Anisotropic Mean Shift Clustering using Distance Metric Learning

Student Name: Gullal Singh Cheema

IIIT-D-MTech-CS-14-008
August 8, 2016

Indraprastha Institute of Information Technology
New Delhi

Thesis Committee
Dr. Chetan Arora
Dr. K. K. Biswas
Dr. Saket Anand

Submitted in partial fulfillment of the requirements
for the Degree of M.Tech. in Computer Science

Keywords: clustering, metric learning, classification, kernel learning, Mahalanobis distance

Certificate

This is to certify that the thesis titled **"Anisotropic Mean Shift Clustering using Distance Metric Learning"** submitted by **Gullal Singh Cheema** for the partial fulfillment of the requirements for the degree of *Master of Technology* in *Computer Science & Engineering* is a record of the bonafide work carried out by him under my supervision. This work has not been submitted anywhere else for the reward of any other degree.

Dr. Saket Anand
Indraprastha Institute of Information Technology, New Delhi

Abstract

Mean shift is a non-parametric mode seeking procedure widely used in many computer vision problems. Mean shift clustering in particular is a well studied and established algorithm, which has many merits over the classic k-means clustering algorithm. These algorithms repeatedly calculate distance between data points to compute mean shift vector and cluster mean respectively using some distance function. In most of the cases, Euclidean distance function is used which weighs every dimension equally in the input space and thus often fails to capture the semantics of the data. To alleviate this problem, a general form of distance metric based on Mahalanobis distance is used that can be learned using the training data.

Distance metric learning has received a lot of attention in recent years and has proven to be very successful in various problem domains. By learning a Mahalanobis distance metric, the input space is transformed such that, similar points get closer to each other and dissimilar points move further apart. A lot of research has been done on learning a global metric and integrating it with k-means algorithm, but there have been very few efforts of integrating metric learning with mean shift clustering.

This work focuses on developing a unified framework for improving mean shift clustering by using global and local metric learning. We use a recently proposed Sparse Compositional Metric Learning (SCML) framework and integrate it with mean shift clustering to investigate the affect of using local metrics over a global metric. We also perform kernelization in the proposed framework that can handle datasets with non-linear decision boundaries. To establish the effectiveness of our approach, we performed experiments on 6 datasets of varying difficulty.

Acknowledgments

This work would not have been possible without the guidance and support of several individuals who in one way or the other assisted in the preparation and completion of this work. I would like to express my sincere gratitude and respect towards Dr. Saket Anand, for being my supervisor and under whose able and encouraging guidance, I was able to complete my work in a better and exhaustive manner. He has inspired me to do good work, and has supported me all along the way.

I would like to thank Dr. Pankaj Jalote for making IIIT-Delhi such a wonderful place to work. I owe particular thanks to my parents and siblings who have always encouraged me in my pursuits, without their help all this would have not been possible.

I would like to thank my friends, batch mates and PhD students (especially Ankita Shukla, PhD student with Dr. Saket Anand) for providing such a good environment for fruitful discussions, many of which helped me immensely.

Lastly, I would also like to thank my parents for motivating from time to time throughout the course of my thesis which enabled me to pursue my research in an efficient and structured manner.

Gullal Singh Cheema
M.Tech (CSE)

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

The problem of finding relevant clusters has been a focus of considerable research in machine learning and pattern recognition community. The aim of clustering is to find structure/pattern in the data so as to group similar points into the same cluster and dissimilar points into different clusters. Although, clustering is an unsupervised learning problem, many algorithms have been proposed in the past that use some labeled data or pairwise/triplet constraints on some examples to improve clustering performance. Specifically, the supervision is usually added either by modifying the clustering algorithm and enforcing the constraints during the clustering process [11, 27], or by learning a distance metric that satisfies the imposed constraints and then running the clustering algorithm which uses the learned metric [2, 10, 30]. Some of the works [4, 20, 33] have integrated both by performing distance metric learning with each clustering iteration to capture arbitrary shaped clusters. In this work, we are interested in the former approach, where we independently learn global and local metrics and then use these metrics in mean shift clustering.

Mean shift is a general non-parametric mode seeking procedure, which was first introduced by Fukunga and Hostetler [13] with its application to clustering and data filtering. Unlike k-means, mean shift is a robust procedure which does not make any explicit assumptions on the number, shape or any kind of random initialization of clusters. The procedure works iteratively by maximizing the kernel density estimate (KDE) to locate modes in the underlying probability distribution. Due to these properties, mean shift has been an active topic of research over the years and has been commonly used in computer vision tasks, e.g., image segmentation and smoothing [7] and real-time object tracking [6, 8, 31]. While k-means has been integrated with metric learning in many previous works [2, 4, 10, 20, 30, 33], mean shift has hardly been explored in such a setting. However, there have been some efforts of introducing weak supervision [26] and semi-supervision [1] in the kernel induced feature space.

A distance metric (function) plays a pivotal role in many classification and clustering algorithms. In most of the clustering algorithms, a distance metric is required to compute distance

between data points and determines the type (compact/loose) of clusters discovered by the algorithm. In general, distance metric learning techniques use training samples to learn a distance function that is semantically consistent with the data. The default Euclidean distance weighs each dimension equally in the input space and is often inadequate to capture the semantics of the data. To incorporate the correlation between features, the well studied Mahalanobis distance metric is often used which is characterized by a symmetric positive semidefinite (PSD) matrix. Much of its popularity is because it can be easily extended to non-linear spaces via kernelization [10, 18, 33] and leads to simpler formulations.

Learning a global Mahalanobis metric is equivalent to learning a linear transformation of the input space where the given constraints are satisfied. One of the earliest works by Xing *et al.* [30] learned a PSD Mahalanobis metric using similarity (dissimilarity) constraints by a combination of gradient descent and iterative projections for k-means clustering. Although, using a single Mahalanobis metric has shown to be quite effective, it is often unable to capture the complexity of the task in heterogeneous data. To overcome this limitation Bilenko *et al.* [4] learned global as well as multiple cluster metrics for k-means clustering to capture arbitrary shaped clusters and improve clustering performance.

More recently, local metric learning has been explored in the context of nearest neighbor classification and has been shown to improve classification accuracies over global metric learning. This line of work is motivated by the fact that locally, simple linear metrics perform well [24]. Most of the existing methods [16, 23, 28, 29] learn full rank matrices, which do not scale well with high dimensional data and are thus prone to overfitting. In this work, we use the recently proposed sparse compositional metric learning (SCML) [25] by Shi *et al.* which uses low-rank locally discriminative metrics extracted from the training data and integrates these into meaningful global and local metrics. The method learns a sparse combination of low-rank metrics and thus requires learning much less parameters for both global and local metrics than the existing methods. We derive the mean shift clustering formulation for both global and local metrics. We also show the benefit of using local metrics over global metrics with mean shift clustering through empirical experiments on various datasets.

1.2 Motivation and Aim

More often than not we encounter such tasks in which a linear global metric is not able to accurately capture the complexity (modality/complex boundaries) of the problem. In such cases multiple (more than one) metrics or instance-specific metrics can be learned to capture complex data patterns. The idea is that by allowing the metric to vary across feature space, semantic information can be better captured by local metrics than by a global metric.

The objective function used in SCML-Global 2.12 and SCML-Local 2.15 promotes local structure in the data by imposing large margin nearest neighbor (LMNN) [29] constraints. Also, the local metrics correspond to a distance measure that locally adapts to the underlying

structure of the data. This ability gives more power to an algorithm that uses local metrics over a global metric. SCML learns these local metrics by using locally discriminative low-rank metrics extracted from local regions in the data, and this discriminatory information can additionally help in detecting better separated clusters in the data.

With respect to mean shift clustering, the benefit of using point-wise bandwidths over a single global bandwidth has been very well studied in [9,14]. Using a global metric on top of that can be visualized as locating modes in the KDE in the transformed space. On the other hand, the use of local metrics fits very well with point-wise bandwidths in the kernel density estimate. In this case, local metrics capture the local structure w.r.t the point and the point-wise bandwidth acts as the scaling parameter for that point. Thus, the idea of using local metrics with point-wise bandwidths is more appropriate than using a global metric.

Therefore, the aim of this work is to determine how local metrics can increase the performance of mean shift clustering over the use of a global metric.

1.3 Organization of Thesis

This thesis is organized as follows:

- Chapter 2 provides a brief introduction to mean shift clustering and metric learning which are required for developing the algorithm proposed in chapter 4.
- Chapter 3 underlines the related work that has been done on combining metric learning and clustering into a unified framework.
- Chapter 4 proposes the framework for integrating global and local metric learning with mean shift clustering. It also proposes the kernelization of proposed framework.
- Chapter 5 applies the approaches proposed in 4 on various datasets to show the effectiveness of the proposed approach. It also provides the details of preprocessing the data and evaluation metrics used for evaluating the framework.
- Chapter 6 concludes the thesis by summarizing the contribution of work and also gives possible extensions or improvements that can be further explored.

Chapter 2

Preliminaries

This chapter briefly discusses the basic theoretical concepts of mean shift clustering and metric learning which are required to understand our work. We specifically discuss about Sparse Compositional Metric learning (SCML) which is central to our work.

2.1 Adaptive Mean Shift Clustering

We first discuss about kernel density estimate in section 2.1.1 and then mean shift vector computation in section 2.1.2.

2.1.1 Kernel Density Estimate

Kernel density estimate (KDE) is a nonparametric technique to estimate the underlying probability density function (p.d.f) from the training data. The idea behind KDE is simple, more the data in a region, larger is the density function. These dense regions correspond to the regions where the probability of finding a mode is very high.

Given n data points $\mathbf{x}_i$, $i = 1, \ldots, n$ in a d-dimensional space $\mathbb{R}^d$ and the corresponding point-wise isotropic bandwidth matrices $h_i \mathbf{I}_{d \times d}$, sample point density estimator is given by

$$f_K(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{h_i^d} \, k\left(\left\| \frac{\mathbf{x} - \mathbf{x}_i}{h_i} \right\|^2 \right) \tag{2.1}$$

which is the average of differently scaled kernels centered at each data point. Here, $k(x)$ is the kernel *profile* of the radially symmetric kernel K, which is non-negative and integrates to one. With kernel profile $k(x)$, for $x \geq 0$, kernel K satisfies

$$K(\mathbf{x}) = c_{k,d} k\left(\|\mathbf{x}\|^2 \right) \tag{2.2}$$

where $c_{k,d}$ is the normalization constant which makes kernel $K(\mathbf{x})$ integrate to one.

Comaniciu, Ramesh and Meer in [9] showed that using point-wise bandwidths gives better performance and estimate of the underlying distribution than using fixed bandwidth. The mean shift procedure also converges faster when using point-wise bandwidths.

The number of modes (peaks) in the KDE depends on the bandwidths and the kernel function. If the bandwidths are too small, the KDE will be noisy with more number of modes, whereas if bandwidths are too large, the KDE will be over-smoothed with less number of modes. In such a case, reasonable bandwidths can be learned from the training data before the clustering process. The simplest way to estimate h_i's is by nearest neighbor approach. If $\mathbf{x}_{i,k}$ is the k-nearest neighbor of point $\mathbf{x}_i$, then

$$h_i = \|\mathbf{x}_i - \mathbf{x}_{i,k}\|_2 \tag{2.3}$$

where k should be large enough such that density increases within the support of the differently scaled kernels.

A normal (gaussian) kernel function that gives more weight to closer (to $\mathbf{x}$) points than farther points gives a smooth KDE and reasonable number of modes. On the other hand, Epanechnikov kernel which gives zero weight to points which are farther than a threshold, results in a less smoother KDE and more number of modes for the same bandwidths.

2.1.2 Mean Shift Procedure

The intuition behind mean shift computation is that the local mean centered at $\mathbf{x}$ shifts toward the region in which the majority of points reside. The point at which the shift is negligible is called the stationary point. In terms of density estimate, these modes or stationary points are located at the point where gradient of the density estimate is zero, i.e. $\nabla f(\mathbf{x}) = 0$. This gives an elegant procedure to estimate the modes in the data without explicitly estimating the KDE. Therefore, by taking the gradient of Eq. 2.1, we get the following equation that is satisfied by stationary points

$$\frac{2}{n}\sum_{i=1}^{n} \frac{1}{h_i^{d+2}}(\mathbf{x} - \mathbf{x}_i)\; k'\left(\left\|\frac{\mathbf{x} - \mathbf{x}_i}{h_i}\right\|^2\right) = 0 \tag{2.4}$$

Using $g(x) = -k'(x)$ assuming that derivative of $k(x)$ exists and introducing it into the Eq. 2.4, the stationary point can be iteratively reached by following the mean shift procedure starting with $\mathbf{y}_1 = \mathbf{x}$

$$\mathbf{y}_{j+1} = \frac{\sum_{i=1}^{n} \frac{\mathbf{x}_i}{h_i^{d+2}}\; g\left(\left\|\frac{\mathbf{y}_j - \mathbf{x}_i}{h_i}\right\|^2\right)}{\sum_{i=1}^{n} \frac{1}{h_i^{d+2}}\; g\left(\left\|\frac{\mathbf{y}_j - \mathbf{x}_i}{h_i}\right\|^2\right)} \qquad j = 1, 2 \ldots \tag{2.5}$$

where $\{\mathbf{y}_j\}_{j=1,2\ldots}$ is the sequence of locations or the path leading to a stationary point of the estimated p.d.f. The mean shift procedure stops when

$$\|\delta\mathbf{y}\| = \|\mathbf{y}_{j+1} - \mathbf{y}_j\|_2 \approx 0 \tag{2.6}$$

where $\mathbf{y}_j$ and $\mathbf{y}_{j+1}$ is the current mean and shifted mean respectively, and $\delta\mathbf{y}$ is the mean shift vector.

The mean shift procedure is an adaptive gradient ascent method where the learning rate changes w.r.t the neighborhood. In regions of low-density, $\delta\mathbf{y}$ is large and in regions of high density or when near local maxima, $\delta\mathbf{y}$ is small. The mean shift vector always points towards the direction of maximum increase in the density. This was first mentioned by Fukunga *et al.* in [13] and the convergence guarantee that a point converges to the local mode of the density was later shown by Comaniciu *et al.* in [7].

For clustering the points, the mean shift procedure is run on every point and the points converging to the same mode are clustered together.

2.2 Metric Learning

Metric learning plays an important role in capturing the underlying structure (correlation between features) for measuring distances between data points, which standard Euclidean metric often fails to capture. Therefore, a lot of research has been done on metric learning, which is summarized in this recent survey by Bellet, Habrard and Sebban [3]. In this section, we briefly discuss about what a metric is, its properties and describe Mahalanobis distance metric.

A metric is real valued function mapping $d : \mathcal{X} \times \mathcal{X} \rightarrow \Re_0^+$ over a m-dimensional vector space $\mathcal{X}$, if $\forall \mathbf{x}, \mathbf{x}', \mathbf{x}'' \in \mathcal{X}$, it satisfies the following properties:

- $d(\mathbf{x}, \mathbf{x}') + d(\mathbf{x}', \mathbf{x}'') \geq d(\mathbf{x}, \mathbf{x}'')$ (triangle Inequality)
- $d(\mathbf{x}, \mathbf{x}') \geq 0$ (non-negativity)
- $d(\mathbf{x}, \mathbf{x}') = d(\mathbf{x}', \mathbf{x})$ (symmetric)
- $d(\mathbf{x}, \mathbf{x}) = 0$ (identity)

Although, there are many methods (linear and non-linear) to learn a metric as summarized in [3], learning a globally linear Mahalanobis distance metric is the most common one. Mahalanobis distance refers to generalized squared distance parameterized by a $m \times m$ symmetric positive semi-definite (PSD) matrix $\mathbf{M}$, which satisfies the above mentioned properties.

$$d_{\mathbf{M}}(\mathbf{x}, \mathbf{x}') = \sqrt{(\mathbf{x} - \mathbf{x}')^T \mathbf{M} (\mathbf{x} - \mathbf{x}')} \qquad \mathbf{M} \in \mathbb{S}_+^m \tag{2.7}$$

$\mathbf{M}$ can also be expressed as $\mathbf{M} = \mathbf{L}^T\mathbf{L}$, where $\mathbf{L} \in \mathbb{R}^{r \times m}$ is transformation matrix and r ($\leq m$) is the rank of $\mathbf{M}$. In such a case, the Eq. 2.7 becomes the Euclidean distance in the transformed space after linear projection of the data by $\mathbf{L}$.

$$d_{\mathbf{M}}(\mathbf{x}, \mathbf{x}') = \sqrt{(\mathbf{L}\mathbf{x} - \mathbf{L}\mathbf{x}')^T (\mathbf{L}\mathbf{x} - \mathbf{L}\mathbf{x}')} \tag{2.8}$$

If $\mathbf{M}$ is low rank ($r \leq m$), then it induces a linear projection of data into a lower (r) dimension space. This produces a compact representation of data and leads to cheaper distance computations.

The task of Mahalanobis distance metric learning is to learn a PSD matrix $\mathbf{M}$ that satisfies the distance constraints imposed by the training data. There are a number of challenges when learning $\mathbf{M}$: 1) it scales poorly with dimensionality of the data, as it requires learning $\mathcal{O}(m^2)$ parameters, 2) optimization involves projecting $\mathbf{M}$ onto the PSD cone to maintain $\mathbf{M} \in \mathbb{S}_+^m$ which is an expensive $\mathcal{O}(m^3)$ procedure, and 3) to preferably learn a low-rank $\mathbf{M}$, especially in case of high dimensional data. Unfortunately, learning a low-rank $\mathbf{M}$ is NP hard and most methods [19, 22] fix the rank and use special regularizer functions to learn the matrix.

Most of the metric learning methods use pairwise or triplet distance constraints to learn a PSD matrix $\mathbf{M}$.

- Pairwise distance constraints are in the form of similarity (must-link) and dissimilarity (cannot-link) pairs of points, where the learned metric ensures that similar points move closer whereas dissimilar points move further apart.

- Triplet constraints are in the form of relative comparisons ($\mathbf{x}$ is closer to $\mathbf{x}'$ than $\mathbf{x}''$) where the learned metric tries to maximize the difference between two relative distances. In this form, the distance between $\mathbf{x}$ and $\mathbf{x}'$ can increase after transforming the space with the learned metric.

The optimization function 2.9 usually has a loss function ($\mathcal{L}$) which incurs a penalty on violation of distance constraints and a regularization function ($\mathcal{R}$) on the parameters of the learned metric $\mathbf{M}$ to limit the risk of overfitting.

$$f(\mathbf{M}, C, \beta) = \min_{\mathbf{M}} \mathcal{L}(\mathbf{M}, C) + \beta \mathcal{R}(\mathbf{M}) \tag{2.9}$$

2.2.1 Sparse Compositional Metric Learning

SCML [25] is a unified and flexible framework from which three different formulations, namely global, multi-task and local metric learning can be derived. In this work, we are only interested in global and local metrics.

The framework focuses on learning higher-rank linear metrics from a sparse combination of rank-1 "basis metrics". These basis metrics are rank-1 locally discriminative metrics extracted efficiently from the training data by applying Fisher discriminant analysis in the local regions. The task in SCML is to select the most useful basis metrics by learning the combination weights. By using these readily available basis elements, the number of parameters to learn is much smaller (linear in number of basis elements) and projection on to the PSD cone is not required as in other existing methods, which is expensive in case of high dimensional data.

In SCML, a $m \times m$ PSD Mahalanobis matrix $\mathbf{M}$ can be represented using K rank-1 PSD matrices as

$$M = \sum_{i=1}^{K} w_i \mathbf{b}_i {\mathbf{b}_i}^T \tag{2.10}$$

where $w \geq 0$ and $\mathbf{b}_i$'s are d-dimensional column vectors.

SCML uses a set of triplet constraints C where each $(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) \in C$ indicates $\mathbf{x}_i$ should closer to $\mathbf{x}_j$ than $\mathbf{x}_k$. These constraints are constructed in a fully supervised manner as in **LMNN** [29], where $\mathbf{x}_j$ (target neighbor) is the nearest neighbor to $\mathbf{x}_i$ from the same class as $\mathbf{x}_i$ and $\mathbf{x}_k$ (imposter) is the nearest neighbor from a different class. The constraints can also be generated in a weakly supervised manner to incorporate relative comparisons.

The loss function in both the global and local formulations is the margin-based hinge loss function as defined in **LMNN**, i.e.

$$L_w(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) = [1 + d_w(\mathbf{x}_i, \mathbf{x}_j) - d_w(\mathbf{x}_i, \mathbf{x}_k)]_+ \tag{2.11}$$

where $d_w(x_i, x_j)$ is the Mahalanobis distance as defined in Eq. 2.7 parameterized by w, and term $[c]_+ = max(c, 0)$ is the standard hinge loss.

The loss function in Eq. 2.11 incurs a penalty when the imposter $\mathbf{x}_k$ is closer to $\mathbf{x}_i$ than $\mathbf{x}_j$. The learned metric pushes the imposters such that there is a unit margin between the perimeter defined by target neighbors, in which case the loss function would be minimum.

Global Metric Learning

The formulation for SCML-Global is simple and just involves learning K parameters by combining the basis elements into a global metric that satisfies the constraints in C.

$$\min_{w} \frac{1}{|C|} \sum_{(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) \in C} L_w(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) + \beta \left\| w \right\|_1 \tag{2.12}$$

where $\beta \geq 0$ is the regularization parameter with the l_1 norm regularizer that induces sparsity in w, which allows $\mathbf{M}$ to be dependent only on the most useful basis metrics. The optimization function in Eq. 2.12 is linear in both terms, which makes it convex and thus is bounded below by a global minimum. The generalization bound for SCML-Global is provided in [25].

Local Metric Learning

When learning instance-specific metrics, the number of parameters can scale very rapidly w.r.t to number of training points as well as the dimensionality of the data, and thus cause overfitting. Another challenge with existing local metric learning methods is that, there is no proper way of generalizing the learned metrics to new regions of the space at the test time.

SCML's local metric framework addresses the above challenges by; 1) learning a *metric tensor*

(function) $\mathcal{T}(\mathbf{x})$, which is smooth and maps any instance $\mathbf{x}$ to its metric matrix [24]. This allows distances to be computed from $\mathbf{x}$ by its own metric, as

$$d_{\mathcal{T}}(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^T \mathcal{T}(x)(\mathbf{x} - \mathbf{x}') \qquad , \mathcal{T}(\mathbf{x}) = \sum_{i=1}^{K} w_{\mathbf{X},i} \mathbf{b}_i {\mathbf{b}_i}^T \tag{2.13}$$

where $w_{\mathbf{X}}$ is the weight vector for instance $\mathbf{x}$; 2) instead of learning a weight vector for every training instance, the weight vector itself parametrically depends on some embedding of $\mathbf{x}$. The metric tensor in Eq. 2.13, then corresponds to

$$\mathcal{T}(\mathbf{x}) = \sum_{i=1}^{K} ({a_i}^T z_{\mathbf{X}} + c_i)^2 \mathbf{b}_i {\mathbf{b}_i}^T \tag{2.14}$$

where $\mathbf{A} = [\mathbf{a_1} \dots \mathbf{a_K}]^T \in \mathbb{R}^{d' \times K}$, $c \in \mathbb{R}^K$ and $z_{\mathbf{X}}$ is an embedding of $\mathbf{x}$.

The above parametrization reduces the number of parameters to just $K(d'+1)$ and also gives the generalization ability of learning a metric for any point in the feature space. The formulation for SCML-Local corresponds to

$$\min_{\tilde{\mathbf{A}}} \frac{1}{|C|} \sum_{(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) \in C} L_{\mathcal{T}_{\mathbf{A},c}}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) + \beta \left\| \tilde{\mathbf{A}} \right\|_{2,1} \tag{2.15}$$

where $\tilde{\mathbf{A}} \in \mathbb{R}^{(d'+1) \times K}$ is a matrix obtained by concatenating $\tilde{\mathbf{A}}$ and c. The l_2/l_1 norm regularizer introduces sparsity at the column level and forces local metrics to use the same basis subset. SCML-local is nonconvex and is thus subject to local minima.

Chapter 3

Related Work

As we are using a metric learning framework which has been used for multi-class classification, we will discuss other techniques that have introduced multiple metric learning for improving classification performance. We will also discuss metric learning frameworks that have been introduced specifically for improving clustering performance.

Multiple/Local Metric Learning

In the last decade, several multiple metric learning algorithms have been proposed in the nearest neighbor scheme. MM-LMNN [29] proposed by Weinberger and Saul is an extension of LMNN which learns multiple (typically one per class) metrics to overcome overfitting. msNCA [16] by Hong *et al.* learns a function that splits the input space into small regions and then learns a metric per region using NCA [15]. PLML [28] by Wang *et al.* on the other hand learns a combination of full-rank metrics just like SCML, but here weights are not sparse and use no discriminatory information as they depend only on a manifold assumption. Another problem with PLML is that it learns a weight vector for every training instance and there is no way to generalize to new instances. While all these approaches are discriminative in nature, GLML [23] proposed by Noh *et al.* learns a metric for each point in a generative way under some assumption for the class distributions by minimizing the 1-NN expected error. All these approaches learn a large number of parameters which do not scale well with the dimensionality of the data and the number of metrics (classes). Another approach which learns multiple metrics was recently proposed by Bohne *et al.* [5] that uses similarity and dissimilarity pairs instead of triplets to learn a dissimilarity function. The dissimilarity function is obtained by aggregating local metrics which are estimated by solving a convex optimization problem.

Metric Learning and Clustering

Most approaches that integrate metric learning and clustering use semi-supervision in the form of pairwise distance constraints to improve the performance of k-means clustering algorithms.

One of the earliest works by Wagstaff *et al.* [27] imposes pairwise distance constraints on k regions such that similarity pairs lie in the same cluster while dissimilarity pairs lie in different clusters. Xing *et al.* [30] proposed a better approach of integrating metric learning with k-means by using pairwise distance constraints to learn the Mahalanobis metric as well as drive the clustering process as in [27], that helped in generalizing to unseen data points. Similarly, Bilenko *et al.* [4] proposed an approach that iteratively learns a Mahalanobis metric during the clustering process that permits better separated clusters than the previous approaches. They also introduced cluster specific metrics that helps in capturing arbitrary shaped clusters in the data. While these methods use pairwise distance constraints, Kumar *et al.* [20] proposed a metric learning approach similar to [4] but uses triplet constraints to update the metric during the clustering process. All these methods are quite effective but they often perform poorly for datasets with non-linear decision boundaries. To overcome this problem, Yin *et al.* [33] proposed a metric learning approach with nonlinear semi-supervised clustering that improves the separability of the data for clustering. Unlike all the above supervised methods, Ye *et al.* [32] introduced an unsupervised framework that simultaneously performs metric learning and clustering. This approach projects the data onto a low-dimensional manifold with an aim to maximize separability between different classes.

As these methods are designed for k-means clustering, they are limited by and depend on the parameter k, which is the number of clusters (classes). This poses a serious problem in cases where classes in datasets are multimodal and cannot be separated into k clusters. In such cases, density based methods such as DBSCAN [12] has been shown to perform effectively. The method uses two parameters (radius and minimum points within radius) to group points in the dense regions into compact clusters of any shape. SSDBSCAN [21] is a semi-supervised variant of DBSCAN that uses labels of very few points from each class to determine the required parameters. Mean shift clustering on the other hand provides a robust way to detect multiple clusters by locating modes in the density. In [26] Tuzel *et al.* proposed semi-supervised mean shift clustering that imposes similarity pairwise distance constraints in the kernel induced feature space to promote clustering within same class points. An extension to this work by Anand *et al.* in [1] imposes both similarity and dissimilarity pairwise distance constraints to learn a low-rank kernel matrix, that promotes clustering of similar class points and separation of dissimilar class points.

Chapter 4

Proposed Method

In this chapter, we propose a framework that integrates global and local metric learning with mean shift clustering. We also propose kernelization of the algorithm, which helps to achieve better clustering performance in complex datasets that have non-linear decision boundaries.

Suppose we have a set of points $\{\mathbf{x}_i\}_{i=1\ldots n} \in \mathbb{R}^d$ and the corresponding point-wise bandwidths h_i's and for mean shift clustering, we have the multivariate normal *profile*

$$k(x) = \exp\left(-\frac{1}{2}x\right) \qquad , \; x \geq 0 \tag{4.1}$$

which produces the multivariate normal kernel

$$K(\mathbf{x}) = (2\pi)^{(-\frac{d}{2})}\exp\left(-\frac{1}{2}\|\mathbf{x}\|^2\right) \tag{4.2}$$

4.1 Mean Shift Clustering with Global Metric

4.1.1 Bandwidth Estimation

For computing the bandwidths, we use the k-nearest neighbor approach by transforming the points, where k is supplied by us. If $\mathbf{L}$ is the linear transformation corresponding to the learned global metric $\mathbf{M}$ where $\mathbf{M} = \mathbf{L}^T\mathbf{L}$ and $\mathbf{x}_{i,k}$ is the k-nearest neighbor of $\mathbf{x}_i$, then

$$h_i = \|\mathbf{L}\mathbf{x}_i - \mathbf{L}\mathbf{x}_{i,k}\|_2 \tag{4.3}$$

4.1.2 Mean Shift Procedure

Let $\mathbf{M}$ be the global Mahalanobis PSD matrix obtained from Eq. 2.10 by using the learned w and the basis elements. In this case, the sample point density estimator defined in Eq. 2.1

becomes

$$f(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{h_i^d \sqrt{\det \mathbf{M}^{-1}}}\, k(z_i) \qquad , \; z_i = \frac{(\mathbf{x}-\mathbf{x}_i)^T \mathbf{M} (\mathbf{x}-\mathbf{x}_i)}{h_i^2} \tag{4.4}$$

By taking the gradient of $f(\mathbf{x})$ and equating it to zero, we get

$$\nabla f(\mathbf{x}) \;=\; \frac{2\mathbf{M}}{n\sqrt{\det \mathbf{M}^{-1}}} \sum_{i=1}^{n} \frac{1}{h_i^{d+2}} (\mathbf{x}_i - \mathbf{x})\; g(z_i) \;=\; 0$$

where $g(z_i) = -k'(z_i)$.

$$\mathbf{x} \sum_{i=1}^{n} \frac{1}{h_i^{d+2}}\; g(z_i) \;=\; \sum_{i=1}^{n} \frac{\mathbf{x}_i}{h_i^{d+2}}\; g(z_i)$$

$$\mathbf{x} = \frac{\sum_{i=1}^{n} \frac{\mathbf{x}_i}{h_i^{d+2}}\; g(z_i)}{\sum_{i=1}^{n} \frac{1}{h_i^{d+2}}\; g(z_i)} \tag{4.5}$$

which is the same expression as Eq. 2.5 but with Mahalanobis distance instead of the standard Euclidean distance. Here, using $\mathbf{M}$ and h_i is equivalent to first rotating and scaling the input space and then again scaling each dimension equally by $\frac{1}{h_i}$.

4.2 Mean Shift Clustering with Local Metrics

4.2.1 Bandwidth Estimation

We use the same technique as in previous section but for estimating the bandwidth for point $\mathbf{x}_i$, its own $\mathbf{L}_i$ is used

$$h_i = \left\| \mathbf{L}_i \mathbf{x}_i - \mathbf{L}_i \mathbf{x}_{i,k} \right\|_2 \tag{4.6}$$

4.2.2 Mean Shift Procedure

Let $\{\mathbf{M}_i\}_{i=1\ldots n}$ be the local Mahalanobis PSD matrices obtained from *metric tensor* $\mathcal{T}(\mathbf{x}_i)$, which maps every point $\mathbf{x}_i$ to its metric $\mathbf{M}_i$. Therefore, the problem becomes heteroscedastic as every point has its own metric to capture the varying local structure. The sample point density estimator is then

$$f(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{h_i^d}\; \frac{k(z_i)}{\sqrt{\det \mathbf{M}_i^{-1}}} \qquad , \; z_i = \frac{(\mathbf{x}-\mathbf{x}_i)^T \mathbf{M}_i (\mathbf{x}-\mathbf{x}_i)}{h_i^2} \tag{4.7}$$

which is the average of differently rotated and scaled kernels centered at each data point.

Following a similar derivation as in the previous section, we get

$$\nabla f(\mathbf{x}) \;=\; \frac{2}{n}\sum_{i=1}^{n} \frac{\mathbf{M}_i}{h_i^{d+2}\sqrt{\det \mathbf{M}_i^{-1}}}(\mathbf{x}_i - \mathbf{x})\; g(z_i) \;=\; 0$$

$$= \frac{2}{n}\sum_{i=1}^{n} \mathbf{B}_i(\mathbf{x}_i - \mathbf{x})\; g(z_i) \;=\; 0$$

where $\mathbf{B}_i = \frac{\mathbf{M}_i}{h_i^{d+2}\sqrt{\det \mathbf{M}_i^{-1}}}$

$$\left[\sum_{i=1}^{n} \mathbf{B}_i\; g(z_i)\right]\mathbf{x} \;=\; \sum_{i=1}^{n} \mathbf{B}_i\; \mathbf{x}_i\; g(z_i)$$

$$\mathbf{x} \;=\; \left[\sum_{i=1}^{n} \mathbf{B}_i\; g(z_i)\right]^{-1}\left[\sum_{i=1}^{n} \mathbf{B}_i\; \mathbf{x}_i\; g(z_i)\right] \tag{4.8}$$

4.3 Kernelizing the Algorithm

For datasets that have non-linear boundaries and are inseparable in the input feature space, kernelization improves the separability between clusters when metric learning is performed in the kernel induced feature space. To kernelize the algorithm, we obtain a mapping of every point $\mathbf{x}$ in a high dimensional feature space and then apply metric learning on these new mapped points.

Let K_σ be the $n \times n$ kernel matrix obtained with the Gaussian kernel

$$K_\sigma(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right) \tag{4.9}$$

where $\mathbf{x}_i,\ \mathbf{x}_j \in \mathbb{R}^d$ and σ is the bandwidth of the kernel.

By using singular value decomposition (SVD), the kernel matrix decomposes into

$$K_\sigma = USV^T \tag{4.10}$$

where U and V are right and left singular vectors, and S are the singular values of K_σ. As K_σ is a symmetric positive semidefinite matrix, USV^T is equivalent to USU^T.

To get a low dimensional representation of the points in the kernel space, we choose first r singular vectors that explain 95% of the total variance. Then the new representation of points using U_r and S_r becomes

$$\hat{X} = U_r\sqrt{S_r} \tag{4.11}$$

Now, the metric learning and mean shift clustering is performed in the kernel induced space

using $\hat{X}$.

Chapter 5

Experiments and Results

In this section we will evaluate the performance of adaptive mean shift clustering with global and local metric learning and compare them.

5.1 Datasets and Preprocessing

We use 5 datasets from UCI[1] and USPS[2] handwritten digits for our experiments. We use the same preprocessing strategy and parameter settings as used by SCML [25]. We preprocess all the datasets by first standardizing the input features so that their mean is 0 and variance is 1. Then we normalize the instances so that their L2-norm is 1. We also use PCA for reducing the dimensionality of USPS by keeping the PCA components that conserved 95% of their total variance. We split the data into train/test that is mentioned in table 5.1. All the results are averaged over 5 random splits.

Datasets	**# Samples**	**# Classes**	**# Features**	**Data split**
WDBC	569	2	30	60%/40%
Wine	178	3	13	60%/40%
Vehicle	846	4	18	60%/40%
Segment	2310	7	18	60%/40%
Vowel	990	11	10	60%/40%
USPS	11000	10	256	1200/800

Table 5.1: Datasets for adaptive mean shift clustering using global and local metrics

[1] `http://archive.ics.uci.edu/ml/`
[2] `http://www.cs.nyu.edu/~roweis/data.html`

5.1.1 Setup for Global and Local Metric Learning

Setup for metric learning is the same as used in SCML [25]. We use the same basis set and triplets for both the formulations. Triplets are generated by selecting 3 nearest neighbors with same label and 10 nearest neighbors with different label for each instance. Kernel PCA is used for embedding of instances in SCML-Local. Parameter setting for all the datasets is given below in table 5.2.

Datasets	**# Basis Elements**	**Embedding Dimension**
WDBC	200	40
Wine	200	40
Vehicle	400	40
Segment	400	40
Vowel	400	40
USPS	800	100

Table 5.2: Parameter Setting for SCML

5.1.2 Setup for Mean Shift Clustering and Kernelization

The parameter k for estimating bandwidths in mean shift is provided by us and it varies for every dataset. The value of k is varied between six values and we select those results which give reasonable number of clusters and error rate. We will show the effect of k on the number of clusters for a few datasets. The bandwidth parameter σ for the kernelization is also provided by us. The best σ is selected by viewing the difference between the initial kernel matrix and the learned kernel matrix.

Datasets	**range of k**		σ
	Global MSC	**Local MSC**	
WDBC	32 - 37	35 - 40	$--$
Wine	20 - 25	22 -27	$--$
Vehicle	13 - 18	15 - 20	1
Segment	32 - 37	35 - 40	1
Vowel	20 - 25	25 - 30	1
USPS	15 - 20	17 - 22	1.5

Table 5.3: Parameter Setting for Mean Shift and Kernelization

5.2 Evaluation Criteria

Since the labels of all the datasets are known, we use Adjusted Rand Index (ARI) [17] and error rate (%) for analyzing the performance of mean shift clustering. We use training labels only for the purpose of labeling and evaluating the clusters obtained from mean shift clustering.

5.2.1 Adjusted Rand Index (ARI)

The Rand Index (RI) measures the percentage of predictions for which, the both predicted and true clusterings agree. A problem with Rand Index is that it does not penalize random clustering assignments. Therefore, ARI is the "adjusted for chance" version of RI, which corrects the agreement that might be solely due to chance between clusterings.

$$ARI = \frac{RI - Expected\_RI}{max(RI) - Expected\_RI} \tag{5.1}$$

It takes a value of 0 for randomly labeled independent clusterings and value of 1 when the clusterings are identical.

5.3 Results and Discussion

5.3.1 Mean Shift Clustering in Input Feature Space

Here, we show the results for mean shift clustering in the input feature space evaluated on the test set. Average Error rates and ARI values are mentioned in the tables 5.4 and 5.5 respectively, and the number of clusters detected are mentioned in table 5.6.

Datasets	**Global MSC**	**Local MSC**
WDBC	4.04 ± 1.47	**3.51 ± 1.49**
Wine	2.53 ± 1.17	**2.25 ± 1.61**
Vehicle	34.67 ± 4.45	**28.05 ± 4.20**
Segment	28.05 ± 3.85	**24.78 ± 1.30**
Vowel	67.42 ± 1.80	**63.56 ± 1.52**
USPS	18.30 ± 2.42	**15.67 ± 2.74**

Table 5.4: Average Error Rates (%) for Global and Local Mean Shift Clustering (MSC)

5.3.2 Discussion

- Local mean shift clustering performs better in all the datasets but, for Vehicle, Segment and Vowel, both global and local versions produce large number of clusters than the number

Datasets	Global MSC	Local MSC
WDBC	0.80 ± 0.06	**0.85 ± 0.06**
Wine	0.92 ± 0.03	**0.93 ± 0.05**
Vehicle	0.45 ± 0.04	**0.52 ± 0.08**
Segment	0.57 ± 0.4	**0.59 ± 0.02**
Vowel	0.17 ± 0.03	**0.19 ± 0.03**
USPS	0.70 ± 0.03	**0.75 ± 0.04**

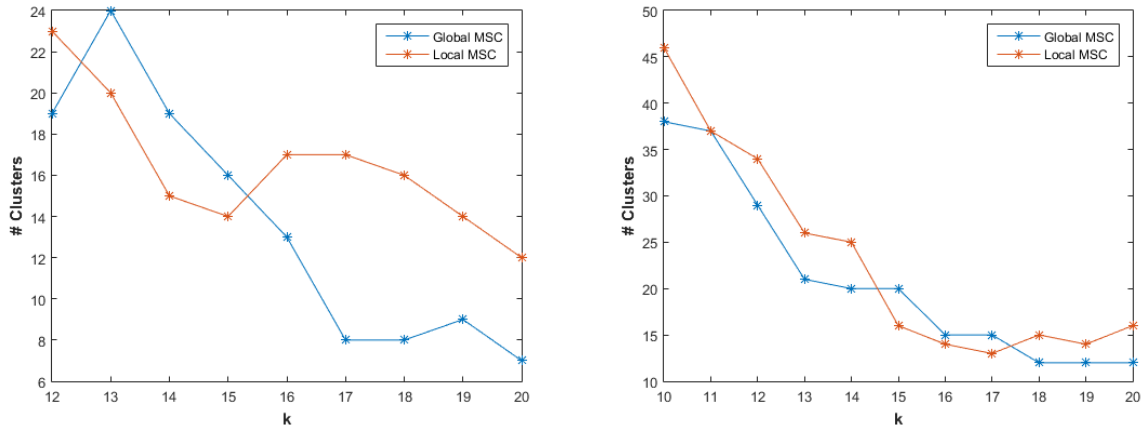
Table 5.5: Average ARI values for Global and Local Mean Shift Clustering (MSC)

Datasets	Global MSC	Local MSC	# Classes
WDBC	2 ± 0	2 ± 0	2
Wine	3.4 ± 0.54	3 ± 0	3
Vehicle	15.2 ± 1.46	13.6 ± 1.48	4
Segment	16.6 ± 0.89	14.4 ± 1.14	7
Vowel	25.2 ± 5.1	24.2 ± 3.8	11
USPS	15.8 ± 1.9	15.2 ± 1.4	10

Table 5.6: Average number of clusters detected

of classes.

- This is due to the relative (triplet) distance constraints that promote local structure within the same class, which results in much more number of clusters than the number of classes.
- Local mean shift clustering also takes longer (larger values of k) than global mean shift clustering to converge to the same number of clusters. This behavior is expected because every point has its own metric to compute distance to other points instead of a global metric.



(a) Vehicle data set

(b) USPS data set

Figure 5.1: Comparison between Global and Local Mean Shift Clustering showing the relationship between k and the number of clusters detected.

5.3.3 Mean Shift Clustering in Kernel Induced Feature Space

We perform kernelization only on Vehicle, Segment, Vowel and USPS because mean shift clustering performs poorly both in terms of error rate and number of clusters.

Datasets	**Global MSC**	**Local MSC**
Vehicle	29.56 ± 2.18	**27.35 ± 1.54**
Segment	19.25 ± 1.56	**16.75 ± 1.32**
Vowel	12.32 ± 2.95	**10.52 ± 1.75**
USPS	7.1 ± 1.53	**6.54 ± 1.12**

Table 5.7: Average Error Rates (%) for Kernelized Global and Local Mean Shift Clustering (MSC)

Datasets	**Global MSC**	**Local MSC**
Vehicle	0.61 ± 0.02	**0.63 ± 0.03**
Segment	0.63 ± 0.02	**0.66 ± 0.02**
Vowel	0.78 ± 0.05	**0.81 ± 0.04**
USPS	0.86 ± 0.03	**0.88 ± 0.01**

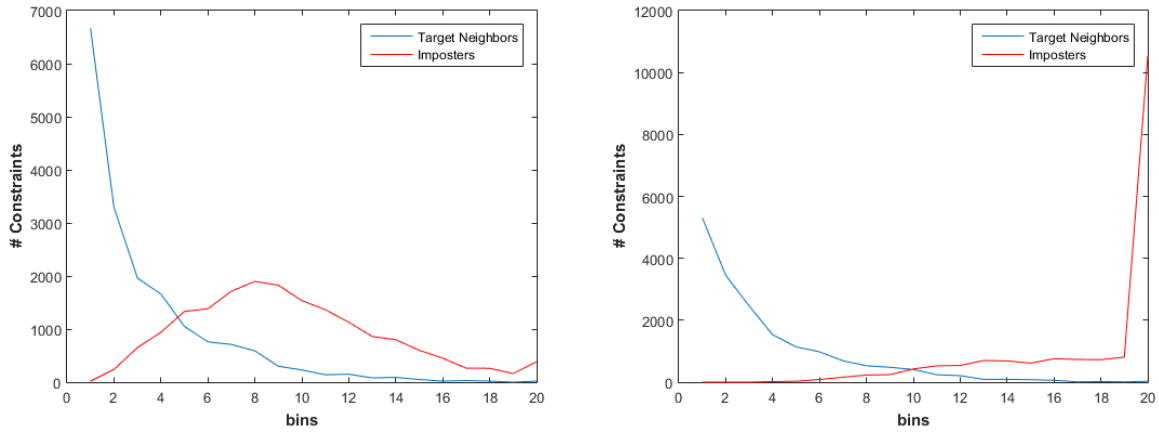
Table 5.8: Average ARI values for Kernelized Global and Local Mean Shift Clustering (MSC)

Datasets	**Global MSC**	**Local MSC**	**# Classes**
Vehicle	7.4 ± 0.55	8 ± 0.7	4
Segment	10.4 ± 1.14	10.4 ± 1.14	7
Vowel	16 ± 0.71	15.8 ± 0.84	11
USPS	10.6 ± 0.54	10.4 ± 0.54	10

Table 5.9: Average number of clusters detected

5.3.4 Discussion

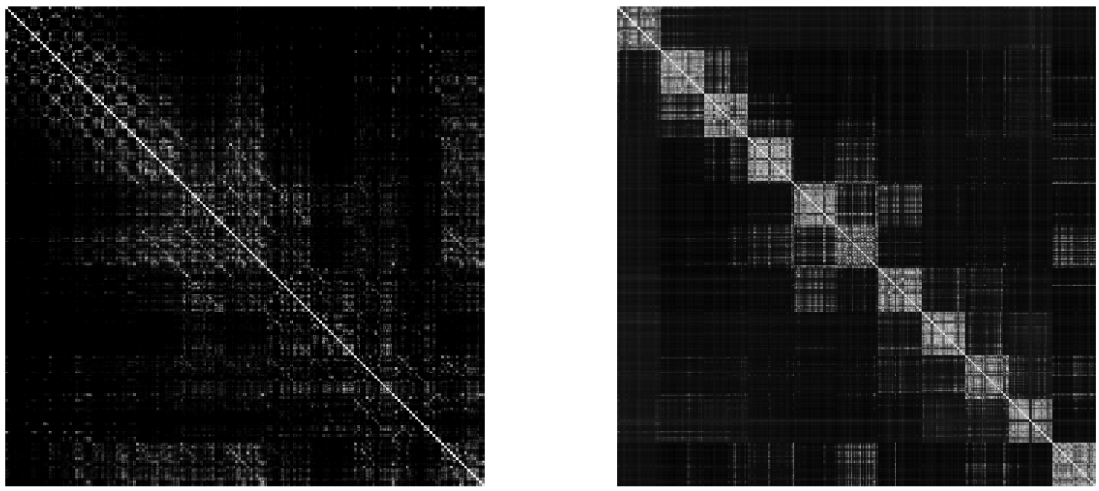
- Kernelization improves the separation between clusters for all the four datasets. This is because more number of distance constraints are satisfied by a larger margin between target neighbors and imposters. Fig. 5.2 shows the difference between distance constraints after learning in input space and kernel induce space for Vowel dataset. The number of clusters detected significantly reduces as shown in table 5.9.
- Effect of kernelization and metric learning can be seen in the kernel matrices in Fig. 5.3 for Vowel dataset. There is a significant difference in the kernel matrices, with the learned kernel matrix nearly block diagonal.



(a) Input feature space

(b) Kernel induced feature space

Figure 5.2: Comparison of constraints between point-neighbor pairs and point-impostor pairs in input feature space and kernel induced feature space for the Vowel dataset.



(a) Initial Kernel Matrix

(b) Kernel Matrix after metric learning

Figure 5.3: Comparison between initial kernel matrix and kernel matrix (990 x 990) after metric learning on Vowel data set.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

In this work, we investigated the problem of mean shift clustering using metric learning. We wanted to determine the performance improvement of mean shift clustering using local metrics over a global metric. For metric learning, we used the recently proposed SCML framework which uses locally discriminatory low-rank metrics to learn global and local metrics, and has been shown to work very effectively in k-NN classification. We integrated global and local metric with mean shift and derived two different formulations for mean shift computation. We also kernelized the algorithm for problems where the data is not linearly separable and it is not feasible to get homogeneous clusters. We performed experiments on 6 datasets of varying difficulty and observed that local mean shift clustering performs better than global mean shift clustering.

6.2 Future Scope

Large margin nearest neighbor framework promotes local regions of points in the input space and thus is not much advantageous for clustering. Using triplet constraints with the clustering framework to learn a metric (or metrics) should be more beneficial and thus can be a prospective extension to the problem.

Bibliography

[1] Anand, S., Mittal, S., Tuzel, O., and Meer, P. Semi-supervised kernel mean shift clustering. *IEEE transactions on pattern analysis and machine intelligence 36*, 6 (2014), 1201–1215.

[2] Bar-Hillel, A., Hertz, T., Shental, N., and Weinshall, D. Learning distance functions using equivalence relations. In *ICML* (2003), vol. 3, pp. 11–18.

[3] Bellet, A., Habrard, A., and Sebban, M. A survey on metric learning for feature vectors and structured data. *arXiv preprint arXiv:1306.6709* (2013).

[4] Bilenko, M., Basu, S., and Mooney, R. J. Integrating constraints and metric learning in semi-supervised clustering. In *Proceedings of the twenty-first international conference on Machine learning* (2004), ACM, p. 11.

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

[6] Collins, R. T. Mean-shift blob tracking through scale space. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on* (2003), vol. 2, IEEE, pp. II–234.

[7] Comaniciu, D., and Meer, P. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence 24*, 5 (2002), 603–619.

[8] Comaniciu, D., Ramesh, V., and Meer, P. Real-time tracking of non-rigid objects using mean shift. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on* (2000), vol. 2, IEEE, pp. 142–149.

[9] Comaniciu, D., Ramesh, V., and Meer, P. The variable bandwidth mean shift and data-driven scale selection. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on* (2001), vol. 1, IEEE, pp. 438–445.

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

[11] Demiriz, A., Bennett, K. P., and Embrechts, M. J. Semi-supervised clustering using genetic algorithms. *Artificial neural networks in engineering (ANNIE-99)* (1999), 809–814.

[12] Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd* (1996), vol. 96, pp. 226–231.

[13] Fukunaga, K., and Hostetler, L. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on information theory 21*, 1 (1975), 32–40.

[14] Georgescu, B., Shimshoni, I., and Meer, P. Mean shift based clustering in high dimensions: A texture classification example. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on* (2003), IEEE, pp. 456–463.

[15] Goldberger, J., Hinton, G. E., Roweis, S. T., and Salakhutdinov, R. Neighbourhood components analysis. In *Advances in neural information processing systems* (2004), pp. 513–520.

[16] Hong, Y., Li, Q., Jiang, J., and Tu, Z. Learning a mixture of sparse distance metrics for classification and dimensionality reduction. In *2011 International Conference on Computer Vision* (2011), IEEE, pp. 906–913.

[17] Hubert, L., and Arabie, P. Comparing partitions. *Journal of classification 2*, 1 (1985), 193–218.

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

[19] Kulis, B., Sustik, M. A., and Dhillon, I. S. Low-rank kernel learning with bregman matrix divergences. *Journal of Machine Learning Research 10*, Feb (2009), 341–376.

[20] Kumar, N., and Kummamuru, K. Semisupervised clustering with metric learning using relative comparisons. *IEEE Transactions on Knowledge and Data Engineering 20*, 4 (2008), 496–503.

[21] Lelis, L., and Sander, J. Semi-supervised density-based clustering. In *2009 Ninth IEEE International Conference on Data Mining* (2009), IEEE, pp. 842–847.

[22] Liu, W., Mu, C., Ji, R., Ma, S., Smith, J. R., and Chang, S.-F. Low-rank similarity metric learning in high dimensions. In *AAAI* (2015), pp. 2792–2799.

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

[24] Ramanan, D., and Baker, S. Local distance functions: A taxonomy, new algorithms, and an evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 33*, 4 (2011), 794–806.

[25] Shi, Y., Bellet, A., and Sha, F. Sparse compositional metric learning. *arXiv preprint arXiv:1404.4105* (2014).

[26] Tuzel, O., Porikli, F., and Meer, P. Kernel methods for weakly supervised mean shift clustering. In *2009 IEEE 12th International Conference on Computer Vision* (2009), IEEE, pp. 48–55.

[27] Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S., et al. Constrained k-means clustering with background knowledge. In *ICML* (2001), vol. 1, pp. 577–584.

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

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

[30] Xing, E. P., Ng, A. Y., Jordan, M. I., and Russell, S. Distance metric learning with application to clustering with side-information. *Advances in neural information processing systems 15* (2003), 505–512.

[31] Yang, C., Duraiswami, R., and Davis, L. Efficient mean-shift tracking via a new similarity measure. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)* (2005), vol. 1, IEEE, pp. 176–183.

[32] Ye, J., Zhao, Z., and Liu, H. Adaptive distance metric learning for clustering. In *2007 IEEE Conference on Computer Vision and Pattern Recognition* (2007), IEEE, pp. 1–7.

[33] Yin, X., Chen, S., Hu, E., and Zhang, D. Semi-supervised clustering with metric learning: An adaptive kernel method. *Pattern Recognition 43*, 4 (2010), 1320–1333.