Knowledge-Based Systems 146 (2018) 91–103

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Constraint selection in metric learning

Hoel Le Capitaine

Ecole Polytechnique de Nantes, LS2N UMR CNRS 6004, France

ARTICLE INFO

Article history:
Received 9 March 2017
Revised 23 January 2018
Accepted 26 January 2018
Available online 21 February 2018

Keywords:
Active learning
Dynamic constraint selection
Metric learning
Sample weighting
Stochastic learning

ABSTRACT

A number of machine learning and knowledge-based algorithms are using a metric, or a distance, in order to compare individuals. The Euclidean distance is usually employed, but it may be more efficient to learn a parametric distance such as Mahalanobis metric. Learning such a metric is a hot topic since more than ten years now, and a number of methods have been proposed to efficiently learn it. However, the nature of the problem makes it quite difficult for large scale data, as well as data for which classes overlap. This paper presents a simple way of improving accuracy and scalability of any iterative metric learning algorithm, where constraints are obtained prior to the algorithm. The proposed approach relies on a loss-dependent weighted selection of constraints that are used for learning the metric. Using the corresponding dedicated loss function, the method clearly allows to obtain better results than state-of-the-art methods, both in terms of accuracy and time complexity. Some experimental results on real world, and potentially large, datasets are demonstrating the effectiveness of our proposition.

1. Introduction

The concepts of distance (or norm) and similarity are essentials in machine learning, data mining and pattern recognition methods, and more generally in all applications where data is used for analysis and decision making. In particular, observations are grouped together depending on this measure in clustering, or compared to prototypes in classification. However, it is also well known that these measures are highly dependent of the data distribution in the feature space [32]. Historically, methods that are taking this distribution (or manifold) into account are unsupervised (i.e. no class labels are available). Their objective is to project the data into a new space (whose dimension may be lower, for dimensionality reduction, or potentially larger, through kernelization, for finding a separating hyperplane) in which usual machine learning methods are used. The first, most established and widely used method is certainly the Principal Component Analysis. In this kind of approach, called *manifold learning*, the objective is to preserve the geometric properties of the original feature space while decreasing its dimension so as to obtain a useful projection of the data in a lower dimensional manifold, refer to e.g. MDS, ISOMAP, LLE, SNE (see [34] and references therein), or more recently t-SNE [33], a Student-based variation of SNE.

Thereafter, class label information has been used in order to guide this projection, particularly by focusing on easing the prediction task (see e.g. Fisher linear discriminant analysis and its variants for dimension reduction [31]). Here again, the objective is to project the data into a new space that is a linear combination of the original features.

More recently, researchers tried to directly learn the distance (or similarity) measure in the original feature space, without projection.[1]

As opposed to manifold learning, which is unsupervised, *metric learning* uses some background (or side) information. For instance, in the seminal paper of Xing et al. [38], the metric learning problem has been formulated as an optimization problem with constraints. The basic assumption behind this formulation is that the distance between similar objects should be smaller than the distance between different objects. Therefore, we generally consider whether pairs or triplets of observations as the constraints of the optimization problem. More formally, given two observations $\mathbf{x}_i$ and $\mathbf{x}_j$ lying in $\mathbb{R}^p$, one wants to minimize the distance $d(\mathbf{x}_i, \mathbf{x}_j)$ if these two observations are considered as similar, and maximize the distance if they are considered as dissimilar [38]. Alternatively, if the constraints are under the form of triplets of observations, we may minimize the distance $d(\mathbf{x}_i, \mathbf{x}_j)$ between similar objects and maximize the distance $d(\mathbf{x}_i, \mathbf{x}_k)$ between dissimilar objects, as in [6].

Depending on the application, similar and dissimilar objects can be obtained through their class labels for supervised problems,

[1] We will see that in fact, using a Mahalanobis distance is equivalent to perform a linear projection of the data, and then compute the Euclidean distance in this new space.

E-mail address: hoel.lecapitaine@univ-nantes.fr

https://doi.org/10.1016/j.knosys.2018.01.026
0950-7051/

or with must-link and cannot-link, or side information for semi-supervised problems [38]. This information can also be obtained interactively with the help of the user. In that case, online learning algorithms are particularly well suited.

Due to the ever growing size of available data sets, online metric learning has also received a lot of interest. As opposed to batch learning where the entire learning set is available, online learning processes one observation (or pairs, triplets) at a time. Based on the output of each iteration, these approaches rely on getting a feedback of the quality of the metric (for instance a specified loss), and update the model accordingly. This process is repeated until convergence of the metric. The results given by these methods are not as good as batch algorithms, but allows to tackle larger problems [4].

The vast majority of metric learning approaches are using, as metric, the squared Mahalanobis distance between two p-dimensional objects $\mathbf{x}_i$ and $\mathbf{x}_j$ defined by

$$d_A^2(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^T A(\mathbf{x}_i - \mathbf{x}_j) \quad (1)$$

where A is a $p \times p$ positive semi-definite (PSD) matrix. Note that if $A = I$, d_A^2 reduces to the squared Euclidean distance. In this setting, the learning task consists in finding a matrix A that is satisfying some given constraints. In order to ensure that (1) defines a proper metric (i.e. a binary function holding the symmetry, triangle inequality and identity properties), A must remain PSD. Note that in the following, we denote this distance as $d_A(\mathbf{x}_i, \mathbf{x}_j)$. Note also that the matrix A is often the inverse covariance matrix of the data $X = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$.

In practice, it may be intractable, so that several solutions have been proposed. The first one consists in relaxing the metric constraints. For example, in [4], the authors use a bilinear similarity function defined by $s(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T A \mathbf{x}_j$. In this specific case, A is not required to be PSD, so that optimization is facilitated.

Another solution consists in considering a factorization of A as $L^T L$. This decomposition presents two major advantages over A. First, the PSD constraint on A is ensured and second, one can project the data into a lower dimensional space by using the projection matrix L. In particular, if the rank of the matrix A is k, then the matrix $L \in \mathbb{R}^{k \times p}$ is used to project the data into a k-dimensional space ($k < p$). Indeed, some simple algebraic manipulations show that one can write $d_A^2(\mathbf{x}_i, \mathbf{x}_j)$ as $\|L\mathbf{x}_i - L\mathbf{x}_j\|^2$. This projection, similarly to manifold learning, allows to better separate the data for a classification task, see an example in Fig. 1. In this sense, metric learning can also be seen as a supervised dimensionality reduction technique, and also belongs to the hot topic of representation learning.

One of the most important step in metric learning is to define the constraints with respect to the available information (class labels, relative constraints).

The vast majority of learning algorithms are choosing the constraints by randomly selecting pairs (or triplets) of observations that satisfy the constraint, and then feed this pair (triplet) as a constraint into the learning process. However, such random selection presents several drawbacks. First it may not focus on the most important regions of the feature space (e.g. boundary of the classes), and second it remains constant over time, without taking into account the current metric. Those two aspects must be considered for metric learning, and have unfortunately never been jointly considered in metric learning algorithms. The objectives of this paper are to propose solutions taking into account the two limitations of usual approaches.

In this paper, we propose to dynamically generate the constraints, by setting their weights as a function of the current metric. This way, constraints will evolve over time, and will be adapted to the learned metric. Furthermore, the importance of less satisfied constraints (controlled by a margin) are up-weighted, and well satisfied constraints are down-weighted. Such constraints selection allows to focus on difficult observations of the feature space (often lying at the boundaries of classes), and evolve as the model becomes more accurate. Note that the proposed approach is not restricted to a particular metric learning algorithm. More precisely, it can be used in any iterative metric learning algorithm.

Before presenting in detail the proposed approach let us briefly describe some existing metric learning algorithms.

2. Metric learning and related works

2.1. Basic material in metric learning

The literature on metric learning is continuously growing, so that the presentation given here only mentions the most common and well known methods. The interested reader can refer to surveys on this topic, see e.g. [3,15,39]. The general formulation of metric learning is to find A such that $\ell(A, \mathcal{C}) + \lambda R(A)$, where ℓ is a loss function penalizing unsatisfied constraints, $\mathcal{C}$ is the set of constraints. λ is a trade-off parameter between regularization and the loss, and $R(A)$ is a regularizer on A. This model is generally casted as a constrained optimization problem

$$\min R(A)$$
$$\text{s.t. } \ell(A, i) \leq 0, \forall i \in \mathcal{C}$$

Large Margin Nearest Neighbors (LMNN, [37]) is one of the early attempts to learn a Mahalanobis distance metric as a convex optimization problem over the set of PSD matrices. The loss function is composed of the linear combination of two terms ε_{pull} and ε_{push}. The first term aims at penalizing large distances between an observation and other observations sharing the same label, while the second term objective is to penalize small distances between observations from different classes. The loss function is then casted as a semidefinite program. Note that there is no regularization term in the objective function so that LMNN is prone to overfitting. Another well known problem, related to the proposed approach, is that the selection of neighbors is initially made using Euclidean distance, which may not be adapted.

Information-Theoretic Metric Learning (ITML, [6,12]) formulates the distance learning problem as that of minimizing the differential relative divergence between two multivariate Gaussian distribution under constraints on the distance function. In this approach, the regularizer $R(A)$ is taken as the LogDet divergence between successive A_t's. The main benefit is that it ensures that A remains positive semidefinite. Constraints are incorporated through slack variables, and the optimal matrix A is obtained by successive Bregman projections.

It is related to an information-theoretic approach, which mean that there exists a simple bijection (up to a scaling function) between the set of Mahalanobis distances and the set of equal mean multivariate Gaussian distributions.

This method can handle general pairwise constraints, meaning it is sufficiently flexible to support a variety of constraints. ITML does not require eigenvalue computation or semi-definite programming, which allows it to be both efficient and fast for many problems. However, the computational complexity of updating in this algorithm is $O(cp^2)$, the cost increases as the square of the dimensionality. Therefore, this method is not suitable, at least in place, for large-dimensional datasets.

The Online Algorithm for Scalable Image Similarity (OASIS) [4] is an online dual approach using the Passive-Aggressive [5] family of learning algorithms. It learns a similarity function with a large margin criterion and an efficient hinge loss cost. Its goal is to learn a parameterized similarity function of the form $S_W(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T W \mathbf{x}_j$. With this formulation, W plays a similar role as A in metric learning. As in the metric learning formulation, the