Kilian Q. Weinberger,
Fei Sha, and
Lawrence K. Saul

Convex Optimizations for Distance Metric Learning and Pattern Classification

The goal of machine learning is to build automated systems that can classify and recognize complex patterns in data. Not surprisingly, the representation of the data plays an important role in determining what types of patterns can be automatically discovered. Many algorithms for machine learning assume that the data are represented as elements in a metric space. For example, in popular algorithms such as nearest-neighbor classification, vector quantization, and kernel density estimation, the metric distances between different examples provide a measure of their dissimilarity [1]. The performance of these algorithms can depend sensitively on the manner in which distances are measured.

When data are represented as points in a multidimensional vector space, simple Euclidean distances are often used to measure the dissimilarity between different examples. However, such distances often do not yield reliable judgments; in addition, they cannot highlight the distinctive features that play a role in certain types of classification, but not others. For example, consider two schemes for clustering images of faces: one by age, one by gender. Images can be represented as points in a multidimensional vector space in many ways—for example, by enumerating their pixel values, or by computing color histograms. However the images are represented, different components of these feature vectors are likely to be relevant for clustering by age versus clustering by gender. Naturally, for these different types of clustering, we need different ways of measuring dissimilarity; in particular, we need different metrics for computing distances between feature vectors. This article describes two algorithms for learning such distance metrics based on recent developments in convex optimization.

Digital Object Identifier 10.1109/MSP.2010.936013

DISTANCE METRIC LEARNING

Distance metric learning is an emerging subarea of machine learning in which the underlying metric is itself adapted to improve the results of classification and pattern recognition [2]–[5]. Algorithms for distance metric learning attempt to improve on ad-hoc or default choices of distance metrics. In many applications, a simple but effective strategy is to replace Euclidean distances by so-called Mahalanobis distances. A Mahalanobis distance metric computes the distance between vectors $\mathbf{x}$ and $\mathbf{z}$ as

$$d_{\mathbf{M}}(\mathbf{x}, \mathbf{z}) = \sqrt{(\mathbf{x}-\mathbf{z})^{\mathrm{T}}\mathbf{M}(\mathbf{x}-\mathbf{z})}, \quad (1)$$

where the matrix $\mathbf{M} \succcurlyeq 0$ is required to be positive semidefinite. More formally, (1) defines a pseudo-metric. A pseudo-metric is a metric except that $\mathrm{dist}(\mathbf{x}, \mathbf{z}) = 0$ does not imply that $\mathbf{x} = \mathbf{z}$. Mahalanobis distances play an important role in multivariate statistics, where the matrix $\mathbf{M}$ is often estimated from the data's inverse covariance matrix. We will consider other ways to estimate such matrices, while still requiring them to be positive semidefinite. When $\mathbf{M}$ is equal to the identity matrix, (1) reduces to the Euclidean distance metric.

In this article, we describe the problem of distance metric learning as it arises in two popular models of classification. Most importantly, in these models, we show how to formulate the required optimizations for distance metric learning as instances of convex programming. Convex programming is a generalization of linear programming in which linear costs and constraints are replaced by convex costs and constraints. Convex programs can be solved efficiently on modern computers due to recent advances in numerical optimization [6]. Distance metric learning lends itself naturally to convex programming because the constraint $\mathbf{M} \succcurlyeq 0$ in (1) is convex; in particular, the set of positive semidefinite matrices is a convex set.

We have explored convex optimizations for distance metric learning in both nonparametric and parametric models of classification. In the first part of the article, we describe how to learn a distance metric to improve the accuracy of k-nearest neighbor (kNN) classification [7]. Specifically, we show how to learn a linear transformation of the input space that shrinks the distances between nearby examples from the same class and expands the distances between examples from different classes. In the second part of the article, we consider how to model the examples in each class by a multivariate Gaussian distribution [8]. When these distributions are used for multiway classification, we show that recent methods in distance metric learning can yield much better results than maximum likelihood estimation.

METRIC LEARNING FOR NEAREST-NEIGHBOR CLASSIFICATION

First we consider how to learn a distance metric to improve the results of kNN classification. Nearest-neighbor classification is based on access to a labeled set of training examples, consisting of vector-valued inputs $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \Re^d$ and their class labels $y_1, \ldots, y_n \in \{1, \ldots, c\}$. The kNN decision rule classifies unlabeled inputs by the majority label of their kNN in the training set. Naturally, the performance of this algorithm depends strongly on the distance metric used to identify kNNs.

The performance of a kNN classifier can be estimated on the training examples by the leave-one-out error rate. The leave-one-out error rate measures the fraction of training examples that are misclassified based on the majority labels of their kNNs. When computing the leave-one-out error rate, each training example is excluded from the search for its own kNNs; in particular, this search is confined to the remaining $n-1$ inputs in the training set.

With kNN classification in mind, we attempt to find the distance metric that minimizes the leave-one-out error rate. We note that the leave-one-out error rate vanishes if the k nearest neighbors of each input x_i share the same class label y_i. Our algorithm is based on the following intuition. For robust kNN classification, not only should we minimize the leave-one-out error rate, but we should adapt the distance metric so that the nearest neighbors of x_i from different classes lie much farther away than the k nearest neighbors with label y_i. The remainder of this section discusses how to formalize this intuition as a convex optimization.

NOTATION AND TERMINOLOGY

Among the neighbors of each training example x_i, we distinguish between two types: target neighbors and impostors. These different types of neighbors will play competing roles in the convex optimization for learning a distance metric.

The target neighbors of an input x_i are the k inputs that we desire to be the k nearest neighbors of x_i. In many applications, we can identify these inputs using domain knowledge; in the absence of such knowledge, we use the Euclidean distance metric to identify the k nearest inputs with the same class label. As notation, we use the relation $j \rightsquigarrow i$ to denote that x_j is one of the k target neighbors of x_i. However the target neighbors are identified, this relation implies that $y_j = y_i$.

For kNN classification to succeed, each input's target neighbors should be closer than other inputs with different class labels. To make this notion precise, we define the following set of triples:

$$T = \{(i,j,\ell) \mid i,j,\ell \in \{1,\ldots,n\},\ j \rightsquigarrow i,\ y_i \neq y_\ell\}. \tag{2}$$

THE GOAL OF MACHINE LEARNING IS TO BUILD AUTOMATED SYSTEMS THAT CAN CLASSIFY AND RECOGNIZE COMPLEX PATTERNS IN DATA.

The set T contains all triples (i,j,ℓ) of inputs x_i, target neighbors x_j, and inputs of different classes x_ℓ. For robust kNN classification, we require that for each input x_i, the distances to differently labeled inputs x_ℓ exceed the distances to target neighbors x_j by a large margin. This will be true if each triple in T satisfies the constraint

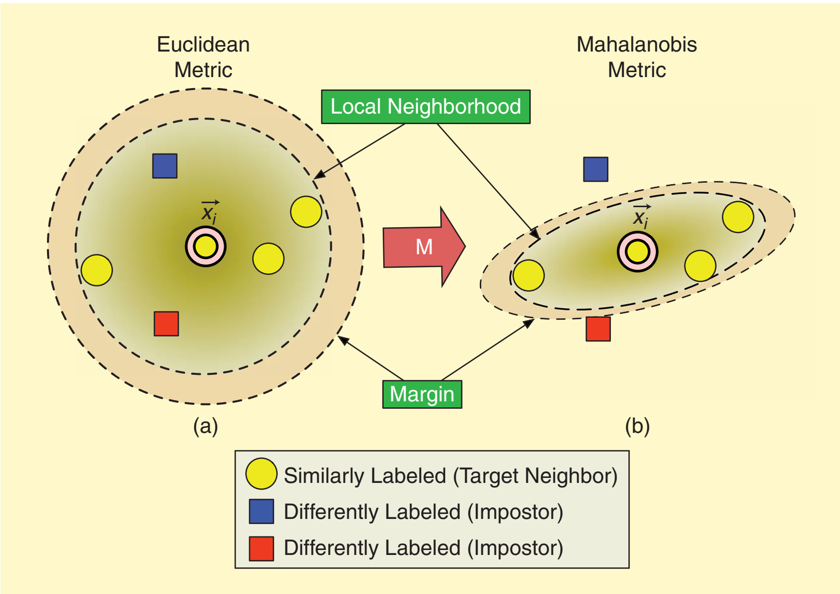
$$\forall (i,j,\ell) \in T\ d^2_{\mathrm{M}}(\mathrm{x}_i, \mathrm{x}_\ell) \geq d^2_{\mathrm{M}}(\mathrm{x}_i, \mathrm{x}_j) + 1, \tag{3}$$

where the Mahalanobis metric $\mathbf{M} \succcurlyeq 0$ is used to compute distances. The constraint in (3) requires that distances to differently labeled inputs exceed those to target neighbors by a margin of (at least) one unit of squared distance. Note that the size of the margin is relative to the scale of the $\mathbf{M}$, which appears linearly in the other terms of this constraint.

The impostors of an input x_i are differently labeled inputs x_ℓ that violate this constraint for some target neighbor x_j. Impostors do not share the same label y_i, but they lie within one unit of distance from the hyper-ellipsoid centered at x_i with radius $d_{\mathrm{M}}(\mathrm{x}_i, \mathrm{x}_j)$. Thus, they identify examples that are likely to cause misclassifications using the kNN decision rule.

CONVEX OPTIMIZATION

With this notation and terminology, we now consider how to learn a distance metric that satisfies many, if not all, of the constraints in (3). Note that the distance metric $\mathbf{M}$ defines the shape of the hyper-ellipsoid that encloses each input and its target neighbors. Intuitively, the goal of distance metric learning is to change the shape of this ellipsoid so that it includes the target neighbors but excludes the impostors. Figure 1



[FIG1] LMNN illustrated on a schematic example. (a) Initially, under the Euclidean distance, the data point x_i has two impostors in its neighborhood. (b) The Mahalanobis metric transforms the space such that inputs with similar label are pulled closer and inputs with different labels are pushed apart. In particular, dissimilar inputs are separated from the local neighborhood by a large margin.

illustrates how a Mahalanobis metric can achieve this goal where a Euclidean metric fails.

To achieve this goal in practice, we optimize the distance metric in (3) over the space of positive semidefinite matrices $\mathbf{M} \succcurlyeq \mathbf{0}$. The optimization is based on two distinct subgoals: i) to shrink the distances between inputs and target neighbors and ii) to minimize the number of impostors that violate the constraints in (3). Specifically, we consider the objective

$$\min_{\mathbf{M}\succcurlyeq 0}\Bigg\{\sum_{j\rightsquigarrow i} d_{\mathbf{M}}^2(\mathbf{x}_i,\mathbf{x}_j) + C\sum_{(i,j,\ell)\in T}\max[0, d_{\mathbf{M}}^2(\mathbf{x}_i,\mathbf{x}_j) + 1 - d_{\mathbf{M}}^2(\mathbf{x}_i,\mathbf{x}_\ell)]\Bigg\}. \tag{4}$$

The first term in (4) penalizes the squared distances between inputs and target neighbors, while the second term accumulates the total amount by which the constraints in (3) are violated. The constant $C > 0$ controls the tradeoff between these two terms.

We can cast the optimization in (4) as an instance of semidefinite programming [6]. Semidefinite programs (SDPs) are linear programs with an additional (convex) constraint that a matrix whose elements are linear in the unknown variables must be positive semidefinite. To cast (4) as an SDP, we recognize that the squared distances $d_{\mathbf{M}}^2(\mathbf{x}_i, \mathbf{x}_j)$ are linear in the elements of $\mathbf{M}$ and that the matrix $\mathbf{M}$ is constrained to be positive semidefinite. We also introduce nonnegative slack variables $\xi_{ij\ell} \geq 0$ to monitor the violations of the constraints in (3). Then, the objective in (4) can be rewritten as

$$\min_{\mathbf{M},\{\xi_{ij\ell}\}}\Big\{\sum_{j\rightsquigarrow i}(\mathbf{x}_i-\mathbf{x}_j)^\top\mathbf{M}(\mathbf{x}_i-\mathbf{x}_j) + C\sum_{(i,j,\ell)\in T}\xi_{ij\ell}\Big\} \text{ subject to:}$$

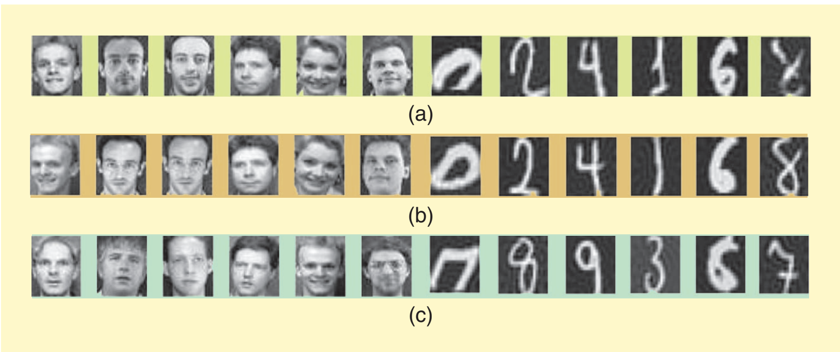
1) $(\mathbf{x}_i-\mathbf{x}_\ell)^\top\mathbf{M}(\mathbf{x}_i-\mathbf{x}_\ell) - (\mathbf{x}_i-\mathbf{x}_j)^\top\mathbf{M}(\mathbf{x}_i-\mathbf{x}_j) \geq 1-\xi_{ij\ell}$ for all $(i,j,\ell)\in T$
2) $\xi_{ij\ell} \geq 0$ for all $(i,j,\ell)\in T$
3) $\mathbf{M} \succcurlyeq 0$.

SDPs can be solved efficiently by interior point algorithms [6]. For this particular problem, however, we have mainly used projected subgradient descent methods on the objective in (4). More details can be found in our earlier work [7].

The above SDP for distance metric learning has many similarities to the quadratic program for large margin classification in support vector machines [9]. Due to these similarities, we refer to the approach in this section as large-margin nearest-neighbor (LMNN) classification.

EXPERIMENTAL RESULTS

We experimented with LMNN classification on problems in face recognition and handwritten digit recognition. Figure 2 visualizes the effect of distance metric learning on these tasks, showing how different nearest neighbors are identified using Mahalanobis versus Euclidean distances. On both tasks, the kNN classification was significantly improved by learning a Mahalanobis distance metric. Here, we briefly summarize the main results; more details can be found in the original study [7]. For optimal results, the neighborhood size k should be set by cross validation. As the algorithm is not particularly sensitive to the neighborhood size, we arbitrarily set $k = 3$. To avoid overfitting, we applied early stopping with a 30% hold-out set.



[FIG2] The effect of distance metric learning on kNN classification of images of faces and handwritten digits. In each task, LMNN learns a Mahalanobis distance metric that leads to more accurate kNN classification. (a) Shows images from the test sets. (b) Shows nearest-neighbor images that are correctly identified by Mahalanobis distance but not by Euclidean distance. (c) Shows nearest-neighbor images that are mistakenly identified by Euclidean distance but not by Mahalanobis distance.

For face recognition, we experimented with images from the Olivetti face recognition data set (available at http://www.uk.research.att.com/facedatabase.html). This data set contains 400 gray scale images of 40 subjects in ten different poses. For kNN classification, we preprocessed the images by downsampling and projecting them into the subspace spanned by the leading 200 principal components. Using these compressed images, we created training and test sets by randomly sampling seven images of each subject for training and three images for testing. We evaluated LMNN classification on 100 random splits of the data in this way. The goal of learning was to recognize a face from an unseen pose, thus giving rise to a problem in 40-way classification. By learning a distance metric on this task, we reduced the average kNN classification error rate $(k = 3)$ from 6% using Euclidean distances to 3.3% using Mahalanobis distances.

For digit recognition, we experimented with images from an extensively benchmarked data set of handwritten digits (available at http://yann.lecun.com/exdb/mnist). For kNN classification, we deskewed the original 28×28 gray scale images, then reduced their dimensionality by projecting them onto their leading 164 principal components (enough to capture 95% of their overall variance). We learned a Mahalanobis distance metric on the 60,000 images in the training set. On the images in the test set, this procedure reduced the kNN classification error rate $(k = 3)$ from 2.4% using Euclidean distances to 1.7% using Mahalanobis distances.

EXTENSIONS AND APPLICATIONS

The ideas behind LMNN classification have been extended and applied in many ways. We mention a few examples that follow up closely on the ideas described in this section. Torresani and Lee have shown how to perform LMNN classification in a reproducing kernel Hilbert space [10]; as in support vector machines, the use of nonlinear kernels can lead to significantly fewer classification errors. Weinberger and Saul have extended LMNN to learn multiple, local distance metrics in a globally coordinated manner [7]. Chechik et al. have proposed an online version of LMNN classification that relaxes the positive semidefinite constraint and scales to data sets with millions of inputs [11]. Finally, Tran et al. have applied LMNN classification to video signals for problems in human-activity recognition [12].

METRIC LEARNING FOR GAUSSIAN MIXTURE MODELS

Nonparametric methods such as kNN classification do not make any simplifying assumptions about the underlying distribution of the data. Though kNN classifiers can model highly nonlinear and irregular decision boundaries, this flexibility entails certain costs. For example, in LMNN classification, even after learning the distance metric, it remains necessary to store all the training examples $\{(\mathrm{x}_i, y_i)\}_{i=1}^{n}$. Second, for each new classification, it is necessary to identify the nearest neighbors in the set of all training examples. These costs can be prohibitive for large data sets.

Parametric methods for classification avoid these costs by making simplifying assumptions about the data. In this section, we describe a parametric method that is based on similar large margin constraints as LMNN classification. The method assumes that the data are clustered around representative prototypes and that the class boundaries are determined by distances from these prototypes. We begin by reviewing how the parameters of such models are traditionally estimated.

MAXIMUM LIKELIHOOD ESTIMATION

If we assume that the examples in each class are modeled by a multivariate Gaussian distribution, then we can learn the model parameters for each class using maximum likelihood estimation [1]. In particular, for each class c, we model its distribution of examples as

$$P(\mathrm{x}|c) = \frac{1}{\sqrt{(2\pi)^d |\Sigma_c|}} e^{-\frac{1}{2}(\mathrm{x}-\mu_c)^\top \Sigma_c^{-1}(\mathrm{x}-\mu_c)}, \tag{5}$$

where $\boldsymbol{\mu}_c$ and $\boldsymbol{\Sigma}_c$ are given by their maximum likelihood estimates—specifically, the sample mean and covariance matrix of the examples in class c. These models can be used to classify unlabeled examples by computing the posterior distribution $P(c|\mathrm{x})$ using Bayes' rule. Assigning the most likely class label under this distribution, and working with log probabilities, we obtain a decision rule of the form

DISTANCE METRIC LEARNING IS AN EMERGING SUBAREA OF MACHINE LEARNING.

$$y = \arg\min_c \Big[(\mathrm{x}-\boldsymbol{\mu}_c)^\top \boldsymbol{\Sigma}_c^{-1} (\mathrm{x}-\boldsymbol{\mu}_c)/2 + \log\,(2\pi)^{d/2} |\Sigma_c|^{1/2} - \log P(c) \Big], \tag{6}$$

where $P(c)$ are the prior class probabilities. Note that the last two terms on the right-hand side depend on the class label c but not the input x being classified.

Though maximum likelihood estimation is the simplest way to fit the Gaussian distributions in (5), it does not generally yield the best results when these Gaussian mixture models (GMMs) are used for multiway classification. Next, we describe a learning algorithm inspired by LMNN classification.

PARALLELS TO LMNN

We can develop an alternative model whose decision rule is similar to (6) but whose parameters are learned by optimizing the classification error rate. In particular, suppose that we characterize each class c by a Mahalanobis distance metric $\mathbf{M}_c$, a centroid $\mathbf{r}_c$, and a scalar parameter θ_c. Given these parameters, we consider the decision rule

$$y = \arg\min_c \big[(\mathrm{x}-\mathbf{r}_c)^\top \mathbf{M}_c (\mathrm{x}-\mathbf{r}_c) + \theta_c \big]. \tag{7}$$

The decision rule in (7) is also a form of quadratic discriminant analysis [1]. In this decision rule, the centroid $\mathbf{r}_c$ and metric $\mathbf{M}_c$ are playing the same roles as the sample mean $\boldsymbol{\mu}_c$ and inverse covariance matrix $\boldsymbol{\Sigma}_c^{-1}$, while the scalar θ_c modulates the score of each class in a similar way as the other terms in (5). However we will estimate these parameters differently to obtain a more accurate classifier.

Starting from (7), we can now develop the analogy to LMNN. For a training example x_i to be classified correctly, it must be closer to the centroid $\mathbf{r}_{y_i}$ of class y_i than the centroid of any other class $c \neq y_i$, where closeness is measured by the right-hand side of (7). For robustness, we seek model parameters that correctly classify all training examples by a large margin, as in Figure 3. This will be true if

$$\forall i, c \neq y_i, \quad d^2_{\mathbf{M}_c}(\mathrm{x}_i, \mathbf{r}_c) + \theta_c \geq d^2_{\mathbf{M}_{y_i}}(\mathrm{x}_i, \mathbf{r}_{y_i}) + \theta_{y_i} + 1. \tag{8}$$

The large-margin constraints for GMMs in (8) are analogous to those for LMNN classification in (3). To estimate parameters $(\mathbf{M}_c, \mathbf{r}_c, \theta_c)$ that satisfy as many of these constraints as possible, we consider the objective

$$\min_{\{\mathbf{M}_c \succcurlyeq 0, \mathbf{r}_c \in \Re^d, \theta_c \geq 0\}} \Big\{ \sum_c \mathrm{trace}(\mathbf{M}_c) + C \sum_{i, c \neq y_i} \max\big[0, d^2_{\mathbf{M}_{y_i}}(\mathrm{x}_i, \mathbf{r}_{y_i}) + \theta_{y_i} + 1 - d^2_{\mathbf{M}_c}(\mathrm{x}_i, \mathbf{r}_c) - \theta_c\big] \Big\}. \tag{9}$$

The objective for GMMs in (9) is analogous to the objective for LMNN in (4). The first term regularizes the trace of the Mahalanobis distance metrics, while the second term accumulates the total amount by which the constraints in (8) are violated. The constant $C > 0$ controls the tradeoff between these terms. Without loss of generality, we constrain θ_c to be nonnegative since the decision rule is unaffected by uniform shifts $\theta_c \leftarrow \theta_c + \Delta$.

CONVEX OPTIMIZATION

We can reformulate (9) as an instance of semidefinite programming by making a simple change of variables. In particular, we collect the parameters $(\mathrm{M}_c, \mathrm{r}_c, \theta_c)$ into a single positive semidefinite matrix A_c that is one row and one column larger than the distance metric M_c

$$\mathrm{A}_c = \begin{bmatrix} \mathrm{M}_c & -\mathrm{M}_c\mathrm{r}_c \\ -\mathrm{r}_c^\top \mathrm{M}_c & \mathrm{r}_c^\top \mathrm{M}_c \mathrm{r}_c + \theta_c \end{bmatrix}. \quad (10)$$

The parameters $(\mathrm{M}_c, \mathrm{r}_c, \theta_c)$ are uniquely determined by A_c when A_c is positive definite; when A_c is rank-deficient (as occurs in practice), we cannot recover the original parameters but the scores in (7) remain well-defined. More importantly, in terms of the matrices A_c, the constraints in (8) can be written as

$$\forall i, c \neq y_i,\ \mathrm{v}_i^\top (\mathrm{A}_c - \mathrm{A}_{y_i})\, \mathrm{v}_i \geq 1 \quad \text{where} \quad \mathrm{v}_i = \begin{bmatrix} \mathrm{x}_i \\ 1 \end{bmatrix}. \quad (11)$$

Note that these constraints are linear in the matrices A_c. Analogous to the optimization for LMNN, we introduce nonnegative slack variables $\xi_{\mathrm{ic}} \geq 0$ to monitor the violations of these constraints. Then the optimization in (9) can be rewritten as

$$\min_{\{\mathrm{A}_c,\, \xi_{\mathrm{ic}}\}} \left\{ \sum_c \mathrm{trace}(\mathrm{M}_c) + C \sum_{i, c \neq y_i} \xi_{\mathrm{ic}} \right\}$$

subject to:

1) $\mathrm{v}_i^\top (\mathrm{A}_c - \mathrm{A}_{y_i}) \mathrm{v}_i \geq 1 - \xi_{\mathrm{ic}}$ for all $i, c \neq y_i$
2) $\xi_{\mathrm{ic}} \geq 0$ for all $i, c \neq y_i$
3) $\mathrm{A}_c \succcurlyeq 0$ for all c.

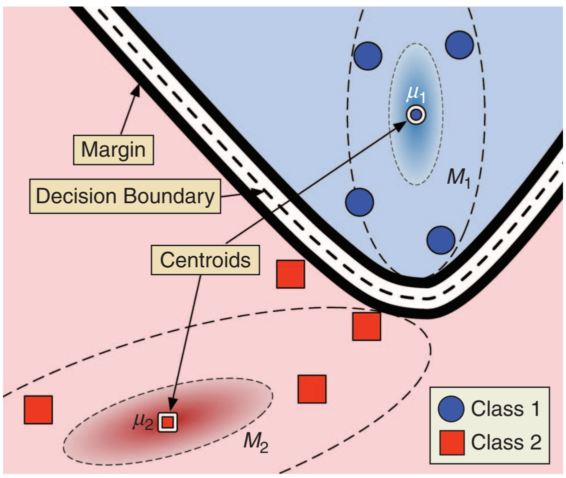
The above optimization is easily recognized as an instance of semidefinite programming. Note that this SDP involves many fewer constraints than the SDP for LMNN classification; thus it scales better to larger data sets. Due to the large margin constraints in (8), models trained in this way are known as large-margin GMMs.

DISTANCE METRIC LEARNING LENDS ITSELF NATURALLY TO CONVEX PROGRAMMING.

EXPERIMENTAL RESULTS

We experimented with large-margin GMMs on problems in handwritten digit recognition [8] and phoneme classification [8]. On the MNIST data set of handwritten digits, using similarly preprocessed images as described above, the large-margin GMM obtained a test error rate of 1.4%. This result was significantly better than the test error rate (4.2%) obtained for the same GMM trained by maximum likelihood estimation. In fact, despite its simplifying parametric assumptions, the large-margin GMM also outperformed the simplest variant of LMNN on this task.

We also trained large-margin GMMs to identify phonemes being articulated in short snippets (25 ms) of speech. The phonemes were classified based on features computed from acoustic waveforms. For this task, we experimented on the TIMIT speech database (available at http://www.ldc.upenn.edu/Catalog/), which contains over 6,000 utterances along with manually aligned phonetic transcriptions. As training examples, we extracted over 1 million phonetically labeled windows of speech. (Note that this large amount of training data precludes a straightforward implementation of LMNN.) On this task, which involved 48-way classification of phonetic categories, we observed significant improvements in performance with large-margin training of GMMs. In particular, on a representative test set with over 50,000 examples, the model in (9) yielded a test-error rate of 36%, whereas the GMM in (5) trained by maximum likelihood estimation yielded a test-error rate of 45%.



[FIG3] The decision boundary in a large-margin GMM for binary classification, consisting of all points with equal Mahalanobis distance to μ_1 and μ_2. The model is trained by penalizing examples that do not lie at least one unit of distance away from the decision boundary.

EXTENSIONS AND APPLICATIONS

Beyond the framework described above, we have extended large-margin GMMs in two important ways [8]. First, we have shown how to train more flexible GMMs that use more than one multivariate Gaussian distribution to model each class of labeled examples. Second, we have shown how to train these GMMs as components of continuous-density hidden Markov models (CD-HMMs) for automatic speech recognition. In both these cases, the optimization remains convex (though it is no longer an instance of semidefinite programming). The extended types of training have yielded gains in performance beyond maximum likelihood estimation and other popular frameworks for parameter estimation in CD-HMMs.

CONCLUSION

In this article, we have described two recent applications of semidefinite programming to problems in machine learning and pattern recognition. Semidefinite programming arises naturally in these problems for two reasons: first, because a positive semidefinite matrix is required to

(continued on page 158)

is also an ill-posed matrix inverse problem. In that case, the solution should be constrained by physical properties of water diffusion in white matter fibers in the brain. Solving these problems usually require a multidisciplinary approach by taking advantages of domain knowledge, be it in biomedical imaging, physics, or physiology with advanced signal processing strategies.

AUTHORS

Kelvin Wong (kwong@tmhs.org) is an assistant professor of electronic engineering in radiology at Weill Cornell Medical College and adjunct assistant professor of pediatrics at Baylor College of Medicine. He is director of the Translational Multimodality Optical Imaging Lab, The Methodist Hospital Research Institute and codirector of the Small Animal Imaging Facility, Texas Children's Hospital.

Chipan Tam (cptam@math.hkbu.edu.hk) is a graduate researcher in the Department of Mathematics at Hong Kong Baptist University.

Michael Ng (mng@math.hkbu.edu.hk) is a professor in the Centre for Mathematical Imaging and Vision, as well as the Department of Mathematics at the Hong Kong Baptist University.

REFERENCES

[1] K. K. Wong, C. P. Tam, M. Ng, S. T. Wong, and G. S. Young, "Improved residue function and reduced flow dependence in MR perfusion using least-absolute-deviation regularization," *Magn. Reson. Med.*, vol. 61, pp. 418–428, Feb. 2009.

[2] J. M. Olivot, M. Mlynash, V. N. Thijs, S. Kemp, M. G. Lansberg, L. Wechsler, R. Bammer, M. P. Marks, and G. W. Albers, "Optimal Tmax threshold for predicting penumbral tissue in acute stroke," *Stroke*, vol. 40, pp. 469–475, Feb. 2009.

[3] M. Ibaraki, E. Shimosegawa, H. Toyoshima, K. Takahashi, S. Miura, and I. Kanno, "Tracer delay correction of cerebral blood flow with dynamic susceptibility contrast-enhanced MRI," *J. Cereb. Blood Flow Metab.*, vol. 25, pp. 378–390, Mar. 2005.

[4] L. H. Cheong, T. S. Koh, and Z. Hou, "An automatic approach for estimating bolus arrival time in dynamic contrast MRI using piecewise continuous regression models," *Phys. Med. Biol.*, vol. 48, pp. N83–N88, Mar. 2003.

[5] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell, "Distance metric learning with application to clustering with side-information," in *Advances in Neural Information Processing Systems 15*, S. Becker, S. Thrun, and K. Obermayer, Eds. Cambridge, MA: MIT Press, 2003, pp. 505–512.

[6] S. E. Rose, A. L. Janke, M. Griffin, S. Finnigan, and J. B. Chalk, "Improved prediction of final infarct volume using bolus delay-corrected perfusion-weighted MRI implications for the ischemic penumbra," *Stroke*, vol. 35, pp. 2466–2471, Nov. 2004.

[7] M. R. Smith, H. Lu, S. Trochet, and R. Frayne, "Removing the effect of SVD algorithmic artifacts present in quantitative MR perfusion studies," *Magn. Reson. Med.*, vol. 10, no. 2, pp. 631–634, Mar. 2004.

[8] F. Calamante, D. G. Gadian, and A. Connelly, "Quantification of bolus-tracking MRI: Improved characterization of the tissue residue function using Tikhonov regularization," *Magn. Reson. Med.*, vol. 50, pp. 1237–1247, Dec. 2003.

[9] S. Sourbron, R. Luypaert, D. Morhard, K. Seelos, M. Reiser, and M. Peller, "Deconvolution of bolus-tracking data: A comparison of discretization methods," *Phys. Med. Biol.*, vol. 52, pp. 6761–6778, Nov. 2007.

[10] L. Torresani and K.-C. Lee, "Large margin component analysis," in *Advances in Neural Information Processing Systems 19*, B. Schölkopf, J. Platt, and T. Hoffman, Eds. Cambridge, MA: MIT Press, 2007, pp. 1385–1392.

[11] G. Chechik, U. Shalit, V. Sharma, and S. Bengio, "An online algorithm for large scale image similarity learning," in *Advances in Neural Information Processing Systems 22*, Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, Eds. 2009, pp. 306–314.

SP

applications CORNER continued from page 150

define a valid distance metric; second, because linear inequalities in the elements of this matrix can ensure that inputs are correctly labeled by kNN classification or Gaussian mixture modeling. Large-scale applications of these ideas are made possible by recent advances in numerical optimization [6]. Looking forward, we anticipate many such applications given the ubiquitous role of distance metrics in both nonparametric and parametric models of classification.

AUTHORS

Kilian Q. Weinberger (kilian@cse.wustl.edu) is an assistant professor in the Department of Computer Science and Engineering at Washington University, St. Louis.

Fei Sha (feisha@usc.edu) is an assistant professor in the Viterbi School of Engineering at the University of Southern California.

Lawrence K. Saul (saul@cs.ucsd.edu) is an associate professor in the Department of Computer Science and Engineering at the University of California San Diego.

REFERENCES

[1] R. Duda, P. Hart, and D. Stork, *Pattern Classification*. New York: Wiley, 2000.

[2] A. Globerson and S. Roweis, "Metric learning by collapsing classes," in *Advances in Neural Information Processing Systems 18*, Y. Weiss, B. Schölkopf, and J. Platt, Eds. Cambridge, MA: MIT Press, 2006, pp. 451–458.

[3] N. Shental, T. Hertz, D. Weinshall, and M. Pavel, "Adjustment learning and relevant component analysis," in *Proc. 7th European Conf. Computer Vision* (Lecture Notes in Computer Science), 2002, pp. 776–792.

[4] S. Shalev-Shwartz, Y. Singer, and A. Ng, "Online and batch learning of pseudo-metrics," in *Proc. 21st Int. Conf. Machine Learning (ICML'04)*, Banff, Canada, 2004, pp. 94–101.

[5] E. Xing, A. Ng, M. Jordan, and S. Russell, "Distance metric learning with application to clustering with side-information," in *Adv. Neural Inform. Process. Syst. 15*. Cambridge, MA: MIT Press, pp. 505–512, 2003.

[6] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[7] K. Weinberger and L. Saul, "Distance metric learning for large margin nearest neighbor classification," *J. Mach. Learn. Res.*, vol. 10, no. 2, pp. 207–244, 2009.

[8] F. Sha and L. K. Saul, "Large margin hidden Markov models for automatic speech recognition," in *Advances in Neural Information Processing Systems 19*, B. Schölkopf, J. Platt, and T. Hofmann, Eds. Cambridge, MA: MIT Press, 2007, pp. 1249–1256.

[9] B. Boser, I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifiers," in *Proc. 5th Annu. Workshop on Computational Learning Theory (COLT'02)*, Pittsburgh, PA, 1992, pp. 144–152.

[10] L. Torresani and K.-C. Lee, "Large margin component analysis," in *Adv. Neural Inform. Process. Syst. 19*, vol. 22, B. Schölkopf, J. Platt, and T. Hoffman, Eds. Cambridge, MA: MIT Press, pp. 1385–1392, 2007.

[11] G. Chechik, U. Shalit, V. Sharma, and S. Bengio, "An online algorithm for large scale image similarity learning," in *Advances in Neural Information Processing Systems 21*, Cambridge, MA: MIT Press, 2009, pp. 306–314.

[12] D. Tran, A. Sorokin, and D. Forsyth, "Human activity recognition with metric learning," in *Proc. European Conf. Computer Vision (ECCV'08)*, 2008, pp. 548–561.

SP