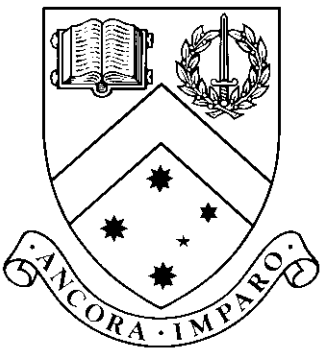
Clayton School of Information Technology

Monash University
Clayton VIC 3800, Australia



Technical Report No. 2010/256

A Simple Gradient-based Metric Learning Algorithm for Object Recognition

Nayyar A. Zaidi, David McG. Squire, and David Suter

Nayyar.Zaidi@monash.edu, David.Squire@monash.edu

Abstract

The Nearest Neighbor (NN) classification/regression techniques, besides their simplicity, is one of the most widely applied and well studied techniques for pattern recognition in machine learning. Their only drawback is the assumption of the availability of a proper metric used to measure distances to k nearest neighbors. It has been shown that K-NN classifier's with a right distance metric can perform better than other sophisticated alternatives like Support Vector Machines (SVM) and Gaussian Processes (GP) classifiers. That's why recent research in k-NN methods has focused on metric learning i.e., finding an optimized metric. In this paper we have proposed a simple gradient based algorithm for metric learning. We discuss in detail the motivations behind metric learning, i.e., error minimization and margin maximization. Our formulation is different from the prevalent techniques in metric learning where goal is to maximize the classifier's margin. Instead our proposed technique (MEGM) finds an optimal metric by directly minimizing the mean square error. Our technique not only resulted in greatly improving k-NN performance but also performed better than competing metric learning techniques. We also compared our algorithm's performance with that of SVM. Promising results are reported on major faces, digits, object and UCIML databases.

1 Introduction

Nearest neighbor methods for pattern recognition have been proven to be very useful in machine learning. Despite the simplicity, their performance is comparable to other sophisticated classification and regression techniques like Support Vector Machines (SVM) and Guassian Process (GP) and they have been widely applied for a huge variety of problems. Computer vision research has benefited greatly from advancements in nearest neighbor methods, for example some state-of-the-art techniques for object recognition are based on nearest neighbor analysis (1),(2). For a given query point, a nearest neighbor classifier works by assigning it the label of the majority class in its neighborhood.

It is self evident that k-NN classifier's simplicity leads to one of its major pros. A k-NN classifier deals with multi-class classification scenario effortlessly. On the other hand we need one-versus-one and one-versus-all techniques to deal with multi-class scenario in case of binary classifiers like SVM. This makes them computationally expensive. As k-NN classification involves no training they are computational efficient. But still the effectiveness of k-NN methods stems from their asymptotic properties. The asymptotic results in (3), (4), (5) suggests that a 1-NN method based on a simple Euclidean distance will perform well provided the number of training samples is not too small. And 1-NN will achieve the performance of a Bayes Optimal classifier as the number of training data becomes very large. These asymptotic results are based on the fact that bias in the prediction of function $f(x)$ becomes vanishingly small if the number of training data N is large with few features p i.e., $N >>> p$. But typical machine learning data has large number of features and data required to validate these asymptotic results is humongous which is not feasible. This is also known as Curse-of-Dimensionality (COD). Also due to COD most of the data points are very far apart and k-NN neighborhoods are no longer 'local', refer to section 2.5 in (6). Therefore we need to modify distances in high dimensions to alleviate the COD, reduce bias and make neighborhood local. This calls for tuning a metric and hence metric learning.

As discussed, the performance of a nearest neighbor classifier depends critically on two major factors: (a) the distance metric used and (b) size of the neighborhood (specified by K which denotes the number of nearest neighbors). The value of K controls the Mean Square Error (MSE) which is defined as MSE = $\text{bias}^2 + \text{var}$. Small K implies small bias but high variance and vice-versa. Since K is specified in terms of the number of nearest neighbors of a query point x which implicitly depends on a distance measure, MSE can

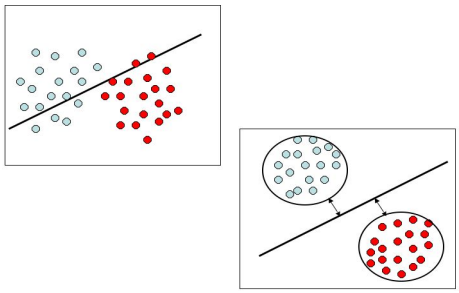


Figure 1: Contrived example demonstrating the impact of metric on margin

be controlled by estimating a distance metric (a metric is generally specified through a norm and a positive semi-definite matrix). Typically we estimate the inverse square roots of metric. That is we learn a matrix parameterizing the linear transformation of the input space such that in the transformed space k-NN performs well. If we denote such transformation by a matrix A, we are effectively learning a metric defined by A^TA such that $d(x, y) = (x - y)A^TA(x - y) = (Ax - Ay)^T(Ax - Ay)$.

In the current research on nearest neighbor methods, a dichotomy exists for metric learning methods in terms of their goals. Most 'Metric Learning' algorithms aim at finding a metric that results in small intra-class and large inter-class distances (7), (8), (9), (10), (11), (12). This results in maximizing the margin [1]. Figure 1 depicts a simple contrived example of data belonging to two classes represented by red and blue dots. As can be seen that the classes are linearly separable and a hyper-plane is represented by a dark black line. In this scenario, margin can be maximized in two ways, either we modify the hyper-plane to better fit the training data or we transform the training data to maximize margin with respect to a certain hyper-plane. The later has been the goal of most metric learning algorithms. SVMs on the other hand optimizes margin by finding an optimal hyper-plane. They are designed to minimize empirical risk with a bound on generalization error. Metric learning can also be used to minimize empirical risk i.e., maximizing the margin by modifying hyper-plane. Such a kind of strategy has been introduced in (13) where metric learning has been employed as bias reduction strategy and reducing MSE to better fit training data.

In this paper we have presented a novel metric learning algorithm with the goals of maximizing the margin by reducing MSE directly. We propose a simple MSE gradient minimization (MEGM - Mean square Error's Gradient Minimization) approach for improving the performance of k-NN neighbor classifier. Our method is based on gradient descent of MSE objective function. We compared MEGM's performance with other metric learning approaches for margin maximization for example neighborhood component analysis (NCA). As will be shown in section 4 our method not only results in significant improvement in the performance of k-NN classifier but also outperforms other metric learning algorithm on most data-sets. As we will discuss in section 5, unlike SVM we minimize the empirical risk only. We have not catered for generalization in our algorithm. But in our experiments we did not experience any over-fitting. The inclusion of generalization term can be introduced in our framework and has been left as a future work.

Rest of the paper is organized as follows: we will discuss related work in section 2. Our proposed MEGM algorithm is described in detail in section 3. Detail description about experimental setup and comparison results on UCIML, face, object and digits data-sets

[1] The margin of a point is defined as the distance between the point and the closest point on the classification boundary

are given in section 4. We conclude in section 5 with pointers on future work.

2 Related Work

Our proposed algorithm MEGM is very close in nature to (14) where a gradient based technique is used for selecting relevant features. That is, only diagonal terms of the covariance matrix are estimated. In our method we learn a full covariance matrix rather than estimating only diagonal terms. Thats why MEGM is superior to technique proposed in (14).

The other notable techniques for metric learning are LMNN (15), RCA (16) and NCA (7). Relevant Component Analysis (RCA) (16) constructs a Mahalanobis distance metric from a weighted sum of in-class covariance matrices. It is similar to Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) in its reliance on second order statistics. Large Margin Nearest Neighbor (LMNN) algorithm in (15) is posed as a convex problem, and thus the reach of the global solution is guaranteed. However, a special optimization solver is needed for efficient implementation.

Neighborhood Component Analysis (NCA) (7) maximizes margin by minimizing the probability of error under stochastic neighborhood assignment. In particular each point i selects another point j as its neighbor with some probability p_{ij}, and inherits its class labels from the point it selects. p_{ij} is defined as a softmax over Euclidean distances in the transformed space, parameterized by A:

$$p_{ij} = \frac{-\exp(\|Ax_i - Ax_j\|^2)}{\sum_{k \neq i} \exp(-\|Ax_i - Ax_k\|^2)} \tag{1}$$

NCA maximizes the p_{ij} in above equation by finding an optimal A matrix. That is the probability of the number of points correctly classified is maximized.

The comparison of our proposed algorithm (MEGM) with NCA has been a major motivation of this work. Though NCA is sound in theory, our empirical results in section 4 suggests that MEGM performs better than NCA on most data-sets. We will mention in section 5 about an approach to combine both MEGM and NCA to improve MEGM's generalization capacity.

3 Approach

In a typical regression setting, an unknown function $f : R^D \rightarrow R$ is predicted from the training data $\{(\vec{x}_1, y_1), (\vec{x}_2, y_2), ...(\vec{x}_N, y_N)\}$, where $\vec{x}_i$ is a data point and y is the corresponding target value. The predicted function $\hat{f}$ is chosen to be the one that minimizes some loss function such as 'mean squared error' (MSE) etc. The MSE for a data set containing N of points is given in the following equation:

$$\text{MSE}(\hat{f}) = \sum_{i=1}^{N} (f(\vec{x}_i) - \hat{f}(\vec{x}_i))^2 \tag{2}$$

For classification task having T classes we can replace above error function as:

$$\text{MSE}(\hat{y}) = \sum_{t=1}^{T} \sum_{i=1}^{N} (y_{ti} - \hat{y}_{ti}) \tag{3}$$

where $\hat{y}_i$ denotes the predicted probability of point $\vec{x}_i$ and y_i denotes the true label (either 0 or 1) of point $\vec{x}_i$. For brevity we have denoted $\hat{y}(\vec{x_{ti}})$ with $\hat{y}_{ti}$ and $y(\vec{x_{ti}})$ with y_{ti}. In the

following discussion we will assume that there are only two classes to make our derivations simple. For any query point $\vec{x}_i$, nearest neighbor methods work by predicting the value $\hat{y}_i$ by considering the labels of its k nearest neighbors. In order to have a smooth boundary, each neighbor votes for the query label based on its distance from the query point (refer to (6) for details). Equation 4 shows the Nadaraya-Watson kernel for regression:

$$\hat{y}(\vec{x}) = \frac{\sum_j y_j V_j}{\sum_j V_j} \tag{4}$$

The vote V_j casted by each label around the query point $\vec{x}$ is usually chosen to be a function that decays exponentially as the distance from the query point increases, for example a Gaussian kernel:

$$V_j = \exp\left(\frac{-\mathrm{d}(\vec{x}, \vec{x_j})}{2\sigma^2}\right) \tag{5}$$

Determining votes using equation 5 assumes a well defined distance measure. This assumption, as discussed in the previous section, is not always true, due to the COD and irrelevant features, and can lead to bad results. $\mathrm{d}(\vec{x}, \vec{x_j})$ in equation 5 can be replaced by a more general metric: that is $\mathrm{d_L}(\vec{x}, \vec{x_j})$. If $L = A^T A$, then $\mathrm{d_L}(\vec{x}, \vec{x_j}) = (A\vec{x} - A\vec{x_j})^T(A\vec{x} - A\vec{x_j})$. Since MSE is a function of $\hat{y}$ and $\hat{y}$ depends on $||\vec{x} - \vec{x}_j||_L^2$, MSE can be minimized by selecting an optimal value of L. In other words, a change in the L induces a change in the distance, which can alter the votes. This alteration in the votes V_j triggers a change in $\hat{y}$ affecting the MSE. It is more helpful to optimize A rather than L, because optimization for L requires to fulfill semi-positive constraint which is expensive to maintain. Obviously trying all possible values of A is not feasible. Some sort of search mechanism is required to find an optimal value of A. Votes V_j in equation 5 can be replaced by W_j as:

$$W_j = \exp\left(\frac{-\|A\vec{x} - A\vec{x_j}\|_2^2}{2\sigma^2}\right) \tag{6}$$

The proposed gradient based technique (MEGM) is based on a gradient descent algorithm to minimize MSE (lets denote by E_A). The gradient E_A is evaluated to find an optimal A matrix. Convergence to the global minimum is not guaranteed. The risk of local minima can be reduced by running the algorithm several times and choosing the output with minimum error E_A. The gradient of E_A with respect to matrix A is:

$$\frac{\partial E}{\partial A} = (y_i - \hat{y}_i)\frac{1}{\sum_j W_j}\sum_j (y_j - \hat{y}_j))\frac{\partial W_j}{\partial A} \tag{7}$$

The size of the gaussian kernel centered at the query point (σ in equation 6) is set proportional to the distance of the k nearest neighbors. Generally the average distance of half of the nearest neighbors is used, as this measure is more stable under a varying distance metric:

$$\sigma^2 = \frac{1}{2}\frac{1}{P}\sum_{p=1}^{P}\|\vec{x} - \vec{x}_p\|^2 \quad \text{where } P = k/2 \tag{8}$$

$\frac{\partial W_j}{\partial A}$ in equation 7 can be derived as:

$$\frac{\partial W_j}{\partial A} = 2W_j A(\vec{x} - \vec{x}_j)(\vec{x} - \vec{x}_j)^T \tag{9}$$

Combining equations 7 and 9 we can write the gradient of E_A with respect to matrix A as:

$$\frac{\partial E}{\partial A} = 2A(y_i - \hat{y}_i)\frac{1}{\sum_j W_j}\sum_j (y_j - \hat{y}_j)W_j(\vec{x} - \vec{x}_j)(\vec{x} - \vec{x}_j)^T \tag{10}$$

Equation 10 represents the gradient of the error function with respect to matrix A which is minimized to get an optimal A. The Polack-Ribiere flavour of conjugate gradients is used to compute search directions, and a line search using quadratic and cubic polynomial approximations and the Wolfe-Powell stopping criteria is used together with the slope ratio method for guessing initial step sizes.

4 Experimental Results

In this section we present results on various machine learning databases from UCIML repository (17), face, object and digit databases. Database details such as number of data, features and classes as well as experimental setup details such as number of data per category used for training and testing and the number of eigen vectors used are given in table 1. For faces (yalefacs, yalefaceB, AT&T, caltechfaces, caltechfacesB), digit (USPS), Object (Coil100) and Isolet data-set, not only we have compared MEGM's performance with standard k-NN classifier but also with SVM. MEGM's results are also compared with other metric learning approaches like NCA, RCA and LMNN.

To obtain SVM results, multi-class SVM with gaussian kernel is used. C parameter for SVM are tuned through cross-validataion. σ parameter is set to the average distance of k nearest neighbor as this approach has been shown to be more efficient for object databases as shown in (18). For obtaining multi-class SVM results, one-versus-all strategy is employed. Each experiment (except for UCIML databases) is repeated 10 times and mean results and standard deviations are presented.

Faces, USPS, Coil100 and Isolet data-sets are also pre-processed for efficiency. Pre-processing images using PCA is a common approach in object recognition research to reduce dimensionality. This results in vastly reduced computational cost. In our experiments, the results are obtained by reducing the dimensionality of data set by projecting data on first few eigenfaces. Number of eigenfaces used for each database is given in table 1. We have not investigated results without pre-processing images, that is using gray-scale values as features due to computational cost. No pre-processing is done for UCIML databases. The size of neighborhood (k) as discussed in section 3 is consistently set equal to the log2(cardinality of data set) for all databases.

Database	#Data	#Features	#Classes	PCA	#Train/Class	#Test/Class
yalefaces	165	77760	15	50	4	7
yalefacesB	5850	307200	10	20	10	20
caltechfaces	435	47500	29	30	5	10
caltechfacesB	435	47500	2	30	50	100
AT&Tfaces	400	10304	40	50	5	5
Coil100	7200	16384	100	50	10	10
USPS	9298	256	10	50	20	10
Isolet	6238	617	26	30	20	10

Table 1: Details of Face, Object and Digit Databases used for classification

To obtain the final classification results, one nearest neighbor (1-NN) classification is used. As mentioned in section 3, since both NCA and MEGM suffers from local minima problems, some care has to be taken to make sure that it does not effect results. For UCIML and other databases, we run MEGM and NCA thrice with different training data samples and selected the best results. In order to make sure that our results are not biased to NCA and MEGM due to this procedure, reported results for all other techniques for example k-NN, SVM, LMNN and RCA are computed this way. That is each method is run

thrice using different training samples and best results are selected in each run. Percentage correctness rate is reported in case of faces, USPS, Isolet and Coil100 database, whereas error rates are reported for UCIML data-sets. The bar graphs are displayed with matlab 'hot' colormap.

4.1 UCIML Repository Databases

To test the performance of MEGM with other metric learning methods, we selected major UCIML databases. The error rate of each method for different databases is shown in figure 2. The number of data, features and classes for each database is reported in the title. Error rate of each method is obtained using 40 rounds of 2 fold cross-validation. Prior to training all features were normalized to have zero mean and a unit variance.

As can be seen MEGM not only improved k-NN classification performance but in most cases resulted in better performance than other metric learning techniques like NCA, RCA and LMNN. MEGM outperforms other methods on Balance and Hayesroth databases. Also it performed marginally better than other techniques on Credit-screeing, Dermatology, Sonar, Statlog-heart, vowel and Monks2. On Monks1 and Monks3 both MEGM and NCA performs equally well and error rate is close to zero for both these methods.

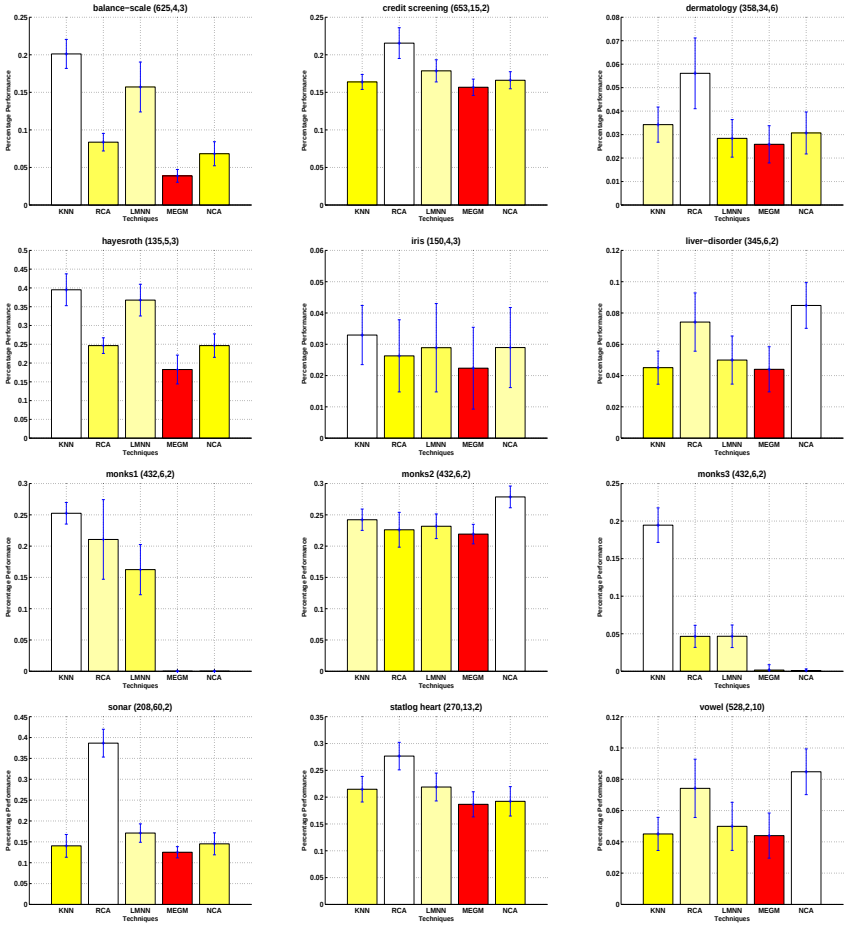


Figure 2: Error rate comparison of various techniques on UCIML databases

Though MEGM performed better than other approaches on most databases as shown in figure 2, NCA performance is also noteworthy especially on balance, monks1 and statlog-heart. It performed marginally better than other techniques on Ionosphere, Housevote and Hepatitis as shown in figure 3.

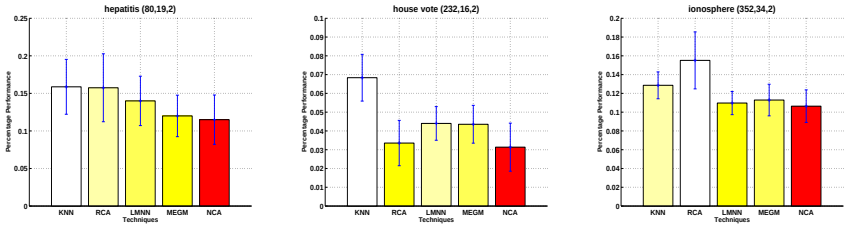


Figure 3: Error rate comparison of various techniques on UCIML databases, NCA performs best on these data-sets

4.2 Face Databases

We have experimented with 5 face databases. Yalefaces, YalefacesB, AT&T and caltechfaces were used to compare the performance of MEGM with margin based metric learning algorithm (NCA) and other classification techniques like k-NN and SVM. Details of these databases are given in table 1. Yalefaces, YalefacesB, AT&T are well-known in face recognition research. The Yalefaces database (19) contains 165 grayscale images of 15 individuals. There are 11 images per subject, one per different facial expression or configuration: center-light, with glasses, happy, left-light, with no glasses, normal, right-light, sad, sleepy, surprised, and wink. The YalefacesB database (20) contains 5760 single light source images of 10 subjects each seen under 576 viewing conditions (9 poses x 64 illumination conditions). For every subject in a particular pose, an image with ambient (background) illumination was also captured. AT&T database (21) has ten different images of each of 40 distinct subjects. For some subjects, the images were taken at different times, varying the lighting, facial expressions (open, closed eyes, smiling, not smiling) and facial details (glasses, no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement)

Caltechfaces and CaltechfacesB constitutes images from face category in Caltech-101 object database (22). Caltech-101 face category has 435 images of around 20 people. Some example images are shown in figure 4. Caltechfaces database in table 1 is based on splitting Caltech-101 face category in 20 categories, each belonging to different person. On the other hand, CaltechfacesB in table 1 is based on splitting Caltech-101 face category in two classes only, male and female.

Figure 4: Example images from caltechfaces and caltechfacesB

The SVM results are obtained by optimizing over C parameter. C is searched from the values: $1, 10, 100, 1000$. The comparative results on face databases are shown in figure 5 where percentage performance is reported. As can be seen that MEGM results in significant improvements in k-NN classification and outperforms NCA on all databases. Though performance gain of MEGM over SVM is not substantial, given that SVM is trained with C parameter optimized through cross-validation, our results are encouraging.

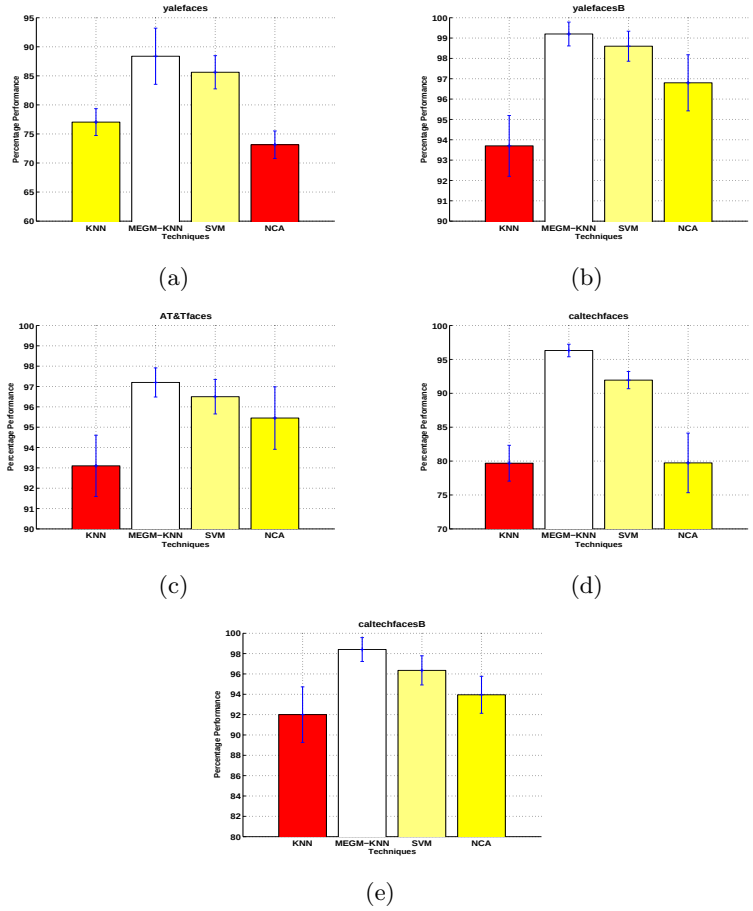


Figure 5: Percentage performance of different techniques on different face databases

4.3 USPS Database

The USPS digits data was gathered at the Center of Excellence in Document Analysis and Recognition (CEDAR) at SUNY Buffalo, as part of a project sponsored by the US Postal Service (23). Sample images from USPS digit data-set is shown in figure 6. Results on

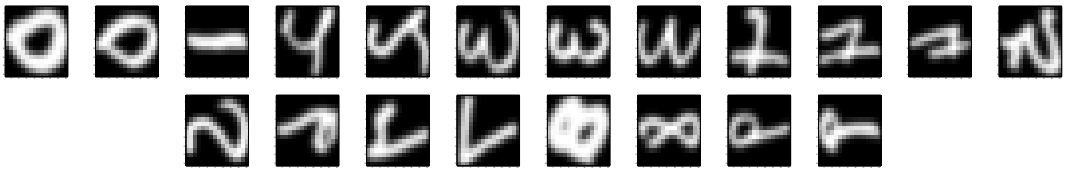
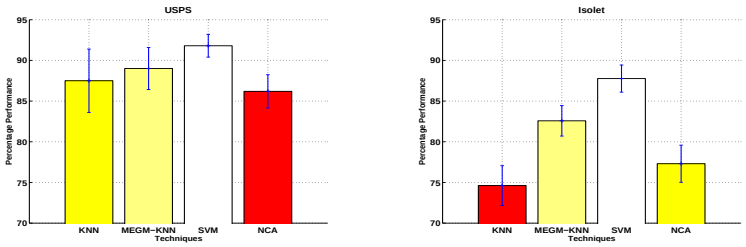


Figure 6: Example images from USPS database

USPS dataset are shown in figure 7(a). We did not find any noticeable increase in k-NN performance on USPS database. Though MEGM marginally improves k-NN performance, SVM performs the best. NCA on the hand did not result in any performance gain. SVM results are optimized over C values of $\{1, 10, 100, 1000\}$.

4.4 Isolet Database

Isolet (Isolated Letter Speech Recognition) (17) consists of 26 categories. 120 subjects spoke the name of each letter of the alphabet twice. Hence, we have 52 training examples



(a) (b)

Figure 7: Percentage performance of different techniques on USPS and Isolet databases

Figure 8: Sample images from Coil Object database.

from each speaker. Results on Isolet database is shown in figure 7(b). MEGM, SVM and NCA resulted in k-NN performance on Isolet database. Like USPS, SVM outperforms our method on this database. MEGM performs better than NCA. Similar to faces and USPS datasets, SVM results are optimized over C values of $\{1, 10, 100, 1000\}$.

4.5 Coil100 Database

Coil100 object database consists of 100 object categories (24). Each category has 72 images taken at different angle of the object. Figure 7 shows results on Coil100 database. Using MEGM and NCA we got some improvement over k-NN performance. SVM did not perform well on this database with only 81% success rate as compared to k-NN method of 86% success rate. NCA also deteriorated k-NN results. The SVM results were obtained by searching for C value from $\{1, 10, 100, 1000\}$.

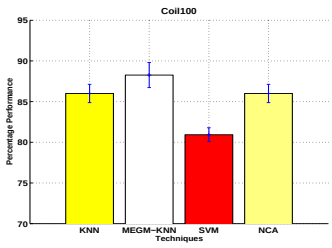


Figure 9: Percentage performance of different techniques on Coil100 object databases

5 Conclusion

The main pro of our proposed MEGM algorithm is its simplicity. As discussed, MEGM minimizes MSE gradient using a simple gradient descent algorithm. MEGM improves k-NN classification by learning a data dependent distance metric and performs as well as SVM on most if not all databases. Also It deals with multi-class problems effortlessly as opposed to binary classifiers like SVM where a one-versus-one and one-versus-all strategy is used. As discussed, SVM's training and testing is computationally expensive, for example it takes very long time to train and test an SVM classifier on Coil100 database. As training involves training 100 classifiers and for classifying an image, all classifiers vote for prediction. On the other hand, once a metric is learnt using MEGM, a simple nearest neighbor classification is required. In typical object recognition tasks where number of classes are very large, nearest neighbor methods should be preferable for their computational efficiency. Therefore k-NN methods equipped with a proper distance metric (for example, one trained with MEGM) can be extremely useful.

A drawback of MEGM includes local minima problem. Standard approaches to avoid local minima are to be used. Also one is tempted to think of over-fitting if the objective function is only MSE. In this work we did not encounter any over-fitting. As a future work we are investigating to modify our objective function to include a generalization term, that is penalize large changes in A matrix to avoid over-fitting. We are currently investigating to combine MEGM's and NCA's objective function to improve our results. As in this study, MEGM which is based on the minimization of MSE resulted in better performance than NCA and other metric learning algorithms which maximizes margin explicitly, a natural extension to the proposed method is to combine the two approaches. That is learn a metric by simultaneously maximizing the margin and minimizing the MSE. The objective functions of MEGM and NCA is combined in the following equation:

$$E_A = \sum_{i=1}^{N} \left(y_i - \exp\left(\frac{-\|A\vec{x} - A\vec{x_j}\|_2^2}{2\sigma^2} \right) \right) + \left(\frac{\exp(\|Ax_i - Ax_j\|^2)}{\sum_{k \neq i} \exp(-\|Ax_i - Ax_k\|^2)} \right) \tag{11}$$

We are investigating gradient based methods to optimize for A in equation 11. A simple gradient based strategy as employed in MEGM can be used. Considering the MEGM results, the combination with NCA can lead to good results.

There has been a lot of work done in adaptive distance metric (25), (26), (27). In adaptive metric learning a separate metric is learnt for each query point. We are currently modifying MEGM to work in such local settings. Training a separate metric for each query point can become computationally expensive. We are investigating clustering techniques to cluster data first and than train a separate metric for each cluster.

In summary, we proposed a simple mean square error's gradient based metric learning algorithm (MEGM) in this paper and showed that MEGM not only results in classification improvement of k-NN classifier but also perform better than other metric learning algorithms. The results are also compared with state-of-the-art classifier, SVM. Results are shown on major UCIML, face, digits and object databases. Our results are encouraging and requires additional investigation to further improve MEGM performance as described above.

References

[1] A. Frome, Y. Singer, and J. Malik, "Image retrieval and classification using local functions," in *NIPS*, 2006.

[2] M. Nilsback and A. Zisserman, "A visual vocabulary for flower classification," in *CVPR*, 2006.

[3] T. Cover, "Rates of convergence for nearest neighbor procedures," in *Inter. Conf. on Systems Sciences*, 1968.

[4] E. Fix and J. Hodges, "Discriminatory analysis - nonparameteric discrimination: consistency properties," Tech Report, Randolph Field Texas, US Airforce School of Aviation Medicine, Tech. Rep., 1951.

[5] R. Snapp and S. Venkatesh, "Asymptotic expansions of the k-nearest neighbor risk," *The Annals of Statistics*, 1998.

[6] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. Springer Series in Statistics, 2001.

[7] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov, "Neighborhood component analysis," in *NIPS*, 2005.

[8] J. Davis and I. Dhillon, "Structured metric learning for high dimensional problems," in *KDD*, 2008.

[9] K. Weinberger, J. Blitzer, and L. Saul, "Distance metric learning for large margin nearest neighbor classification," in *NIPS*, 2005.

[10] B. Sriperumbudar, O. Lang, and G. Lanckriet, "Metric embedding for kernel classification rules," in *ICML*, 2008.

[11] A. Globerson and S. Roweis, "Metric learning by collapsing classes," in *NIPS*, 2005.

[12] E. Xing, A. Ng, M. Jordan, and S. Russell, "Distance metric learning with application to clustering with side-information," in *NIPS*, 2002.

[13] J. Friedman, "Flexible metric nearest neighbor classification," Tech Report, Dept. of Statistics, Stanford University, Tech. Rep., 1994.

[14] D. Lowe, "Similarity metric learning for a variable-kernel classifier," in *NIPS*, 1996.

[15] K. Weinberger, J. Blitzer, and L. Saul, "Distance metric learning for large margin nearest neighbor classification," in *NIPS*, 2006.

[16] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall, "Learning distance functions using equivalence relation," in *ICML*, 2003.

[17] C. Mertz and P. Murphy, "Machine learning repository," 2005. [Online]. Available: http://archive.ics.uci.edu/ml/

[18] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid, "Local features and kernels for classification of texture and object categories: A comprehensive study," in *CVPRW*, 2006.

[19] "The yale face database," 1997. [Online]. Available: http://cvc.yale.edu/projects/yalefaces/yalefaces.html

[20] "The yale face b database," 2001. [Online]. Available: http://cvc.yale.edu/projects/yalefacesB/yalefacesB.html

[21] "The at&t face database," 2002. [Online]. Available: http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html

[22] "Caltech-101 object database," 2006. [Online]. Available: http://www.vision.caltech.edu/Image_Datasets/Caltech101/

[23] J. Hull, "A database for handwritten text recognition research," *IEEE PAMI*, 1994.

[24] S. Nene, S. Nayar, and H. Murase, "Columbia object image library (coil-100)," Technical Report CUCS-006-96, February 1996, Tech. Rep., 1996.

[25] T. Hastie and R. Tibshirani, "Discriminative adaptive nearest neighbor classification," *IEEE transactions on Pattern Analysis and Machine Intelligence*, 1996.

[26] C. Domenciconi, J. Peng, and D. Gunopulos, "An adaptive metric machine for pattern classification," in *NIPS*, 2000.

[27] N. Zaidi, D. Squire, and D. Suter, "BoostML: An adaptive metric learning for nearest neighbor classification," in *PAKDD*, 2010.