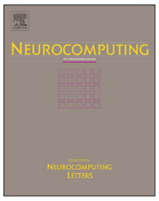
Contents lists available at ScienceDirect

Neurocomputing

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



On the suitability of Prototype Selection methods for kNN classification with distributed data

Jose J. Valero-Mas, Jorge Calvo-Zaragoza*, Juan R. Rico-Juan

Department of Software and Computing Systems, University of Alicante, Carretera San Vicente del Raspeig s/n, 03690 Alicante, Spain

A R T I C L E I N F O

Article history:
Received 28 July 2015
Received in revised form
18 February 2016
Accepted 6 April 2016
Communicated by Jiayu Zhou
Available online 10 May 2016

Keywords:
Prototype Selection
Distributed data
k-Nearest Neighbour
Experimental study

A B S T R A C T

In the current Information Age, data production and processing demands are ever increasing. This has motivated the appearance of large-scale distributed information. This phenomenon also applies to Pattern Recognition so that classic and common algorithms, such as the k-Nearest Neighbour, are unable to be used. To improve the efficiency of this classifier, Prototype Selection (PS) strategies can be used. Nevertheless, current PS algorithms were not designed to deal with distributed data, and their performance is therefore unknown under these conditions. This work is devoted to carrying out an experimental study on a simulated framework in which PS strategies can be compared under classical conditions as well as those expected in distributed scenarios. Our results report a general behaviour that is degraded as conditions approach to more realistic scenarios. However, our experiments also show that some methods are able to achieve a fairly similar performance to that of the non-distributed scenario. Thus, although there is a clear need for developing specific PS methodologies and algorithms for tackling these situations, those that reported a higher robustness against such conditions may be good candidates from which to start.

1. Introduction

Nowadays, our society is strongly characterised by the large amount of information surrounding us. Since the start of the information-related technologies, data production has been reported as constantly growing [1], being this effect more remarkable in the past two decades with the popularisation of the Internet. Data managing algorithms, thus, have to evolve to be able to cope with such requirements [2].

Such sources of information are often distributed in different nodes, especially when huge amounts of data are presented. A clear example of this paradigm would be *crowd-sourcing*, in which a large number of people work in parallel to perform a specific task (e.g., getting labelled data [3]), or *Big Data* [4]. In most cases it is very costly, and even infeasible, to collect all data to be stored in a single node. Therefore, a distributed computing that exploits all the information collected would be of great interest.

Data Mining (DM), considered the main task in the so-called Knowledge Discovery from Databases (KDD) process [5], aims at extracting meaningful patterns from data collections for their later application to the resolution of other problems [6]. In this context, the idea of having several sources of information seems pretty attractive since it stands as an excellent framework to work on. However, current techniques find difficult to manage such scenarios.

Among the different existing DM schemes, supervised learning is the one which aims at obtaining a function out of a set of labelled samples. Within this paradigm, the instance-based learning family comprises those algorithms that directly use the training samples for classification instead of building a model out of them [7].

The *k*-Nearest Neighbour (kNN) rule is one of the most common instance-based learning algorithms in Pattern Recognition (PR) [8]. For a given input, this algorithm hypothesises about its category by querying its *k* nearest neighbours of the training set, following a specified similarity measure. In addition to its straightforward implementation, it reports a very competitive performance in many disparate fields and applications. In turn, it presents important drawbacks, many of which become insurmountable in large-scale scenarios: time efficiency, memory consumption and sensitiveness to noise.

Data Reduction (DR) techniques, which constitute a family of preprocessing methods usually found in the KDD process, have been classically considered for tackling the drawbacks of instance-based classification. These techniques aim at reducing the training set size while trying to maintain, if not increasing, the classification accuracy [9]. Prototype Selection (PS) algorithms, as a

* Corresponding author. Tel.: +34 9 65 903772; fax: +34 9 65 909326.
E-mail addresses: jjvalero@dlsi.ua.es (J.J. Valero-Mas), jcalvo@dlsi.ua.es (J. Calvo-Zaragoza), JuanRamonRico@ua.es (J.R. Rico-Juan).

http://dx.doi.org/10.1016/j.neucom.2016.04.018
0925-2312/© 2016 Elsevier B.V. All rights reserved.

particular example of DR, perform a selection of the most promising instances for classification.

Most PS algorithms require all instances to be processed at the same time. While this premise may be valid in classic problems, it cannot be assumed in aforementioned scenarios as the large quantity of data makes it infeasible. However, due to the distributed nature of this information, a straightforward approach might be to repeatedly apply PS to the different data subsets and then merge the results obtained.

The goal of this paper is to study the behaviour of classic PS algorithms for kNN classification in this distributed context. Particularly, we shall assess the influence of PS when applied to data spread over several partitions, which are eventually joined for creating a single training set of a kNN classifier. For that, we shall consider a simulated scenario that will allow us to measure the performance of the methods as data increasingly approaches to more distributed conditions.

The rest of the paper is organised as follows: Section 2 presents some related work to this topic; the framework proposed to apply PS with distributed data is described in Section 3; Section 4 details the experimentation performed; Section 5 analyses the results obtained and discusses their implications; finally, Section 6 concludes the present work.

2. Related work

As a representative example of instance-based learning, the kNN classification rule generally exhibits a very poor efficiency: since no model or classification function is built out of the training data, each time a new element has to be labelled all training information has to be consulted. This fact has two clear implications: on the one hand, high storage requirements; on the other hand, an elevated computational cost.

These shortcomings have been widely analysed in the literature and several strategies have been proposed to tackle them. In general, they can be divided into three categories:

- Fast Similarity Search (FSS): family of methods which base its performance on the creation of search indexes for fast prototype consulting in the training set.
- Approximated Similarity Search (ASS): approaches which work on the premise of searching sufficiently similar prototypes to a given query in the training set instead of retrieving the exact nearest instance.
- Data Reduction (DR): set of techniques devoted to lower the training set size while maintaining the classification accuracy.

While the two first approaches focus on improving time efficiency, they do not reduce memory consumption. Indeed, some of these techniques speed-up time response at the expense of increasing this factor. Therefore, when large datasets are present in a PR task, the DR framework rises as a suitable option to consider.

DR techniques aim at reducing the size of the initial training set while keeping the same recognition performance [5]. Among the different possible methodologies, the two most common approaches are Prototype Generation (PG) and Prototype Selection (PS) [10]. Both families reduce the initial training set size by discarding redundant information besides removing noisy instances. However, while PG creates new artificial data for replacing the initial information, PS simply selects the most promising elements from the training set. The work presented here focuses on PS techniques, which are less restrictive than PG as they do not require extra knowledge to merge elements from the initial set. Reader may check Ref. [11] for a detailed explanation and thorough study of PG techniques. On the other hand, due to its relevance in the present paper, we now introduce the basics of PS methods.

Given the importance of PS methods in terms of removing both redundant and noisy instances, many different approaches have been proposed. Although a wide range of taxonomies have been proposed for classifying the existing methods, we focus on a particular criterion which establishes three different families:

- *Condensing*: These techniques focus on keeping instances close to decision boundaries and discarding the rest. Accuracy on training set is usually maintained but generalisation accuracy tends to decrease.
- *Editing*: These methods try to minimise the overlapping among the different classes, which generally take place close to the decision boundaries or because of class outliers. Although data reduction figures are lower than in the previous case, generalisation accuracy is higher.
- *Hybrid*: Family of algorithms which looks for a compromise between the two previous methodologies, that is looking for the greatest reduction figure which can improve, or at least maintain, the generalisation accuracy of the initial set.

Reader may check the work of Garcia et al. [12] for a thorough and more comprehensive explanation of taxonomy criteria for PS algorithms.

Although PS may seem a good option to tackle large-scale data, in practice it cannot be directly applied to these scenarios: even if memory requirements could be fulfilled and the algorithms could be applied, as most of them were not designed for so large datasets, an efficient performance might not be possible and incorrect results would be retrieved [13].

To this end, several strategies have been proposed in order to improve the scalability of PS algorithms. A reported successful methodology has been the information stratification, which basically selects a manageable and representative subset with equally distributed classes as training out of the total amount of data [14]. This subset can be latter processed as for instance with evolutionary PS algorithms [15,16] or with the use of memetic PS techniques [17]. With a similar idea, the MapReduce framework has been recently combined with PS in this massive data context [18].

Divide-and-conquer strategies have also been proposed in the literature. A first interesting approach can be found in the work of García-Osorio et al. [19]: the user selects a number of iteration rounds; in each round the initial set is divided into a number of disjoint subsets and a PS process is applied to each of them, receiving a vote each instance selected to be removed; after the established iterations, instances with the highest number of votes are removed. Another remarkable example is the one in [20], in which a number of subsets obtained from the training data are processed using PS and then combined using a voting scheme. A last work to be highlighted is the one of Haro-García et al. [21], in which the divide-and-conquer policy is recursively applied to the data: the initial set is divided into a number of subsets with equally-distributed classes; then each of them is processed with a PS algorithm; the resulting subsets are gathered into one and, then, the process starts over again. The training set is divided into two parts for performing a cross validation evaluation while performing the process and the algorithm iterates until the validation error starts to grow.

As pointed out by several authors [19,22], the division of the initial set into small data excerpts for their independent processing may seriously affect the overall accuracy as each subset only considers a (limited) part of the problem, therefore missing the general vision. While it seems that this effect may be palliated by forcing equality in the class distribution of the instances, this