Pattern Recognition Letters 36 (2014) 22–28

Contents lists available at ScienceDirect

Pattern Recognition Letters

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

Prototype reduction based on Direct Weighted Pruning

K. Nikolaidis, T. Mu, J.Y. Goulermas [*]

School of Electrical Engineering, Electronics and Computer Science, University of Liverpool, Brownlow Hill, Liverpool L69 3GJ, UK

ARTICLE INFO

Article history:
Received 15 March 2013
Available online 4 September 2013

Communicated by A. Petrosino

Keywords:
Data condensation
Prototype reduction
Instance weight learning

ABSTRACT

Instance-based learning methods often suffer from problems related to high storage requirements, large computational costs for searching through the stored instances to find the ones most similar to the queries, and also sensitivity to noisy samples. In order to deal with these issues, various condensation algorithms have been proposed in the literature to reduce the set of prototypes that need to be stored. In this paper, we propose a new algorithm that uses a set of weights to directly control which prototypes have to be discarded or survive. Instead of relying on indirect heuristics, it explicitly optimizes a bi-objective index which incorporates the condensation rate and a measure of the classification inaccuracy as reflected by the nearest neighbor rule. The proposed algorithm, referred to as DWP (Direct Weighted Pruning), performs an efficient search using a simple genetic algorithm, which is however equipped with three novel acceleration mechanisms to notably speed up its convergence. Experiments over a large number of datasets and comparisons against many other successful condensation algorithms, show that DWP is very effective and achieves the highest classification accuracy along with competitive condensation rates.

1. Introduction

One of the simplest but most effective non-parametric classification methods in machine learning, is the k-Nearest Neighbor (k-NN) rule. In its most basic variant it requires no explicit training, but simply relies on the direct storage of the training prototypes in a database. Subsequently, new queries are compared against this database in order to assess their most likely class memberships. However, there are certain concerns associated with the k-NN rule as well as other instance-based classifiers that rely on the storage of training data. Firstly, the need to retain the entire dataset in some type of memory, poses a burden in the implementation of the system. Secondly, having to search through all the stored data patterns becomes very computationally inefficient for large datasets, even when fast retrieval algorithms are employed. Furthermore, since real-world datasets are often subjected to noisy measurements, the classification performance may suffer from the noisy and likely harmful prototypes stored. It has to be noted, that these issues can also be the case either when the raw features of a dataset are used directly, or when the raw data have been pre-processed to generate new features (e.g., using a projection technique) or to select optimal feature subsets (e.g., using a filter or wrapper approach). Although such feature pre-processing aims to reduce the data dimensionality and also improve classification rates, the above issues of data storage, search complexity and sample noise may still persist in large datasets.

In order to mitigate these problems, numerous data condensation (also referred to as instance/prototype reduction) methods have been proposed, and the field has attracted notable attention in the past few decades since the first methods were introduced (Hart, 1968; Wilson, 1972; Chang, 1974). In general, data condensation methods intend to not only prune the number of data prototypes, but simultaneously increase, where possible, the classification accuracies by removing both superfluous and noisy instances. A well known such method is the Edited Nearest Neighbor (ENN) (Wilson, 1972) filtering algorithm, which has been frequently used in various other condensation methods as a noise pre-processing filter. Different methods employ diverse assumptions and mechanisms to facilitate instance removal. For example, considering that the majority of the information needed to describe the data distributions is provided by the samples close to the decision surfaces, various algorithms employ heuristics to distinguish between border and non-border instances (Wilson and Martinez, 2000; Brighton and Mellish, 2002; Fayed and Atiya, 2009; Nikolaidis et al., 2012). Another group of methods that have become popular in the recent years are ones that model the data and their characteristics using graphs. Examples include the Hit Miss Networks (HMN) (Marchiori, 2008), the Class Conditional Nearest Neighbor (Marchiori, 2010) and Voronoi graphs (Toussaint and Poulsen, 1979). Instance Weight Learning (IWL) is used by another category of methods, where weighting coefficients are assigned to

[*] Corresponding author. Tel.: +44 (0) 151 794 4520; fax: +44 (0) 151 794 4540.
E-mail address: j.y.goulermas@liverpool.ac.uk (J.Y. Goulermas).

0167-8655/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.patrec.2013.08.022

every data instance. Although the majority of the IWL methods pursue distance metric learning to improve the classification rates (Paredes and Vidal, 2006a; Ricci and Avesani, 1999), some of them target prototype reduction (Girolami, 2003; Paredes and Vidal, 2000, 2006b). Another type of condensation methods achieve instance abstraction, where a new set of prototypes is generated to replace the original dataset (Lam et al., 2002a,b; Geva and Sitte, 1991). Other condensation algorithms and heuristics can be found in the recent reviewing works of Garcia et al. (2012) and Triguero et al. (2012).

In this work, we introduce a novel algorithm, referred to as Direct Weighted Pruning (DWP) that has certain distinct features and advantages compared to existing methods. Firstly, it explicitly formulates the two principal objectives of classification accuracy and condensation rate, and this enables the entire procedure to be formulated as a single multi-objective optimization problem. One unique advantage resulting from such modeling is that one can regulate directly the trade-off between the two antagonistic objectives. Further, DWP employs an explicit modeling of simple binary weights to establish the final decision making process about which instances to retain and which to discard. These weights are directly used for the measurement and quantification of both objectives simultaneously, and therefore they enable the formation of a unified performance index. Finally, DWP is based on a highly efficient optimization procedure. Although this is based on a very standard evolutionary search, we have proposed three novel enhancing heuristics that significantly accelerate the search. In Section 3, we compare DWP with eight other state-of-the-art data condensation algorithms for which we summarize their main characteristics, and we analyze results across three synthetic and eighteen real-world datasets of diverse characteristics.

The remaining of the paper is structured as follows. Section 2 describes in detail the proposed algorithm, and specifically the weight modeling, the optimization and the speed up heuristics. Section 3 presents experimental evaluations using the synthetic and real-world datasets and comparisons between DWP and other established reduction algorithms. Finally, Section 4 concludes the work.

2. The proposed algorithm

We assume that we have a dataset X of n data prototypes $\mathbf{x}$. Each $\mathbf{x}$ is of d dimensions and is associated with a class label $\psi(\mathbf{x})$. The principal objective is to reduce X to a much smaller number of $m \ll n$ instances, that match the classification performance of X as close as possible. As it is the norm in all previous works (Wilson and Martinez, 2000; Brighton and Mellish, 2002; Nikolaidis et al., 2012; Marchiori, 2008) for reasons of simplicity, we use the 1-NN rule to measure the classification accuracy supported by the reduced dataset and drive the optimization procedure.

2.1. Instance weight modeling

The proposed model depends on a set of design parameters which explicitly control which prototypes are removed and which are retained. These parameters are defined as a set of n binary weights $w(\mathbf{x}) \in \{0,1\}$, each corresponding to a prototype $\mathbf{x} \in X$. A prototype $\mathbf{x}$ is discarded or preserved when its associated $w(\mathbf{x})$ obtains the value of zero or one, respectively. It should be noted, that an alternative formulation of defining the range of each weight would be to use continuous intervals $[0,1]$. However, such modeling would complicate the decision making unnecessarily, because it would require subsequent thresholding to form the final decision of whether a sample $\mathbf{x}$ should be discarded or not. This would also require the setting of an optimal threshold parameter. Additionally, the bi-objective optimization defined below would be made very difficult to quantify and measure. Finally, the adopted binary weight modeling naturally matches the discrete weight optimization procedure we have employed (described in Section 2.2).

The above modeling allows us to optimize the entire weight vector $\mathbf{w} \in \{0,1\}^n$, such that the classification accuracy is compromised minimally. The overall optimization is bi-objective. Firstly, the *condensation ratio* which can be defined as

$$\frac{n-m}{n} = 1 - \frac{1}{n}\sum_{\mathbf{x}\in X} w(\mathbf{x}) \tag{1}$$

needs to be maximized, in order to remove as many instances as possible. However, removing too many instances will deteriorate the system's performance. Therefore, the second objective is to simultaneously maximize the overall *classification rate*.

A straightforward way for measuring this rate using the 1-NN rule, is to use the ratio of the distance of $\mathbf{x}$ from its nearest friend (where friends are defined as other instances in X from the same class as $\mathbf{x}$) to its nearest enemy (where enemies are instances in X from different classes). In the absence of noise, if this ratio is less than the unity, then the sample $\mathbf{x}$ is supported by X, otherwise it is misclassified. However, with this proposed modeling, nearest friends and enemies are not static, as their existence depends on the current state of $\mathbf{w}$. This is because all prototypes are involved in the model optimization procedure. To incorporate the state of $\mathbf{w}$ into the above modeling, we need to firstly express the nearest friend and enemy of every $\mathbf{x}$, as a function of $\mathbf{w}$, according to

$$\begin{aligned} F(\mathbf{x},\mathbf{w}) &= \operatorname*{argmin}_{\substack{\mathbf{z}\in X-\{\mathbf{x}\}\\ \psi(\mathbf{z})=\psi(\mathbf{x})\\ \mathrm{w}(\mathbf{z})\neq 0}} \|\mathbf{z}-\mathbf{x}\|_2 \\ E(\mathbf{x},\mathbf{w}) &= \operatorname*{argmin}_{\substack{\mathbf{z}\in X\\ \psi(\mathbf{z})\neq\psi(\mathbf{x})\\ \mathrm{w}(\mathbf{z})\neq 0}} \|\mathbf{z}-\mathbf{x}\|_2 \end{aligned} \tag{2}$$

where $F(\mathbf{x},\mathbf{w})$ is the nearest surviving neighbor of $\mathbf{x}$, and $E(\mathbf{x},\mathbf{w})$ its nearest surviving enemy. Then, under $\mathbf{w}$, the ratio

$$\gamma(\mathbf{x},\mathbf{w}) = \frac{\|\mathbf{x}-F(\mathbf{x},\mathbf{w})\|_2}{\|\mathbf{x}-E(\mathbf{x},\mathbf{w})\|_2} \tag{3}$$

is used to test whether $\mathbf{x}$ is classified correctly or not.

Ideally, to enforce maximal classification accuracy, the optimization would need to be subjected to the constraints of $\gamma(\mathbf{x},\mathbf{w}) < 1$ for all $\mathbf{x}$. However, this set of hard constraints cannot be satisfied because of the noise, sparse sampling or the nature of the dataset. Instead, we optimize accuracy in a soft way, by using a smooth penalty function $H[\cdot]$ to penalize all cases with ratios exceeding the unity in an aggregate way. A smooth penalty curve allows for a better balancing of the two competitive objectives and copes better with noise and data sparsity. To facilitate the optimization we use a function, also shown in Fig. 1, defined as

$$H[k] = \frac{1}{1+exp(\alpha(1-k))} \tag{4}$$

The fixed parameter α controls the shape of the curve. If it is set to a high value, the curve becomes more like a step function and gives near zero or one penalty values to ratios just below or above the unity, respectively.

Finally, the two objectives controlling condensation and accuracy can be combined within a single weighted-sum objective function $J(\mathbf{w})$ and optimized according to

$$\min_{\mathbf{w}\in\{0,1\}^n} J(\mathbf{w}) \equiv \sum_{\mathbf{x}\in X} H[\gamma(\mathbf{x},\mathbf{w})] + \frac{\lambda}{n}\sum_{\mathbf{x}\in X} w(\mathbf{x}) \tag{5}$$