A Brief Report on Metric Learning

Prateek Chanda

Abstract

There are several machine learning algorithms like K-means clustering algorithm (for clustering purposes) and K-nearest neighbor classification algorithm (for classification purposes) are implemented based on the similarity metrics being used to calculate similarity among the objects. Similarity metric can be defined as a function for determining similarity between two objects in the dataset under consideration. Hence the performance of such algorithms depends heavily on the similarity criteria being used. For example, in K-means , Euclidian Distance as a metric provides good performance for most cases but if we use any other metric for the task, then performance may degrade. The choice of the metric depends on the data and the problem at hand. Using the same data if we define multiple problems , then using the same similarity metric for all such problems is a bad choice. The notion of metric plays an important role in many domains such as classification, regression, clustering. Metric Learning can thus be used to learn a distance function over objects present in the data.

Motivation

As we discussed the performance of the machine learning algorithm depends heavily upon the similarity metric being used. In a clustering application it is desirable that similar objects are clustered together and dissimilar objects lie in separate clusters.

Hence the choice of an appropriate similarity metric is crucial to the performance of such a clustering algorithm.

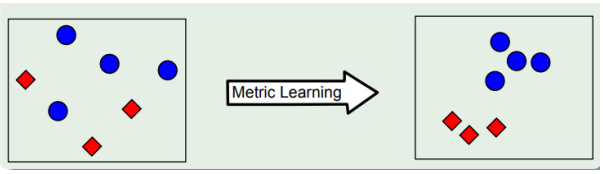

Figure 1: Clustering similar points through suitable learnt metric

Given a set of m points in the dataset

$$x_i \subseteq R^n \quad \forall i \in (1, m)$$

S is the set containing all similar objects

$$S : (x_i, x_j) \in S \quad x_i \ and \ x_j \ are \ similar$$

We therefore want to find a distance learning metric such that it keeps into account the distance relationship objects and similar objects are placed in close proximity.

The learning distance metric

$$d_A(x, y) = ||x - y||_A = \sqrt[2]{(x - y)^T \cdot A \cdot (x - y)}$$

In order to consider this distance function as a suitable metric , we must ensure the non-negativity and triangular inequality holds and thus A must be semi-definite $(A \succeq 0)$.
Setting $A = I$ gives us the Euclidean distance metric. Similarly depending on the the values of the matrix A, we can get different metric similarities upon the same data-set. Metric Learning is thus concerned with learning the suitable distance similarity metric from the input data space which in turn is achieved by learning the matrix A.
The following diagram shows some key properties of metric learning.

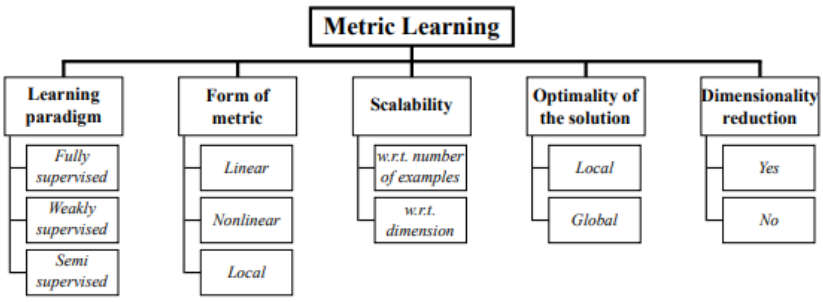

Figure 2: Five key properties of metric learning algorithms.

Previous Research Work

In recent years, there has been considerable work in improving the similarity measures being used in different machine learning applications like clustering, classification or regression. While work has been done improve classifier accuracy and in clustering through research on internal and external measures, improvement of such similarity metric being used often overlooked upon. Hence for the past few years there has been considerable research on distance metric learning.

Each metric learning algorithm has its intrinsic properties (e.g., type of metric, good scalability with dimensionality, generalization guarantees, etc) and emphasis should be placed on those when deciding which method to apply to a given problem.

Learning Paradigm

Fully supervised: the metric learning algorithm has access to a set of labeled training instances where each training example is composed of an instance and a label (or class)
Weakly supervised: the algorithm has no access to the labels of individual training instances. It is only provided with side information in the form of sets of constraints similar constraints, dissimilar constraints and relative constraints.

Metric Learning algorithms can be broadly divided into two categories: **supervised distance metric learning** and **unsupervised distance metric learning**.

Unsupervised Metric Learning

In case of unsupervised metric learning, the goal is to find a low-dimensional embedding of the entire input data space such that the distance relationships between the observed data remains preserved Much work has been done in this regard like Multidimensional Scaling (MDS), and Locally Linear Embedding (LLE). We use the IRIS data set to illustrate the following algorithms.

- Relevant Component Analysis(RCA) is developed for unsupervised learning using equivalence relations. RCA[29] is another one of the older algorithms. It learns a full rank Mahalanobis distance metric based on a weighted sum of in-class covariance matrices. It applies a global linear transformation to assign large weights to relevant dimensions and low weights to irrelevant dimensions. Those relevant dimensions are estimated using chunklets, subsets of points that are known to belong to the same class.

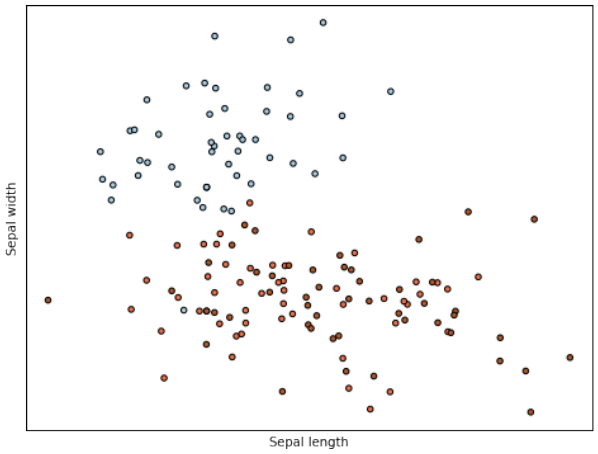

Figure 3: Clustering Iris Data using RCA

Supervised Metric Learning

In supervised metric learning the training set is cast into pairwise constraints: : one where pairs of data points that belong to the same classes, and another where pairs of data points belong to different classes. [1] formulates distance metric learning as a constrained convex programming problem. It learns a global distance metric that minimizes the distance between the data pairs in the same classes subject to the constraint that the data pairs in the dissimilar classes are well separated.
This can also be divided into two categories: **the global distance metric learning**, and the **local distance metric learning**.

Supervised Local Distance Metric Learning

It learns the metric from a global perspective i.e. to satisfy all the pairwise constraints simultaneously Some well-known metric learning algorithm proposed in this field are

proposed here. We use the Iris Data set as an example to study each of the proposed algorithms.

Local Fisher Discriminant Analysis (LFDA)
LFDA mainly deals with dimensionality reduction following the likes of traditional Fisher discriminant analysis (FDA). However, FDA tends to give undesired results if samples in some class form several separate clusters, i.e., multimodal. LFDA solves this problem faced by FDA by taking a local structure of the data into account so the multimodal data can be embedded appropriately.

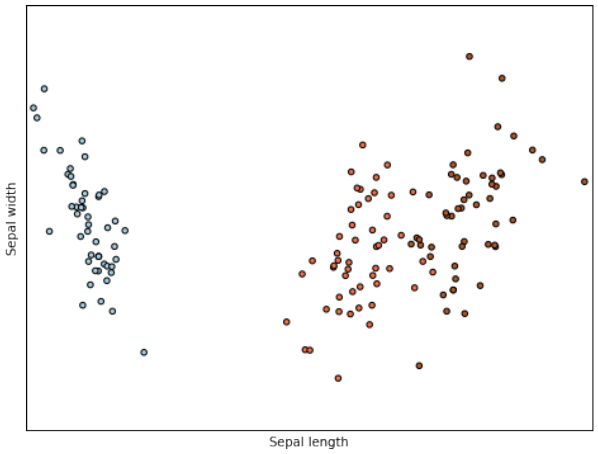

Figure 4: Clustering the Iris Data using LFDA

Information-Theoretic Metric Learning (ITML) The Information-Theoretic Metric Learning proposed by Davis et al. considers the regularizer

$$r(A) = tr(A) - \log\det(A)$$

. It uses the regularizer that automatically enforces a Semi-Definite Positive Matrix condition - the LogDet divergence. It uses soft must-link or cannot like constraints, and a simple algorithm based on Bregman projections.

Supervised Global Distance Metric Learning

It tries to learn the metric by satisfying local pairwise constraints. Few well-known algorithms in this domain include the following:
Discriminative Component Analysis (DCA) and **Kernel Discriminative Component Analysis (KDCA)**

Form of Metric

One may identify three main families of metrics:

Linear metrics, such as the Mahalanobis distance. Their expressive power is limited but they are easier to optimize (they usually lead to convex formulations, and thus global optimality of the solution) and less prone to overfitting.

Nonlinear metrics, such as the X^2 histogram distance. They often give rise to nonconvex formulations (subject to local optimality) and may overfit, but they can capture nonlinear variations in the data.

Local metrics, where multiple (linear or nonlinear) local metrics are learned (typically simultaneously) to better deal with complex problems, such as heterogeneous data. They are however more prone to overfitting than global methods since the number of parameters they learn can be very large.

Metric Learning Usage Through an Example

There are many applications of metric learning , as we have discussed before. Metric Learning can be useful in case of clustering examples, where determining how good the distance metric is crucial to the clustering performance. We will be using the IRIS data-set as for working on our clustering example. We use only two features - *Sepal Width* and *Sepal Length* of the data set that are being plotted.

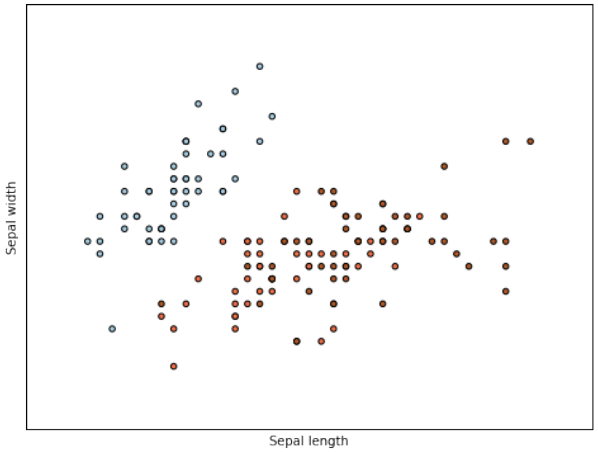

Figure 5: Using normal Euclidean distance as a similarity measure

We can, with prior knowledge of which points are supposed to be closer, figure out a better way to understand distances between points. Especially in higher dimensions when Euclidean distances are a poor way to measure distance, this becomes very useful. Using an appropriate metric learning algorithm, we get a more accurate results as opposed to the previous.

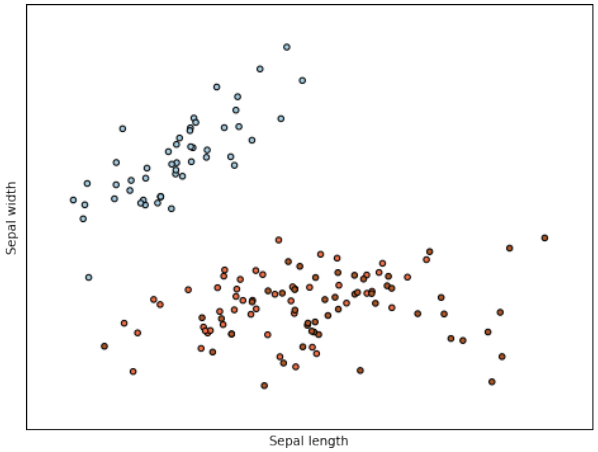

Figure 6: Learning a new distance metric from the data

Theoretical Analysis of Metric Algorithms

As we have seen before metric learning is all about learning the distance matrix A in the following equation

$$d_A(x, y) = ||x - y||_A = \sqrt[2]{(x - y)^T \cdot A \cdot (x - y)}$$

This matrix will depend heavily on what is the distribution of the data.
If the data is of spherical shape , then the distance metric ought to be *Euclidean metric*. In that case $A = I$.

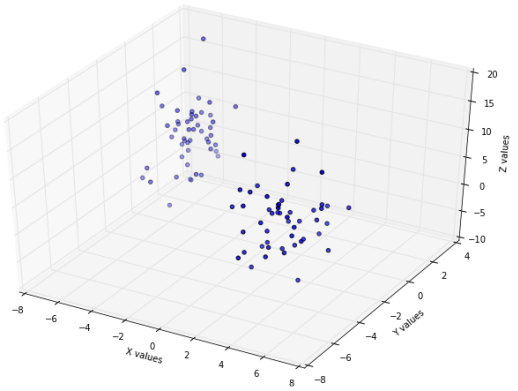

Figure 7: Spherical Data Distribution

On the other hand if the metric turns out to be Mahalanobis distance, then the distance metric can be written as follows.

$$d_A(x, y) = ||x - y||_A = \sqrt[2]{(x - y)^T \cdot \sigma^{-1} \cdot (x - y)}$$

where σ is the covariance matrix of the data. In many cases, the true covariance is unknown, and so a point estimate of the covariance (e.g., the sample covariance) is used.

We discuss here some of the theoretical aspects of well known algorithms implemented along with their merits and demerits.

MMC

This proposed work of Xing was the first established work in the field of metric learning. It relies on a convex formulation with no regularization, which aims at maximizing the sum of distances between dissimilar points while keeping the sum of distances between similar examples small

$$\max_{M \in S} \sum_{(x_i, x_j) \epsilon D} d_M(x_i, x_j)$$

such that

$$\sum_{(x_i, x_j) \epsilon S} d_M(x_i, x_j) \leq 1$$

The algorithm uses a simple projected gradient approach requiring the full eigenvalue decomposition of M at each iteration.

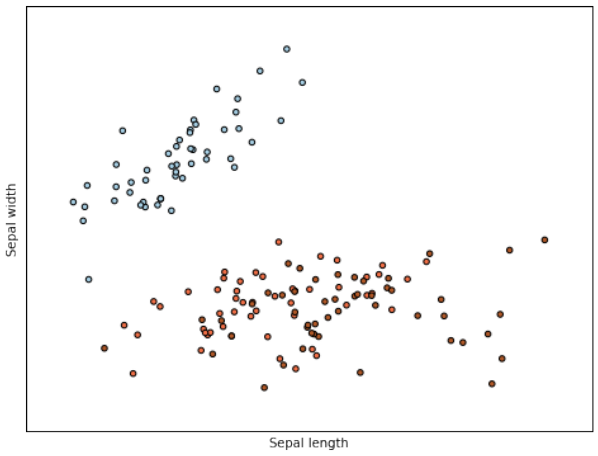

Figure 8: Clustering IRIS data using MMC

Disadvantage The method does not take regularization into account. Hence for a larger dimension it is almost intractable to work out with this method.

ITML

This approach is quite better than the previous one since it calculates a loss using LogDet divergence and tries to minimize that loss.

$$D_{ld}(M, M_o) = tr(M, M_o^{-1}) - logdet(M, M_o^{-1}) - d$$

where d is the dimension of the input space and M_o is some positive definite matrix we want to remain close to.
The objective is to minimize this $D_{ld}(M, M_o)$.

ITML can be formulated as

$$\min_{M \in S} D_{ld}(M, M_o) + \gamma \sum_{i,j} \epsilon_{ij}$$

s.t

$$d^2_M(x_i, x_j) \leq u + \epsilon_{ij} \forall (x_i, x_j) \in S$$
$$d^2_M(x_i, x_j) \geq u - \epsilon_{ij} \forall (x_i, x_j) \in D$$

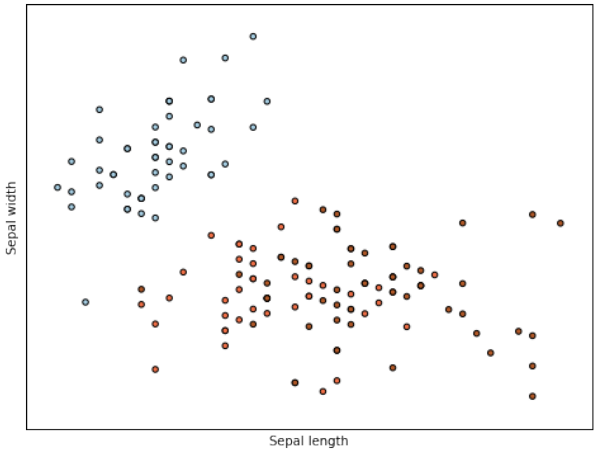

Figure 9: Clustering the Iris Data using ITML

Disadvantage In most cases M_o is set to I and thus the regularization aims at keeping the learned distance close to the Euclidean distance.
Hence the matrix learned is dependent on which base matrix we choose.

LMNN

It is much better as compared to the other methods since the constraints are defined in a more local manner, overcoming intractability in higher dimensions. Formally, the constraints are defined in the following way:

$$S = (x_i, x_j) : y_i = y_j and x_j belongs to the k - neighborhood of x_i$$

$$R = (x_i, x_j, x_k) : (x_i, x_j) \in S, y_i \neq y_k$$

The distance is learned using the following convex program:

$$\min_{M \in S} (1 - \mu) \sum_{(x_i, x_j) \in S} d^2_M(x_i, x_j) + \mu \sum_{i,j,k} \epsilon_{ijk}$$

s.t

$$d^2_M(x_i, x_k) - d^2_M(x_i, x_j) \geq 1 - \epsilon_{ijk} \forall (x_i, x_j, x_k) \in R$$

where $\mu[0, 1]$ controls the pull/push trade-off.

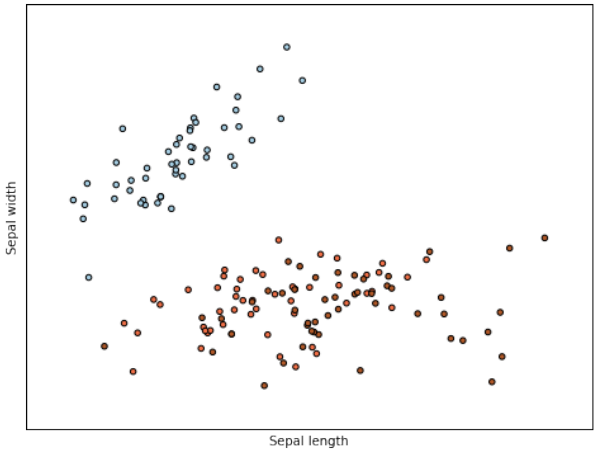

Figure 10: Clustering Iris Data using LMNN

Disadvantages LMNN generally performs very well in practice, although it is sometimes prone to overfitting due to the absence of regularization, especially in high dimension. It is also very sensitive to the ability of the Euclidean distance to select relevant target neighbors.

NCA

Neighborhood Component Analysis (NCA) algorithm proposed in [9] learns a Mahalanobis distance metric for the KNN classifier by maximizing the leave-one-out cross validation. It aims at "learning" a distance metric by finding a linear transformation of input data such that the average leave-one-out (LOO) classification performance is maximized in the transformed space. The key insight to the algorithm is that a matrix A corresponding to the transformation can be found by defining a differentiable objective function for A, followed by use of an iterative solver such as conjugate gradient descent. One of the benefits of this algorithm is that the number of classes k can be determined as a function of A, up to a scalar constant.

They use the decomposition $M = L^T.L$ and they define the probability that x_i is the neighbor of x_j by the following probability

$$p_{ij} = \frac{\exp(-||L.x_i - L.x_j||_2^2)}{\sum_{l \neq i} \exp(-||L.x_i - L.x_l||_2^2)} p_{ii} = 0$$

The probability that x_i is correctly classified is

$$p_i = \sum_{j:y_j = y_i} p_{ij}$$

So we would want to maximize this probability. Hence we need to solve the following

$$\max_L \sum_i p_i$$

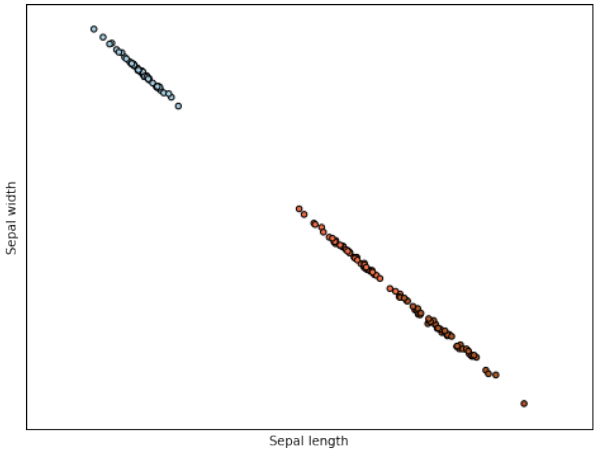

Figure 11: Clustering Iris Data using NCA

Disadvantages NCA calculates a softmax function upon Euclidean distances between a particular object and its neighborhood. The data set might be such that Euclidean Distance as a metric might not be a good option and hence the classification results will not be upto the mark. So dependency on only Euclidean Distance to calculate probability for any generic dataset may not be a good option.

Evaluation of the Proposed Metric Algorithms

The Silhouette Index is one of the best methods for checking how close is the clustering solution to the optimal one. The Silhouette Coecient is calculated using the mean intra-cluster distance and the mean nearest-cluster distance for each sample.

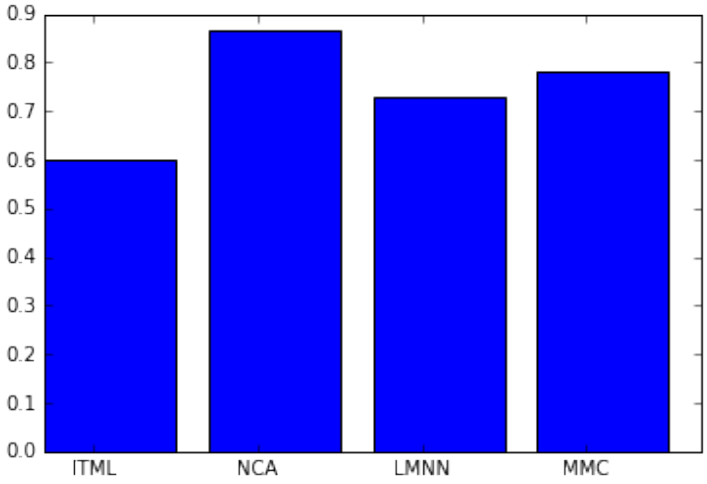

Figure 12: Silhoutte Score of the proposed Algorithms

So as it can be seen NCA performs better with respect to the other proposed approaches since it uses a more probabilistic approach.

We tested the performance of the proposed methods with respect to different UCI Datasets and compared their **Clustering Accuracy**.

Let $\hat{c_i}(i = 1, \cdots, m)$ be the cluster to which point x_i is assigned by an automatic clustering algorithm and let c_i be some "correct" or desired
Accuracy $= \sum_{i \geq j} \frac{1(1(c_i = c_j) = 1(\hat{c_i} = \hat{c_j}))}{0.5m(m-1)}$
where $1(.)$ is the indicator function $(1(True) = 1, 1(False) = 0)$ This is equivalent to the probability that for two points x_i and x_j drawn randomly from the dataset, our clustering $\hat{c}$ agrees with the "true" clustering c on whether x_i and x_j belong to the same or different clusters.
We compared the following performances based on the accuracy model described here.

(a) Iris Data Set

(b) Boston Housing DataSet

(c) Wine DataSet

(d) Breast Cancer DataSet

Figure 13: Plots of Cluster Accuracy for Different UCI Datasets

Non-Linear Metric Algorithms

There have been previous attempts to generalize learning linear distances to nonlinear metrics.

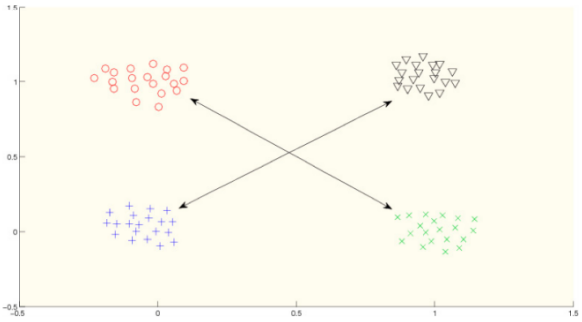

Figure 14: Limitation of linear methods

In the above figure suppose we would like that the black and blue points should be similar to one another, while the red and green point should also be similar to one another (while simultaneously the black and blue points should be dissimilar to the red and green points). No global linear metric will suffice for enforcing the constraints.

Application of metric algorithms to Non-linear Data

To solve the above limitations we use the X^2 histogram distance since they can capture the non-linear variations in the data. Several notable algorithms have been proposed in this respect out of which X^2-LMNN deserves special mention.

In order to learn linear metrics, most proposed approaches are quite robust and have out-of-the-box usability. So far non-linear approaches to metric learning have not been able to replicate this success. The proposed X^2-LMNN is an extension to the LMNN algorithm which provides non-linear capabilities.

The proposed algorithm is specialized for histogram data. It generalizes the non-linear X^2 -distance and learns a metric that strictly preserve the histogram properties of input data on a probability simplex. It successfully combines the simplicity and elegance of the LMNN objective and the domain-specific expressiveness of the X^2-distance.

Histogram Distances

The abundance of such data has sparked the development of several specialized distance metrics designed to compare histograms. . Transforming the inputs with a linear transformation learned with LMNN will almost certainly result in a loss of their histogram properties and the ability to use such distances

X^2 histogram distance

The X^2 distance is a bin-to-bin distance measurement, which takes into account the size of the bins and their differences. The X^2 distance between x_i and x_j can be given by the following expression

$$X^2(x_i, x_j) = \frac{1}{2} \sum_{f=1}^{d} \frac{([x_i]_f - [x_j]_f)^2}{[x_i]_f + [x_j]_f}$$

where $[x_i]_f$ indicates the f^{th} feature value of the vector x_i.

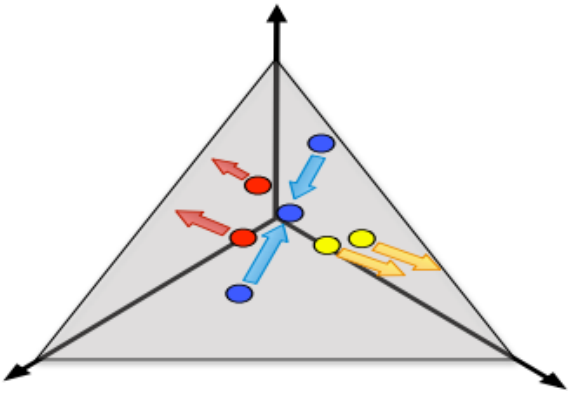

Figure 15: A schematic illustration of the X^2-LMNN optimization. The mapping is constrained to preserve all inputs on the simplex S^3(grey surface). The arrows indicate the push (red and yellow) and pull (blue) forces from the X^2- LMNN objective

Performance of Non-Linear Metric over Traditional Approach

For comparing the performance of the above non-linear metric algorithm we use the **corel** dataset which is a popular data set for histogram distance metric. The dataset contains 773 landscape images in 10 different classes: people in Africa, beaches, outdoor buildings, buses, dinosaurs, elephants, flowers, horses, mountains, and food. There are 50 to 100 images in each class. All images have two types of representation: SIFT and CSIFT.

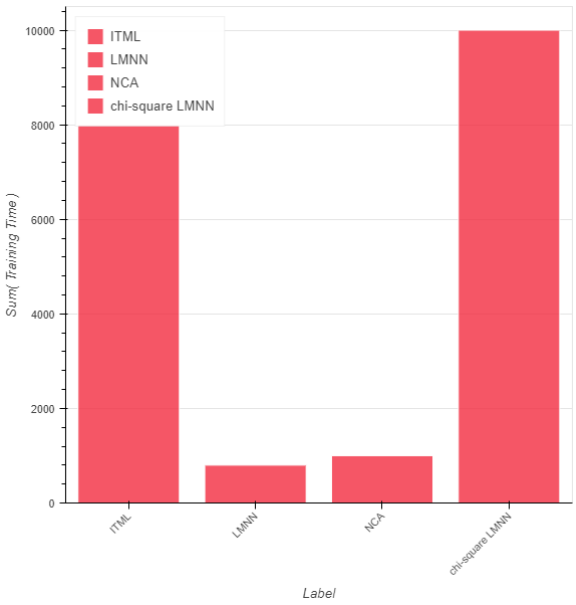

Figure 16: Training times for CSIFT Dataset

Applications

Metric Learning has been used in many applications till date. Recently, it has been applied to problems as diverse as link prediction in networks [3], state representation in reinforcement learning [4] , music recommendation [5], partitioning problems [6] to name a few. In the following, we list three large fields of application where metric learning has been shown to be very useful.

Computer vision There is a great need of appropriate metrics in computer vision, not only to compare images or videos in ad-hoc representationssuch as bags-of-visual-words Li and Perona [7] but also in the pre-processing step consisting in building this very representation . For this reason, there exists a large body of metric learning literature dealing specifically with computer vision problems, such as image classification Mensink et al. [8], object recognition (Frome et al., 2007 [9]) or face recognition (Guillaumin [10]).

Figure 17: In one application, our notion of distance between faces may depend on the pose, whereas in another application it may depend on the identity. So need to use different similarity metrics in different applications

Information retrieval The objective of many information retrieval systems, such as search engines, is to provide the user with the most relevant documents according to his/her query. This ranking is often achieved by using a metric between two documents or between a document and a query. Applications of metric learning to these settings include the work of Lebanon (2006) [11]; Lee et al. (2008) [12].

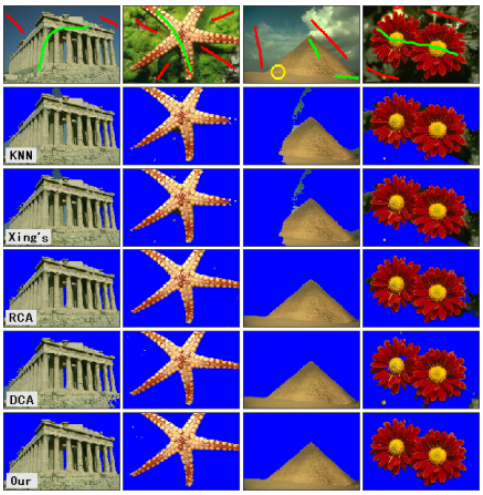

Figure 18: Image Segmentation as application of Metric Learning

Bioinformatics Many problems in bio-informatics involve comparing sequences such as DNA, protein or temporal series. These comparisons are based on structured metrics such as edit distance measures (or related string alignment scores) for strings or Dynamic Time Warping distance for temporal series. Learning these metrics to adapt them to the task of interest can greatly improve the results. Examples include the work of Xiong and Chen (2006) [13]; Saigo et al. (2006) [14] to name a few.

References

[1] E. Xing, A. Ng, M. Jordan, and S. Russell *Distance metric learning with application to clustering with side-information* in Proc. NIPS, 2003.

[2] Flueck, Alexander J. 2005. *ECE 100*[online]. Chicago: Illinois Institute of Technology, Electrical and Computer Engineering Department, 2005 [cited 30 August 2005]. Available from World Wide Web: (http://www.ece.iit.edu/ flueck/ece100).

[3] Blake Shaw, Bert C. Huang, and Tony Jebara. *Learning a Distance Metric from a Network. In Advances in Neural Information Processing Systems (NIPS)* 24, pages 18991907, 2011.

[4] Matthew E. Taylor, Brian Kulis, and Fei Sha. *Metric learning for reinforcement learning agents. In Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 777784, 2011.

[5] Brian McFee, Luke Barrington, and Gert R. G. Lanckriet. *Learning Content Similarity for Music Recommendation. IEEE Transactions on Audio, Speech and Language Processing (TASLP)*, 20(8):22072218, 2012.

[6] Remi Lajugie, Sylvain Arlot, and Francis Bach. *Large-Margin Metric Learning for Constrained Partitioning Problems. In Proceedings of the 31st International Conference on Machine Learning (ICML)*, 2014.

[7] Fei-Fei Li and Pietro Perona. *A Bayesian Hierarchical Model for Learning Natural Scene Categories. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 524531, 2005.

[8] Thomas Mensink, Jakob J. Verbeek, Florent Perronnin, and Gabriela Csurka. Metric Learning for Large Scale Image Classification: Generalizing to New Classes at Near-Zero Cost. In Proceedings of the 12th European Conference on Computer Vision (ECCV), pages 488501, 2012.

[9] Andrea Frome, Yoram Singer, Fei Sha, and Jitendra Malik. *Learning Globally-Consistent Local Distance Functions for Shape-Based Image Retrieval and Classification. In Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 18, 2007.

[10] Matthieu Guillaumin, Jakob J. Verbeek, and Cordelia Schmid. *Is that you? Metric learning approaches for face identification. In Proceddings of the IEEE International Conference on Computer Vision (ICCV)*, pages 498505, 2009b.

[11] Guy Lebanon. *Metric Learning for Text Documents. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 28(4):497508, 2006.

[12] Jung-Eun Lee, Rong Jin, and Anil K. Jain. *Rank-based distance metric learning: An application to image retrieval. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[13] Huilin Xiong and Xue-Wen Chen. *Kernel-based distance metric learning for microarray data classification. BMC Bioinformatics*, 7:299, 2006.

[14] Hiroto Saigo, Jean-Philippe Vert, and Tatsuya Akutsu. *Optimizing amino acid substitution matrices with a local alignment kernel. Bioinformatics*, 7(246):112, 2006.