Pattern Recognition 81 (2018) 562–574

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

An approach to supervised distance metric learning based on difference of convex functions programming

Bac Nguyen*, Bernard De Baets

KERMIT, Department of Data Analysis and Mathematical Modelling, Ghent University, Coupure links 653, Ghent 9000, Belgium

ARTICLE INFO

Article history:
Received 8 January 2018
Revised 26 March 2018
Accepted 24 April 2018
Available online 25 April 2018

Keywords:
Distance metric learning
Nearest neighbor
Linear transformation
DC programming

ABSTRACT

Distance metric learning has motivated a great deal of research over the last years due to its robustness for many pattern recognition problems. In this paper, we develop a supervised distance metric learning method that aims to improve the performance of nearest-neighbor classification. Our method is inspired by the large-margin principle, resulting in an objective function based on a sum of margin violations to be minimized. Due to the use of the ramp loss function, the corresponding objective function is nonconvex, making it more challenging. To overcome this limitation, we formulate our distance metric learning problem as an instance of difference of convex functions (DC) programming. This allows us to design a more robust method than when using standard optimization techniques. The effectiveness of this method is empirically demonstrated through extensive experiments on several standard benchmark data sets.

1. Introduction

Recent advances in distance metric learning have demonstrated a promising approach to compute more effective distance metrics (or, equivalently, similarity measures) for a given problem, provided that some constraints or class labels are available [1,2]. The idea is to adjust distances between examples in order to improve the performance of a learning method. For instance, in nearest-neighbor classification, the distances between examples of the same class (i.e. similar examples) should be smaller than those between examples of different classes (i.e. dissimilar examples) [3]. One then hopes to obtain an appropriate distance metric that constitutes a good dissimilarity measure between examples. In a recent survey paper, Bellet et al. [2] gave an overview of distance metric learning methods and their applications according to different criteria, such as learning paradigm, scalability, and form of the distance metric. On the other hand, a large number of distance metric learning methods can be described in a unified framework proposed by Kulis [1]. Among different distance metric learning methods, learning a Mahalanobis distance metric is one of the most successful and well-studied frameworks due to its simplicity and flexibility. One can see the Mahalanobis distance metric as a generalization of the Euclidean distance metric, which allows for rotation and scaling of features. Mahalanobis distance metric learning has been widely used in different contexts, such as classification [3–6], regression [7], subspace learning [8,9], semi-supervised clustering [10,11], unsupervised learning [12], learning to rank [13], etc.

Mahalanobis distance metric learning can be formulated within a convex optimization framework, which enjoys significant advantages in that the convexity guarantees to reach the global optimum and is not sensitive to initial conditions. A large number of optimization methods have been proposed to deal with convex optimization problems [14]. In particular, convex distance metric learning methods are often cast as solving semidefinite programs, therefore, standard semidefinite programming solvers can be used. In order to make the problem more tractable in large-scale settings, Weinberger and Saul [3] developed an efficient subgradient descent method based on the active set techniques. Davis et al. [15] introduced an iterative Bregman projection method to avoid the projection of the Mahalanobis matrix onto the cone of symmetric positive semidefinite (PSD) matrices. Shen et al. [16] proposed a boosting-based method that learns a linear combination of trace-one rank-one matrices. Recently, Atzmon et al. [17] suggested an efficient solver based on the block-coordinate descent method to avoid the projection and computation of full gradients. Other methods such as the Frank–Wolfe [18] and the projected gradient descent [19] methods have also been employed in the context of distance metric learning.

Convex optimization has become very popular in the pattern recognition community over the last few decades, because of its empirical performance and because it facilitates a deeper mathe-

* Corresponding author.
E-mail addresses: bac.nguyencong@ugent.be (B. Nguyen), bernard.debaets@ugent.be (B. De Baets).

https://doi.org/10.1016/j.patcog.2018.04.024
0031-3203/

matical analysis. Unfortunately, in many practical settings, convexity is not always guaranteed, and one has to resort to nonconvex optimization methods [20]. Various researchers [21–23] have argued that using nonconvex loss functions to approximate the misclassification rate can yield a better performance than using convex loss alternatives such as the hinge loss and the exponential loss. Recent research in this direction has provided a number of nonconvex functions in order to alleviate the limitation of convex functions. Shen et al. [24] and Liu and Shen [25] proposed a Ψ-learning framework that replaces the hinge loss function in Support Vector Machines (SVMs) by a nonconvex Ψ-loss function. In a similar variant of the Ψ-loss function, Collobert et al. [20] and Ertekin et al. [26] introduced the ramp loss function, which gives a constant penalty for large losses. Both the Ψ-loss and ramp loss functions have been shown to be effective in practice. Therefore, it is important to investigate the use of nonconvex loss functions in the context of distance metric learning. In particular, we pay attention to the ramp loss function, since it can be easily written as a difference of convex functions (DC). Consequently, an effective method for DC programming can be applied to solve the problem. To the best of our knowledge, the method presented in this paper is the first distance metric learning method that exploits the benefits of DC programming.

Due to the simplicity and effectiveness, our paper focuses on improving the performance of nearest-neighbor classification. It is well known that the misclassification error rate of the nearest-neighbor classifier converges asymptotically to at most twice the Bayes error rate [27], however, it is extremely sensitive to noise. In order to overcome the latter drawback, we develop a distance metric learning method making the nearest-neighbor classifier more robust to outliers. In short, our main contributions are summarized as follows:

1. A distance metric learning framework is proposed to minimize the misclassification rate of the nearest-neighbor classifier. Particularly, our method is inspired by the success of the large-margin principle [28]. Due to the use of the ramp loss function, our objective function for margin maximization has a strong ability to avoid the influence of outliers.
2. Since the objective function can be decomposed into a DC program, a DC algorithm (DCA) [29] is adopted to solve this problem. Our method iteratively solves a sequence of convex subproblems. We refer to the proposed method as Distance Metric Learning using DC programming (DML-dc).
3. We show that the generalization error analysis of the proposed approach has an important theoretical implication in explaining that minimizing the objective function may improve the generalization performance of nearest-neighbor classification. In particular, the generalization performance is guaranteed via the fat-shattering dimension of Lipschitz classifiers through the combination of a large margin and a low-rank Mahalanobis matrix.

The remainder of this paper is organized as follows. Section 2 gives some formal definitions and notations that will be used throughout this paper. Section 3 briefly reviews some existing approaches that are closely related to our work. Section 4 presents our distance metric learning formulation and the corresponding DCA algorithm. Subsequently, Section 5 provides the generalization error of the proposed approach using the large-margin criterion. Experimental results are discussed in Section 6, followed by some concluding remarks in Section 7.

2. Preliminaries

We introduce some notations and background that will be used in the proposed approach.

2.1. Notations

For the sake of convenience, we use the following notations. Matrices are denoted by bold-face uppercase letters; the identity matrix is denoted by $\boldsymbol{I}$. Vectors are denoted by bold-face lowercase letters. Sets are denoted by calligraphic uppercase letters. The Frobenius norm of a matrix $\boldsymbol{M}$ is denoted by $\|\boldsymbol{M}\|_F$. The cone of PSD matrices $\boldsymbol{M}\succcurlyeq 0$ in $\mathbb{R}^{D\times D}$ is denoted by $\mathbb{S}_+^D$. The inner product between two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ is denoted by $\langle \mathbf{A}, \mathbf{B}\rangle = \mathrm{tr}(\mathbf{A}^\top \mathbf{B})$, where tr(.) denotes the trace of a matrix. The distance between a point $\boldsymbol{x}$ and a finite set $\mathcal{S}$ is defined as $d(\mathbf{x}, \mathcal{S}) = \min\left\{d(\mathbf{x}, \mathbf{x}_i) \mid \mathbf{x}_i \in \mathcal{S}\right\}$ for a given distance metric d.

We will consider the standard supervised classification problem. The set of training examples is denoted by $\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid i \in \{1, \ldots, n\}\} \subset \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} \subseteq \mathbb{R}^D$ denotes the set of feature vectors and $\mathcal{Y}$ denotes the set of class labels. Let us introduce the definitions of hit examples and miss examples.

Definition 1 (Hit examples)**.** Let $\boldsymbol{x}_i$ be an example in $\mathcal{X}$. The hit examples of $\boldsymbol{x}_i$ are the elements of the set $\mathcal{H}_i$ consisting of the examples in $\mathcal{X} \setminus \{\mathbf{x}_i\}$ that share the same class label with $\boldsymbol{x}_i$, i.e. $\mathcal{H}_i = \left\{\mathbf{x}_j \mid j \in \{1, \ldots, n\}, j \neq i, y_j = y_i\right\}$.

Definition 2 (Miss examples)**.** Let $\boldsymbol{x}_i$ be an example in $\mathcal{X}$. The miss examples of $\boldsymbol{x}_i$ are the elements of the set $\mathcal{M}_i$ consisting of the examples in $\mathcal{X}$ that do not share the same class label with $\boldsymbol{x}_i$, i.e. $\mathcal{M}_i = \left\{\mathbf{x}_j \mid j \in \{1, \ldots, n\}, y_j \neq y_i\right\}$.

Next, we briefly discuss the main idea of using margins in machine learning, which motivates our approach.

2.2. Margins

To evaluate the performance of a classifier, it does not suffice to consider the training error, but it is also necessary to consider the confidence of the predictions made by the classifier. The margin is one of the geometric measures for evaluating this confidence [30]. It provides theoretical generalization bounds on the effectiveness of a classifier, i.e. the higher the confidence is, the lower generalization error the classifier obtains. Many machine learning algorithms have been analyzed using margin concepts, such as SVMs [28] and AdaBoost [31].

Given a distance metric d, Crammer et al. [30] define the margin by which a labeled example $\boldsymbol{x}_i$ is classified correctly as

$$\phi(\mathbf{x}_i) = d\big(\mathbf{x}_i, \mathrm{NM}(\mathbf{x}_i)\big) - d\big(\mathbf{x}_i, \mathrm{NH}(\mathbf{x}_i)\big), \tag{1}$$

where $\mathrm{NM}(\boldsymbol{x}_i)$ and $\mathrm{NH}(\boldsymbol{x}_i)$ are the elements of $\mathcal{M}_i$ and $\mathcal{H}_i$ that are closest to $\boldsymbol{x}_i$, called nearest miss (NM) and nearest hit (NH), respectively. This margin was originally defined using the Euclidean distance metric for feature selection purposes. The intuition behind this formulation is that it measures how much $\boldsymbol{x}_i$ can travel in the input space before being misclassified. This margin definition is also adopted implicitly in the well-known RELIEF algorithm [32]. RELIEF predefines the NH and the NM in the original input space using the Euclidean distance metric, and it leads to a convex optimization problem. The major issue with RELIEF is that the NH and the NM in the original input space are not always the same in the transformed space.

3. Related work

Our method is closely related to feature selection methods such as RELIEF [32], I-RELIEF [33], and SIMBA [34]. The reader is referred to [35] for a more detailed discussion about this family of algorithms in a unified framework. These methods are developed for selecting a set of features that capture the relevant properties of