Choosing Prototype Learning Schemes By Exploring their Characteristics

Author:
Fayola PETERS

Supervisor:
Dr. Tim MENZIES

Word Count ≈ 2300

February 2, 2010

Abstract

Prototype Learning Schemes (PLSs) started appearing in order to alleviate the drawbacks of nearest neighbour algorithms. Namely computation time, storage requirements and the effects of outliers on the classification results. To that end all PLSs have endeavored to create or select a *good* representation of training data which is a mere fraction of the size of the original training data. In most of the literature this fraction is about 10%. With the plethora of PLSs represented in the literature, the following research review realizes a two (2) step procedure which will aid a user in determining which prototype learning scheme (PLS) is best suited for the dataset to be reduced.

1 Introduction

Since the creation of the Nearest Neighbour algorithm [CH67] in 1967, various prototype learning schemes (PLSs) have appeared to remedy the three (3) major drawbacks associated with the algorithm and it's variations. First, the high computation costs caused by the need for each test sample to find the distance between it and each training sample. Second, the storage requirement is large since the entire dataset needs to be stored in memory, and third, outliers can negatively affect the accuracy of the classifer. To solve these issues, PLSs are used. Their main purpose is to reduce a training set via various selection and/or creation methods to produce *good prototypes*. *Good* here meaning that the prototypes are a representation of the original training dataset such that they maintain comparable or increased classification accuracy of the classifier.

A review of the literature on prototype learning has yielded at least forty (40) PLSs each indicating with experimental proof, that their particular design is comparable or better than the standard schemes. With that said, the goal of this paper is not to add to the plethora of PLSs available for a user to select from, nor is it to declare a champion from among the bunch. Not even Bezdek in his exploration of 11 PLSs in 2001 attempted this task, "our study did not attempt to find a *best* 1-np design, but rather, to explore the importance of the three characteristics of all 11 designs" [BK01]. Rather, the current study seeks a review of the literature of PLSs to gain knowledge of the important characteristics popular amoung most PLSs, and establishing a procedure by which a user can choose a PLS or create a hybrid PLS that best suits the data of which he/she has intimate knowledge. Fortunately, previous work done by Bezdek [BK00, BK01] in which he categorises PLS into 3 areas namely 1) Selection vs Replacement 2) Pre supervised vs Post supervised designs and 3) User-defined n versus algorithmically defined n, and work by Kim [KO03] where a comparative study on the various creative PLSs was conducted, will make this task a less daunting one.

In attempt to achieve this goal, the remainder of this paper is organized as follows. First some background introducing some of the desired characteristics of PLSs as a foundation for the procedure. Next, keeping in mind Bezedk's words, "not surprisingly, different [PLS] designs were better for different data sets" [BK01], the procedure is created based on the user's knowledge of their data set and the users' desired characteristics of a PLS. Finally, the conclusion for this work is presented.

2 The Major Characteristics of Prototype Learning Schemes

With data set sizes and feature dimensions greatly increasing over the last thrity to forty years, PLSs designed for nearest neighbour classifiers are essential. As a result, researchers have risen to the challenge creating an impressive collection of PLSs. Each scheme unique in terms of the methods used to create prototypes. For instance, various clustering methods such as k-means and fuzzy c-means [BK00, BK01] have been used. Also search algorithms including tabu and genetic algorithm based search have shown up in the literature [BK00, BK01]. Even evolutionary algorithms [CHL05] and auto immune models [Gar08] show some success as prototype generators. Fortunately, a few researchers organized the PLSs into distinct categories. The most popular of these is Bezdek's categories. He and his co-authors presented three (3) categories for PLS in a 2000 paper titled, *Some Notes on Twenty One (21) Nearest Prototype Classifiers*. Not to be left out, other authors, in distinguishing their PLSs from others in the literature, elaborate on the characteristics which form the basis for their scheme. The following section explores Bezdek's three (3) categories for PLSs as well as other popular characteristics for PLSs which show up constantly in the literature.

2.1 Bezdek's Categories

Bezdek et al [BK00, BK01] claims that among the many characteristics of prototype extraction methods for 1-np classifier design that can be discussed, they consider, 1) selection vs replacement, 2) pre-supervised vs post-supervised and 3) user defined n vs algorithmically defined n, all labelled as C1, C2 and C3 respectively, as the most important.

The categories are defined as follows:

(C1) Selection versus replacement. That is, prototypes may be a subset made up of instances of the training data. This is called *selection*. Examples of these are tabu and genetic algorithms, and also the Minimal Consistent Set (MCS) by Dasarathy. Replacement occurs when new instances are formed using the training data. Examples of this includes Chang's 1974 method, Bezdek modified Changs algorithm (MCA), LVQ and its variations. According to the literature, the selection method is best used in situations where the data is supervised and/or noisy. On the other hand this researcher so far has found no specific criteria for using the replacement method.

(C2) Pre-supervised versus post-supervised designs.

> Pre-supervised methods use the data and the class labels to locate the prototypes. Post-supervised methods first find prototypes without regard to the training data labels, and then assign a class label to (relabel) each prototype. Selection methods are naturally pre-supervised, because each prototype is a data point and already has its (presumably true) label [BK00, BK01].

(C3) User-defined n versus algorithmically defined n .

> Most prototype generators require advance specification of n (e.g., classical clustering and competitive learning methods). Some models have "adaptive" variants where an initially specified n can increase or decrease, i.e., prototypes are added or deleted during training under the guidance of a mathematical criterion of prototype "quality". A third group of methods do not specify n at all, instead obtaining it as an output at the termination of training [BK00, BK01].

Looking at the above categories, it is clear that the goal of the paper [BK01] is to figure out the best method in each category. From the experimental results, the authors concluded that replacement prototypes seem to produce better 1-nearest prototype designs than points selected from the training data at a ratio of 2:1. Also that pre-supervision seems to find more useful prototypes for 1-nearest prototype classifiers than post-supervision does at a ratio of 5:1. The experiments further indicated that methods which "automatically" determine the best number of prototypes and methods that are largely based on user specification and trials-and-error are equally likely to yield good 1-nearest prototype classifiers [BK01].

2.2 Beyond Bezdek's Categories

Although Bezdek's claims that the categorgies outlined in his paper are the most important in prototype learning schemes this research has uncovered other characteristics essential for a user in deciding which PLS is best for their data set. These characteristics as listed as below:

- Order Independent vs Order Dependent Methods
- Consistent vs Non Consistent Methods

2.2.1 Order Independent vs Order Dependent Methods

In Devi's et. al. [DM02] research titled, "An incremental prototype set building technique", the authors go out of their way to show readers that their prototype learning scheme was order independent. Their argument against order dependent methods is stated below:

> It is a well-known fact that Condensed Nearest Neighbour (CNN) is order dependent. When it is applied to a particular data set, it gives a certain condensed prototype set leading to a set of non-intersecting Vornoi regions. Now if the order of the data set is changed, it will result in another set of non-intersecting Vornoi regions. If there are n data points, by permuting this data set in n! ways, we can get n! different condensed prototype sets [DM02].

The authors also highlight a major drawback of having n! prototype sets to choose from, i.e. the computational cost of finding out which prototype set will give the optimal result.

In contrast, the order independent procedure outlined in the paper [DM02], indicates that points collected in a misclassified set and a correctly classified set will be the same, but will only differ in the order they appear if the order of the data set is changed.

Although Devi's work casts a dark shadow over order dependent methods, there are some domains where they can be advantagous. For instance in areas such as robotics and online learning where previous experience (which is classified by a teacher) is used to classify future experiences or decide what is the next best move to achieve a goal. A perfect example of this is MESO (Multi-Element Self-Organizing tree) [KM07]. Kasten describes it as "...a perceptual memory system designed to support online, incremental learning, and decision making in autonomic systems". Basically, prototypes generated here are cluster based and are called spheres. Each sphere contains a collection of similar instances. Sphere membership is dictated by a grow function which manages the size of the sphere. So, any new instance finds its closest sphere, if the distance is less than the distance of the growth function then it gets added to the sphere otherwise a new sphere is created with the instance. Organizing the data in this way is perfect when the size of the data set is indefinite as they are in these domains.

2.2.2 Consistent vs Non Consistent Methods

In the literature and this work, consistent is used to mean that all (100% of) the original samples are correctly classified by the prototype subset. Therefore non-consistent is anything less than 100% classification accuracy. From the literature PLSs which guarantees a consistent result or have levels of consistency (above 95%) produce competative nearest neighbour classifiers (NNC) with high classification accuracy. However, before the reader firmly decides that consistent methods are the only way to go, this researcher must admonish about a major limitation of this decision. The problem is overfitting. Veenman, explains:

The problem is that perfect training performance by no means predicts the same performance of the trained classifier on unseen objects. Given that the training data is sampled similarly from the true distribution as the unseen objects, the cause of this problem is twofold. First, the training data set contains an unknown amount of noise in the features and class labels, so that the exact position of the training objects in feature space is uncertain. Second, the training data may be an undersampling of the true data distribution. Unfortunately, this is often the case, so that the model assumptions about the data distribution are not justified [VR05].

3 Guide to Choosing A Prototype Learning Scheme

Along with the above information, a simple two (2) step procedure can be used to aid a user in choosing a PLS best for the dataset to be reduced. First, the user needs to extrapulate certain information about the data. For instance, the size of the dataset (see Figure 1), its feature dimensions, whether it is supervised or unsupervised, an estimate of how noisy the data is, in other words how much data is missing, and finally, what domain is the dataset collected from? Next, based on the previous information, the user can now determine the desired characteristics of the prototype learning scheme for generating prototypes. To make clearer the use of the two step procedure stated here, the following section details two (2) general cases as examples.

Sizes	
Small	$\ll 5,000$
Medium	$\geq 5,000$ *and* $\ll 10,000$
Large	$\geq 10,000$ *and* $\ll 100,000$
Huge	$\gg 100,000$

Figure 1: Definition of Sizes Use in this Paper

3.1 Two (2) Example Cases Defined

The problems addressed in this work are as follows. Please note that for simplicity, all the data in the cases described below are assumed to have no missing data. However, when applicable, mention will be made about PLSs that handle data sets with missing data.

3.1.1 Case 1

A user wants to find a PLS that best suits a medium sized data set where the data is supervised and has a high dimensionality. The user decides that a prototype learning scheme which uses a selection strategy rather than a replacement strategy would be best. Other characteristics desired by the user are stated below.

- Order Independent
- 100% Consistent

The above criteria filter out a short list of prototype learning schemes. These include Devi et. al. [DM02], and to a lesser extent, Dasarathy et. al [Das94]. The algorithm proposed by Devi and co-authors, guarantees consistency at 100% and painstakingly proves to the reader that their work is order independent. The problem of a data set with high dimensionality is also addressed. They admit that especially in data sets with high dimensionality, the boundaries of each class are very diffcult to determine, so in order to partition a region of a class into simpler non-overlapping regions, their algorithm incrementally adds prototypes to a representative prototype set until all the training patterns are classified correctly using this set of representative prototypes [DM02]. Dasarathy's work is also order independent, however, although it is possible, 100% consistency is not guaranteed and no special mention is made about high-dimentional data sets.

3.1.2 Case 2

A user wants to find a PLS that best suits a huge sized data set where the data is supervised. The user decides that a prototype learning scheme which uses a replacement strategy rather than a selection strategy would be best. Also, since the users' data set is huge, a scalable approach is preferred.

A handful of PLSs stand out as suitable to fit the above criteria main because of the data set size. However one of them leaves rooms for hybrid PLS creation. That is "Stratification for scaling up evolutionary prototype selection" by Cano and his co-authors [CHL05]. This method leaves the door open to any PLS the user would like to try beside the evolutionary prototype selection scheme used in the paper. The key here is the stratification procedure which breaks the data set into subsets called strata, maintaining the class distributions in each. Then the prototype learning scheme can be applied to each strata.

4 Conclusions

The two (2) step procedure in this paper offers an effective stategy for choosing a PLS that best suits a particular data set. The user armed with intimate knowledge of the data, can use this knowledge along with a wish list of the characteristics of what they believe will lead to a prototype learning scheme best suited to their data. To illustrate the procedure two (2) cases are presented.

References

[BK00] JC Bezdek and LI Kuncheva. Some notes on twenty one (21) nearest prototype classifiers. In Ferri, FJ and Inesta, JM and Amin, A and Pudil, P, editor, *ADVANCES IN PATTERN RECOGNITION*, volume 1876 of *LECTURE NOTES IN COMPUTER SCIENCE*, pages 1–16. 2000.

[BK01] JC Bezdek and LI Kuncheva. Nearest prototype classifier designs: An experimental study. *INTERNATIONAL JOURNAL OF INTELLIGENT SYSTEMS*, 16(12):1445–1473, DEC 2001.

[CH67] T. Cover and P. Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27, Jan 1967.

[CHL05] Jos Ramn Cano, Francisco Herrera, and Manuel Lozano. Stratification for scaling up evolutionary prototype selection. *Pattern Recognition Letters*, 26(7):953 – 963, 2005.

[Das94] B.V. Dasarathy. Minimal consistent set (mcs) identification for optimal nearest neighbor decision systems design. *Systems, Man and Cybernetics, IEEE Transactions on*, 24(3):511–517, Mar 1994.

[DM02] V. Susheela Devi and M. Narasimha Murty. An incremental prototype set building technique. *Pattern Recognition*, 35(2):505 – 513, 2002.

[Gar08] Utpal Garain. Prototype reduction using an artificial immune model. *Pattern Anal. Appl.*, 11(3-4):353–363, 2008.

[KM07] E.P. Kasten and P.K. McKinley. Meso: Supporting online decision making in autonomic computing systems. *Knowledge and Data Engineering, IEEE Transactions on*, 19(4):485–499, April 2007.

[KO03] SW Kim and BJ Oommen. A brief taxonomy and ranking of creative prototype reduction schemes. *PATTERN ANALYSIS AND APPLICATIONS*, 6(3):232–244, DEC 2003.

[VR05] CJ Veenman and MJT Reinders. The nearest subclass classifier: A compromise between the nearest mean and nearest neighbor classifier. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 27(9):1417–1429, SEP 2005.