{r}_{n+1}}{\mathbf{r}_1} & \mathbf{c}_1 \\ 1 & \cdots & 0 & \mathbf{c}_2 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & \mathbf{c}_{n+1} \end{pmatrix}$$

We have its inverse:

$$W^{-1}=\frac{1}{\langle\mathbf{r},\mathbf{c}\rangle}\begin{pmatrix} -\mathbf{r}_1\mathbf{c}_2 & \langle\mathbf{r},\mathbf{c}\rangle-\mathbf{r}_2\mathbf{c}_2 & -\mathbf{r}_3\mathbf{c}_2 & \cdots & -\mathbf{r}_{n+1}\mathbf{c}_2 \\ -\mathbf{r}_1\mathbf{c}_3 & -\mathbf{r}_2\mathbf{c}_3 & \langle\mathbf{r},\mathbf{c}\rangle-\mathbf{r}_2\mathbf{c}_2 & \cdots & -\mathbf{r}_{n+1}\mathbf{c}_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\mathbf{r}_1\mathbf{c}_{n+1} & -\mathbf{r}_2\mathbf{c}_{n+1} & -\mathbf{r}_2\mathbf{c}_{n+1} & \cdots & \langle\mathbf{r},\mathbf{c}\rangle-\mathbf{r}_{n+1}\mathbf{c}_{n+1} \\ \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \cdots & \mathbf{r}_{n+1} \end{pmatrix},$$

where vector $\mathbf{r}=(\mathbf{r}_1,\mathbf{r}_2,\cdots,\mathbf{r}_{n+1})$ and vector $\mathbf{c}=(\mathbf{c}_1,\mathbf{c}_2,\cdots,\mathbf{c}_{n+1})$ are in $\mathbb{R}^{n+1}$.

Now, we apply for the matrix V where $\mathbf{r}_i=\frac{\mathbf{x}_i}{\alpha_i}$ and $\mathbf{c}_i=\mathbf{x}_i^{\alpha_i-1}\alpha_i\lambda_i$, for all $1\le i\le(n+1)$, and remove the last row to form $\widetilde{V^{-1}}$.

For simplicity, we denote a diagonal matrix $P\in\mathbb{R}^{n\times n}$ where $P_{ii}=\mathbf{x}_{i+1}^{\alpha_{i+1}-1}\alpha_{i+1}\lambda_{i+1}$, for all $1\le i\le n$ and matrix $Q\in\mathbb{R}^{n\times(n+1)}$ as follow:

$$Q=\begin{pmatrix} -\frac{\mathbf{x}_1}{\alpha_1} & \sum\limits_{\substack{1\le i\le n+1\\ i\ne 2}}\frac{\mathbf{x}_i^{\alpha_i}\lambda_i}{\mathbf{x}_2^{\alpha_2-1}\alpha_2\lambda_2} & -\frac{\mathbf{x}_3}{\alpha_3} & \cdots & -\frac{\mathbf{x}_{n+1}}{\alpha_{n+1}} \\ -\frac{\mathbf{x}_1}{\alpha_1} & -\frac{\mathbf{x}_2}{\alpha_2} & \sum\limits_{\substack{1\le i\le n+1\\ i\ne 3}}\frac{\mathbf{x}_i^{\alpha_i}\lambda_i}{\mathbf{x}_3^{\alpha_3-1}\alpha_3\lambda_3} & \cdots & -\frac{\mathbf{x}_{n+1}}{\alpha_{n+1}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\frac{\mathbf{x}_1}{\alpha_1} & -\frac{\mathbf{x}_2}{\alpha_2} & -\frac{\mathbf{x}_3}{\alpha_3} & \cdots & \sum\limits_{i=1}^{n}\frac{\mathbf{x}_i^{\alpha_i}\lambda_i}{\mathbf{x}_{n+1}^{\alpha_{n+1}-1}\alpha_{n+1}\lambda_{n+1}} \end{pmatrix}.$$

So, we have

$$\widetilde{V^{-1}}=\frac{1}{\sum\limits_{i=1}^{n+1}\mathbf{x}_i^{\alpha_i}\lambda_i}PQ.$$

Then, we can compute

$$\widetilde{V^{-1}}D=\frac{1}{2\left(\sum\limits_{i=1}^{n+1}\mathbf{x}_i^{\alpha_i}\lambda_i\right)^{\frac{3}{2}}}P\begin{pmatrix} -\sqrt{\mathbf{x}_1^{\alpha_1}\lambda_1} & \sum\limits_{\substack{1\le i\le n+1\\ i\neq 2}}\frac{\mathbf{x}_i^{\alpha_i}\lambda_i}{\sqrt{\mathbf{x}_2^{\alpha_2}\lambda_2}} & -\sqrt{\mathbf{x}_3^{\alpha_3}\lambda_3} & \cdots & -\sqrt{\mathbf{x}_{n+1}^{\alpha_{n+1}}\lambda_{n+1}} \\ -\sqrt{\mathbf{x}_1^{\alpha_1}\lambda_1} & \sqrt{\mathbf{x}_2^{\alpha_2}\lambda_2} & \sum\limits_{\substack{1\le i\le n+1\\ i\neq 3}}\frac{\mathbf{x}_i^{\alpha_i}\lambda_i}{\mathbf{x}_3^{\alpha_3-1}\alpha_3\lambda_3} & \cdots & -\sqrt{\mathbf{x}_{n+1}^{\alpha_{n+1}}\lambda_{n+1}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\sqrt{\mathbf{x}_1^{\alpha_1}\lambda_1} & \sqrt{\mathbf{x}_2^{\alpha_2}\lambda_2} & -\sqrt{\mathbf{x}_3^{\alpha_3}\lambda_3} & \cdots & \sum\limits_{i=1}^{n}\frac{\mathbf{x}_i^{\alpha_i}\lambda_i}{\sqrt{\mathbf{x}_{n+1}^{\alpha_{n+1}}\lambda_{n+1}}} \end{pmatrix}.$$

Consequently, we have:

$$\widetilde{V^{-1}}D^2\widetilde{V^{-1}}^T=\frac{1}{4\left(\sum\limits_{i=1}^{n+1}\mathbf{x}_i^{\alpha_i}\lambda_i\right)^2}P(\widehat{Q}-\mathbf{1}_{n\times n})P,$$

where $\widehat{Q}$ is a diagonal matrix in $\mathbb{R}^{n\times n}$, $\widehat{Q}_{ii}=\sum\limits_{j=1}^{n+1}\frac{\mathbf{x}_j^{\alpha_j}\lambda_j}{\mathbf{x}_{i+1}^{\alpha_{i+1}}\lambda_{i+1}}$ and $\mathbf{1}_{n\times n}$ is a matrix of 1 in $\mathbb{R}^{n\times n}$. Moreover,

$$\det(\widehat{Q}-\mathbf{1}_{n\times n})=\prod_{i=1}^{n}Q_{ii}-\sum_{i=1}^{n}\prod_{j\neq i}Q_{jj},$$

following Lemma 2 of Lebanon (2005).

Hence, we have:

$$\det(\widehat{Q}-\mathbf{1}_{n\times n})=\frac{\left(\sum\limits_{i=1}^{n+1}\mathbf{x}_i^{\alpha_i}\lambda_i\right)^{n-1}}{\prod\limits_{j=1}^{n+1}\mathbf{x}_j^{\alpha_j}\lambda_j}\left(\mathbf{x}_1^{\alpha_1}\lambda_1\right)^2.$$

Consequently, we have

$$\det\left(\widetilde{V^{-1}}D^2\widetilde{V^{-1}}^T\right)=\frac{1}{4^n\left(\sum\limits_{i=1}^{n+1}\mathbf{x}_i^{\alpha_i}\lambda_i\right)^{2n}}\det^2(P)\det(\widehat{Q}-\mathbf{1}_{n\times n}).$$

Since

$$\det(P)=\prod_{j=2}^{n+1}\mathbf{x}_j^{\alpha_j-1}\alpha_j\lambda_j,$$

we have:

$$\det\left(\widetilde{V^{-1}}D^2\widetilde{V^{-1}}^T\right)=\frac{\left(\frac{\mathbf{x}_1}{\alpha_1}\right)^2\left(\prod\limits_{i=1}^{n+1}\mathbf{x}_i^{\alpha_i-2}\alpha_i^2\lambda_i\right)}{4^n\left(\sum\limits_{i=1}^{n+1}\mathbf{x}_i^{\alpha_i}\lambda_i\right)^{n+1}}.$$

Hence, we have:

$$\det\mathscr{G}=\frac{\left(\sum\limits_{i=1}^{n+1}\frac{\mathbf{x}_i}{\alpha_i}\right)^2\left(\prod\limits_{i=1}^{n+1}\mathbf{x}_i^{\alpha_i-2}\alpha_i^2\lambda_i\right)}{4^n\left(\sum\limits_{i=1}^{n+1}\mathbf{x}_i^{\alpha_i}\lambda_i\right)^{n+1}}.$$

So, we have the proof for the Proposition 2.

Proof for the Proposition 3

The partial derivative of the objective function $\mathscr{F}$ with respect to λ is:

$$\frac{\partial\mathscr{F}}{\partial\lambda}=\frac{1}{m}\sum_{i=1}^{m}\frac{\partial\log\mathrm{dvol}g^{-1}(\mathbf{x_i})}{\partial\lambda}-E\left(\frac{\partial\log\mathrm{dvol}g^{-1}(\mathbf{x})}{\partial\lambda}\right)_{p(\mathbf{x})}$$

Since we have

$$\begin{aligned}
\frac{\partial\log\int_{\mathbb{P}_n}\mathrm{dvol}J^{-1}(\mathbf{x})\mathrm{d}\mathbf{x}}{\partial\lambda} &= \frac{1}{\int_{\mathbb{P}_n}\mathrm{dvol}J^{-1}(\mathbf{x})\mathrm{d}\mathbf{x}}\frac{\partial\int_{\mathbb{P}_n}\mathrm{dvol}J^{-1}(\mathbf{x})\mathrm{d}\mathbf{x}}{\partial\lambda}\\
&= \frac{1}{\int_{\mathbb{P}_n}\mathrm{dvol}J^{-1}(\mathbf{x})\mathrm{d}\mathbf{x}}\int_{\mathbb{P}_n}\frac{\partial\mathrm{dvol}J^{-1}(\mathbf{x})}{\partial\lambda}\mathrm{d}\mathbf{x}\\
&= \frac{1}{\int_{\mathbb{P}_n}\mathrm{dvol}J^{-1}(\mathbf{x})\mathrm{d}\mathbf{x}}\int_{\mathbb{P}_n}\mathrm{dvol}J^{-1}(\mathbf{x})\frac{\partial\log\mathrm{dvol}g^{-1}(\mathbf{x})}{\partial\lambda}\mathrm{d}\mathbf{x}\\
&= \int_{\mathbb{P}_n}\frac{\mathrm{dvol}J^{-1}(\mathbf{x})}{\int_{\mathbb{P}_n}\mathrm{dvol}J^{-1}(\mathbf{z})\mathrm{d}\mathbf{z}}\frac{\partial\log\mathrm{dvol}g^{-1}(\mathbf{x})}{\partial\lambda}\mathrm{d}\mathbf{x}\\
&= E\left(\frac{\partial\log\mathrm{dvol}J^{-1}(\mathbf{x})}{\partial\lambda}\right)_{p(\mathbf{x})}.
\end{aligned}$$

and

$$\frac{\partial\log\mathrm{dvol}J^{-1}(\mathbf{x})}{\partial\lambda}=\frac{n+1}{2\sum\limits_{i=1}^{n+1}\mathbf{x}_i^{\alpha_i}\lambda_i}\left[\mathbf{x}_j^{\alpha_j}\right]_{1\leq j\leq n+1}.$$

So, we have the proof for $\frac{\partial\mathscr{F}}{\partial\lambda}$.

Similarly, we also obtain the proof for $\frac{\partial\mathscr{F}}{\partial\alpha}$.

The content of Chapter 2 was published in Journal of Machine Learning Workshop and Conference Proceedings Vol. 29: Asian Conference on Machine Learning (ACML), pp. 293–308, 2013 and Machine Learning Journal (MLJ), pp. 169–187, 2014. *Springer and the Machine Learning Journal, volume 99, 2015, pp. 169–187, Adaptive Euclidean Maps for Histograms: Generalized Aitchison Embeddings, Tam Le and Marco Cuturi, original copyright notice is given to the publication in which the material was originally published, by adding; with kind permission from Springer Science and Business Media.*

The content of Chapter 3 was published in Journal of Machine Learning Research Workshop and Conference Proceedings Vol. 37: Proceedings of the 32^{nd} International Conference on Machine Learning (ICML), pp. 2002–2011, 2015.

The content of Chapter 4 was published in Journal of the Institute of Image Electronics Engineers of Japan (IIEEJ), Vol. 42, No. 2, pp. 214–221, 2013.