Speeding Up Optimum-Path Forest Training by Path-cost Propagation

Adriana S. Iwashita[1], João P. Papa[1], Alexandre X. Falcão[2], Roberto A. Lotufo[3]
Victor M. de Araujo Oliveira[3], Victor H. Costa de Albuquerque[4], João Manuel R. S. Tavares[5]

[1]*São Paulo State University - Department of Computing, Bauru - Brazil*
[2]*University of Campinas - Institute of Computing, Campinas - Brazil*
[3]*University of Campinas - School of Electrical and Computer Engineering, Campinas - Brazil*
[4]*Center of Technological Sciences - University of Fortaleza, Fortaleza - Brazil*
[5]*Faculty of Engineering of the University of Porto, Porto - Portugal*
E-mail: papa@fc.unesp.br

Abstract

In this paper we present an optimization of the Optimum-Path Forest classifier training procedure, which is based on a theoretical relationship between minimum spanning forest and optimum-path forest for a specific path-cost function. Experiments on public datasets have shown that the proposed approach can obtain similar accuracy to the traditional one but with faster data training.

1. Introduction

Pattern recognition techniques attempt to learn a function that maps the input data to some label based on the samples' behavior. Such learning process is the most time-consuming phase of several pattern recognition techniques, since it can be composed by a training step followed by parameters' optimization, which may require a retraining procure. The problem gets worst in interactive classification tools, in which the user is asked to label samples that will be used for training. The remaining image is then classified and the results can be refined by another samples' manual labeling. This situation may be unacceptable for high resolution images.

In the last decade, several pattern recognition techniques have been proposed and widely used in several applications. Among them, we may cite Support Vector Machines (SVM) [3], Artificial Neural Networks (ANN) and Self-Organizing Maps (SOM) [5], and also Bayesian classifiers. The main problem of such techniques is the high computational burden for training. Although SVM have achieved high recognition rates for several applications, when the training set becomes large, its learning may be impracticable for real time applications with fast feedback responses.

Recently, a novel pattern recognition technique called Optimum Path Forest (OPF) was proposed in order to overcome such challenges. Papa et al. [7] showed that OPF obtained similar effectiveness to SVM, but that is faster in the training phase. The OPF models the problem of pattern recognition as a graph partition into optimum-path trees (OPTs), which are rooted by key samples (prototypes) that try to compete among themselves in order to conquer the remaining samples. Such prototype nodes can be found by computing a Minimum Spanning Tree (MST) over the training set and marking the connected nodes from different labels.

Further, Papa et al. [6] proposed an optimization of the OPF classification algorithm, which lead such approach to be faster than the original one, and with similar recognition rates. In this paper, we present an optimization of the OPF training step based on a theoretical relationship between the optimum-path forest generated by OPF and the Minimum Spanning Forest (MSF), which is essentially a MST generated by OPF in the training phase with the edges between prototypes. Such contribution is very interesting in the context of OPF-based research, mainly in applications involving large datasets. The remainder of this paper is organized as follows. Sections 2 and 3 present the OPF background and the proposed training step optimization algorithm, respectively. Section 4 discusses the experimental results, and Section 5 states the conclusions.

The first and second authors would like to thank Fapesp grants #2010/12697-8 and #2009/16206-1, and CNPq grant #303182/2011-3. The fourth author thanks CNPq and FUNCAP a DCR grant (project number 35.0053/2011.1) from UNIFOR, in Brazil.

2. Pattern Classification using Optimum-Path Forest

The OPF classifier works by modeling the problem of pattern recognition as a graph partition in a given feature space. The nodes are represented by the feature vectors and the edges connect all pairs of them, defining a full connectedness graph. The partition of the graph is carried out by a competition process between some key samples (prototypes), which offer optimum paths to the remaining nodes of the graph. Each prototype sample defines its own OPT, and the collection of all OPTs defines an optimum-path forest, which gives the name to the classifier [7, 6].

2.1 Background Theory

Let $Z = Z_1 \cup Z_2$ be a dataset labeled with a function λ, in which Z_1 and Z_2 are, respectively, a training and test sets such that Z_1 is used to train a given classifier and Z_2 is used to assess its accuracy. Let $S \subseteq Z_1$ a set of prototype samples. Essentially, the OPF classifier creates a discrete optimal partition of the feature space such that any sample $s \in Z_2$ can be classified according to this partition.

The OPF algorithm may be used with any *smooth* path-cost function which can group samples with similar properties [4]. Particularly, we used the path-cost function f_{max}, which is computed as follows:

$$\begin{aligned} f_{max}(\langle s \rangle) &= \begin{cases} 0 & \text{if } s \in S, \\ +\infty & \text{otherwise} \end{cases} \\ f_{max}(\pi \cdot \langle s,t \rangle) &= \max\{f_{max}(\pi), d(s,t)\}, \end{aligned} \quad (1)$$

in which $d(s,t)$ means the distance between samples s and t, and a path π is defined as a sequence of adjacent samples. As such, we have that $f_{max}(\pi)$ computes the maximum distance between adjacent samples in π, when π is not a trivial path.

The OPF algorithm assigns one optimum path $P^*(s)$ from S to every sample $s \in Z_1$, forming an optimum path forest P (a function with no cycles that assigns to each $s \in Z_1 \backslash S$ its predecessor $P(s)$ in $P^*(s)$ or a marker nil when $s \in S$. Let $R(s) \in S$ be the root of $P^*(s)$ that can be reached from $P(s)$. The OPF algorithm computes for each $s \in Z_1$, the cost $C(s)$ of $P^*(s)$, the label $L(s) = \lambda(R(s))$, and the predecessor $P(s)$.

2.2 Training

We say that S^* is an optimum set of prototypes when the OPF algorithm minimizes the classification errors for every $s \in Z_1$. S^* can be found by exploiting the theoretical relation between MST and OPT for f_{max} [2]. The training essentially consists in finding S^* and an OPF classifier rooted at S^*.

By computing an MST in the complete graph (Z_1, A), we obtain a connected acyclic graph whose nodes are all samples of Z_1 and the arcs are undirected and weighted by the distances d between adjacent samples. The spanning tree is optimum in the sense that the sum of its arc weights is minimum as compared to any other spanning tree in the complete graph. In the MST, every pair of samples is connected by a single path which is optimum according to f_{max}. That is, the minimum-spanning tree contains one optimum-path tree for any selected root node. The optimum prototypes are the closest elements of the MST with different labels in Z_1 (i.e., elements that fall in the frontier of the classes). After finding prototypes, we run the competition process in order to build the optimum-path forest.

2.3 Classification

For any sample $t \in Z_2$, we consider all arcs connecting t with samples $s \in Z_1$, as though t were part of the training graph. Considering all possible paths from S^* to t, we find the optimum path $P^*(t)$ from S^* and label t with the class $\lambda(R(t))$ of its most strongly connected prototype $R(t) \in S^*$. This path can be identified incrementally by evaluating the optimum cost $C(t)$ as

$$C(t) = \min\{\max\{C(s), d(s,t)\}\}, \ \forall s \in Z_1. \quad (2)$$

Let the node $s^* \in Z_1$ be the one that satisfies Equation 2 (i.e., the predecessor $P(t)$ in the optimum path $P^*(t)$). Given that $L(s^*) = \lambda(R(t))$, the classification simply assigns $L(s^*)$ as the class of t. An error occurs when $L(s^*) \neq \lambda(t)$.

3 Optimization by Path-cost Propagation

As aforementioned in Section 2.2, the optimum set of prototypes S^*, i.e., the one that minimizes the classification errors in the training step, can be found by exploiting the theoretical relation between MST and OPT for f_{max} [2].

Let G be a graph and F a minimum spanning forest relative to G, which is essentially a collection of minimum spanning trees. Allene et al. [2] have proved that if F is a MSF, F is also an optimum-path forest using f_{max} as path-cost function. This theorem stands that given a MST of G, we can obtain an optimum-path forest of it by just removing the arcs between prototypes and updating the costs of all nodes, which we call path-cost propagation.

The intuitive idea behind this theorem says that the optimum-paths from prototypes to all nodes will follow the shape of MST, and an optimum-path from the prototype of the white class, for instance, will need to pass through the prototype from the black class in order to reach the black nodes. As we do not have negative distances, one prototype can not conquer another one. Therefore, the black prototype works as a sentinel to protect the black nodes from paths that come from the white ones. It should be noted that if we have an unique MST, the OPF training error would be zero.

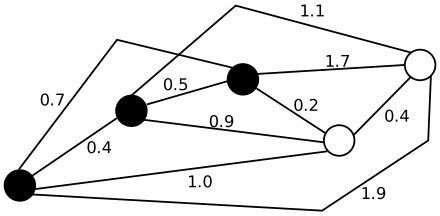

Figure 1. Example of a complete graph. The weights of the edges stand for the distance between nodes.

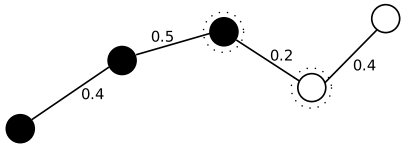

Figure 2. Minimum Spanning Tree of the graph displayed in Figure 1.

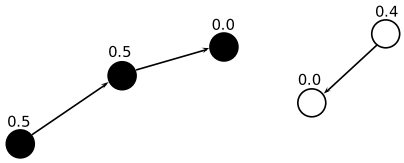

Figure 3. Optimum-Path forest of the graph displayed in Figure 1 using the prototypes obtained in Figure 2.

In Figure 1, we can see a complete graph where the edges between nodes denote the distance between their corresponding feature vectors. The next step of OPF training algorithm is to find the prototypes, which can be done by computing a MST and then marking the nearest nodes from different classes (Figure 2 displays the MST and the bounded nodes stand for the prototypes). Finally, we run the OPF algorithm itself (see Section 2.2) in order to begin the competition process and then to partition the graph into OPTs, which are rooted at each prototype (Figure 3).

From Figure 2, we can also obtain the optimum-path forest shown in Figure 3 by only removing the arc between prototypes, directing the arcs and conducting the path-cost propagation using f_{max}. Therefore, it is not needed to run OPF algorithm, which makes the whole procedure faster for training. However, it is important to emphasize that the proposed optimization approach based on path propagation only works with f_{max}, as stated in [2]. Although other path-cost functions have been proposed, the ones which employs f_{max} is the most used.

4 Experiments

In this section, we discuss the experiments conducted in order to show the efficiency of the proposed optimization for OPF training step. We have used two public datasets with different sizes to assess the training efficiency and also the classification accuracy in distinct scenarios:

- MPEG-7 Shape Dataset [1]: it comprises 1400 silhouette images equally divided in 70 classes, i.e., 20 samples per class. As we have employed the Fourier descriptor to each image, the feature vector contains 126 dimensions; and
- Covtype Dataset[1] : it comprises 29048 images, divided in 7 classes with a feature vector containing 54 dimensions. This version is a subset containing 5% of the original one.

In regard to training and test sets size, we employed a cross-validation procedure with different percentages for training and test sets, which range from 10% to 90% of training set size, with steps of 10. Figure 4 displays the mean execution times in seconds for training concerning the traditional and the proposed approach for MPEG-7 dataset. We not show the accuracy results since the recognition rates in the test set for both approaches were the same. As one can see, the proposed approach becomes still faster when the training set size increases. On the average, the proposed approach has been 1.403 times faster for training than traditional one, without loss of generalization over the test set.

Figures 5 and 6 display, respectively, the mean accuracy and training times for the different percentages of training and test sets for Covtype dataset. In order to assess the results, we have executed an unpaired Student's t-test with significance level equals to 0.05. The

[1] http://archive.ics.uci.edu/ml

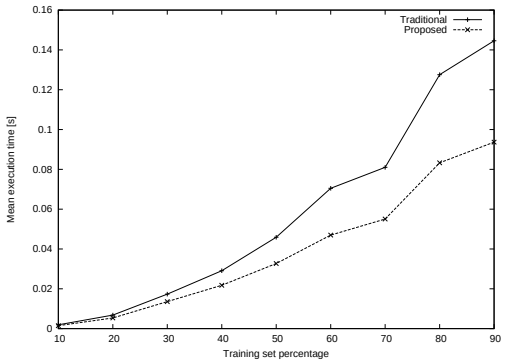

Figure 4. Mean execution times in seconds with respect to training step for MPEG-7 dataset.

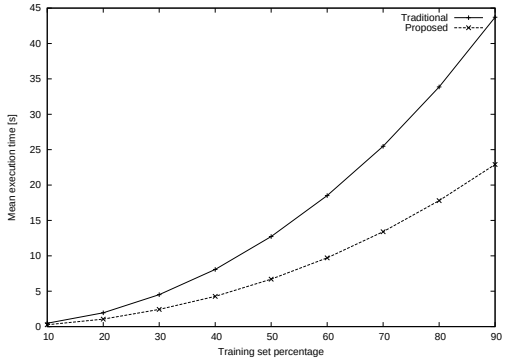

Figure 6. Mean execution times in seconds with respect to training step for Covtype dataset.

null hypothesis, i.e., the ones that guarantees that the proposed training algorithm and the traditional one have the same accuracy was accepted for training set percentages at 10%, 20%, 30%, 70% and 90%. In regard to the remaining training set sizes, the traditional algorithm was slightly better. However, the proposed approach has been 1.878 times faster for training (on average). The speed up can be higher as the training set increases.

It is important to highlight that such differences on the accuracy are due to the presence of several MSTs, i.e., several optimum paths. Suppose we have a node s from the white class that offers an optimum path-cost $C_s(t)$ to a black sample t. Now, assume that we have a black node p that also offers the optimum path-cost $C_p(t)$, i.e., $C_s(t) = C_p(t)$. In this case, if s came out of the queue before p, s will conquer t, and we will have a misclassification.

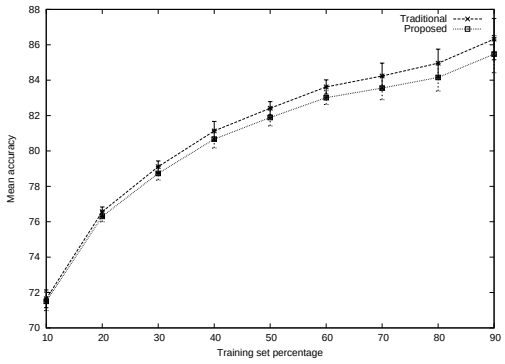

Figure 5. Mean classification accuracy for Covtype dataset.

5 Conclusions

In this paper, we have proposed an optimization of the OPF training step algorithm based on a theoretical relation between minimum spanning forests and optimum-path forests. As the optimum-paths follow the shape of the MST, we can build an optimum-path forest by removing the arcs between prototypes, redirecting the edges and propagating the path-costs using f_{max}. The proposed approach has shown to be up to 2 times faster for training with similar accuracy over the test sets.

References

[1] Mpeg-7: The generic multimedia content description standard, part 1. *IEEE MultiMedia*, 09(2):78–87, 2002.

[2] C. Allène, J.-Y. Audibert, M. Couprie, J. Cousty, and R. Keriven. Some links between min-cuts, optimal spanning forests and watersheds. In *Proceedings of the International Symposium on Mathematical Morphology*, volume 1, pages 253–264, 2007.

[3] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[4] A. Falcão, J. Stolfi, and R. Lotufo. The image foresting transform theory, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(1):19–29, 2004.

[5] S. Haykin. *Neural Networks: A comprehensive foundation*. Prentice-Hall, 1998.

[6] J. P. Papa, A. X. Falcão, V. H. C. Albuquerque, and J. M. R. S. Tavares. Efficient supervised optimum-path forest classification for large datasets. *Pattern Recognition*, 45(1):512–520, 2012.

[7] J. P. Papa, A. X. Falcão, and C. T. N. Suzuki. Supervised pattern classification based on optimum-path forest. *International Journal of Imaging Systems and Technology*, 19(2):120–131, 2009.