Combining global and local classifiers with Bayesian network

Leonardo Nogueira Matos
Federal University of Sergipe
Computer Science Department
lnmatos@ufs.br

João Marques de Carvalho
Federal University of Campina Grande
Electrical Engineering Department
carvalho@dee.ufcg.edu.br

Abstract

This paper introduces a classification method based on feature space segmentation. Since the classification task is equivalent to a probability distribution estimation, a Bayesian network is used as an inference mechanism for dealing with the underlying probability distribution function that, presumably, is complex and factored. The article presents a method for splitting the feature space into regions that are associated to local classifiers. After that, a Bayesian network is used for combining their outputs. Experimental results reveal that this is a suitable approach for speeding up the training phase for large databases as well as to ensure good recognition rates.

1. Introduction

In pattern recognition, the classification problem deals with a mapping between a multidimensional space, \( \mathbb{R}^p \subset \mathbb{R}^n \), the feature space, and a finite and discrete set \( \Omega = \{\omega_i\}_{i=1}^C \). If complete statistical information about the distribution of \( P(\omega_i) \) and \( P(x|\omega_i) \), where \( x \in \mathbb{R}^p \) is a pattern of unnoted class, is known beforehand, then the class associated to \( x \) can be estimated via Bayes formulae ([8]) as

\[
\omega^* = \arg \max_{\omega_i} \{P(\omega_i|x)\} \quad x \in \mathbb{R}^p
\]

with

\[
P(\omega_i|x) = \frac{P(x|\omega_i)P(\omega_i)}{\sum_j P(x|\omega_j)P(\omega_j)}
\]

However, the probabilities \( P(\omega_i) \) and \( P(x|\omega_i) \) are not known in practical problems. Hence, Equation (2) cannot be solved exactly, and needs to be estimated. In this article an original method to estimate the distribution in Eq (2) based on a non parametric mixture model is presented. The target distribution is computed as a weighted sum of partial estimations of \( P(\omega_i|x) \). Those estimations are, by they turn, the outputs of locally specialized classifiers, i.e. classifiers that are trained with patterns that lie in a local subset of the feature space. Therefore, the mixture model developed in the the present work is equivalent to a method for combining classifiers. One original contribution is the scheme used for splitting the feature space into local subsets.

Another important contribution of this paper is the use of an approach for combining classifiers in which the combining rule is implemented by a Bayesian belief network or, shortly, a Bayesian network ([5]). The advantages of using a Bayesian network are twofold. First, this is a method that efficiently estimates a probability distribution function. This implies that complex decisions can be performed by the combiner, allowing the use of weak local classifiers. Second, despite probability distribution estimation be a NP-complex problem, for the restrict class of distributions modeled as politree networks, as in the this proposed method, polynomial time algorithms are available ([5]).

The article is organized as follows: Section 2 presents the feature space segmentation process, which originates locally specialized classifiers; Section 3 discuss how a Bayesian network can be derived for combining the output of local classifiers; Section 4 presents experimental results for an OCR application; finally, Section 5 draws some concluding remarks.

2. Feature space segmentation

Some well known methods like CART ([8]), MARS ([8]) and mixture of experts ([4]) are based on a recursive splitting of the pattern space. The fundamental principle governing those methods is divide and conquer, meaning that the solution of a complex problem can be reached by decomposing
the problem in simpler instances, solving those instances independently and combining their outputs suitably to obtain the desired result.

A divide and conquer algorithm can be a suitable mechanism to combine classifiers, although some statistical drawbacks of that approach should be observed. In a local subspace the bias of the estimator is generally small, allowing the use of a weak classifier. However the variance is generally high because some classifiers are trained with very sparse data sets. In order to mitigate the effects of dealing with scattered data, the literature points out some variance-decreasing devices. The approach used by Jordan and Jacobs [4] consist of performing soft data split, i.e. allowing some data to lie simultaneously in multiple regions. The method proposed in the present work also promotes a soft partitioning of the feature space.

In this text, for convenience, it is used the same notation of Friedman [2] and Peng and Bhanu [6].

**Definition 1** A region, denoted by \( R_m \subset \mathbb{R}^p \), is stamped by its shape \( s_m(\alpha) \) and its center \( u_m \).

Where the shape is given by

\[
s_m(\alpha) = \text{ave}_{x \in R_m} |\alpha^t(x - u_m)|
\]

where \( \alpha \) is a unit vector in \( \mathbb{R}^p \) and ave is an averaging function.

The shape of \( R_m \) is governed by the prediction function \( f(x) \), being larger onto directions where \( f(x) \) and its estimation \( h(x) \) disagree more intensively. This leads to a method that yields new classifiers, specialized in cases that are not learned for the original one. The center and the shape of a partition are described next.

**2.1. Identification of \( u_m \)**

Since a partition corresponds to a subspace where the original classifier is not capable of learning, the partition center \( u_m \) should belong to a neighborhood with a large amount of not-learned samples. The exact solution to this problem involves an exhaustive search, and can not be performed due to the high computational cost. An approximated solution is determined instead by a heuristic search based on entropy. The center of a partition is the element that maximizes the entropy, given by Eq (4).

\[
u_m = \arg\max_{x \in \mathbb{R}^p} \left\{ - \sum h(x_i) \ln h(x_i) \right\}
\]

Any other criteria that measures the classifier outputs, indicating whether a pattern was correctly learned or not, can be used instead of entropy. The mean squared error is an example of such criteria. However the mean square error is very affected by outliers. Although in the neighborhood of an outlier there are no misclassified samples, the mean square error is high. Entropy, by its turn, is not affected by outliers and is also a reliable measurement of classifier correctness.

**2.2. Identification of \( s_m(\alpha) \)**

In a multidimensional space, \( u_m \) is simultaneously placed near many not learned patterns as well as many patterns which have been learned. The more efficient the shape of \( R_m \) is in identifying the patterns around \( u_m \) which need to be submitted to a new training cycle, the better it will be. It is desirable that \( s_m(\alpha) \) extend in the directions of the not-learned patterns only. Those directions are selected by the mean squared error, as stated on Lemma 1.

**Lemma 1** Let \( h(\cdot) : \mathbb{R}^n \rightarrow \mathbb{R}^m \) be a classifier that approximates a distribution of probability. Let \( y = h(x) \) be the output of a test pattern \( x \), i.e., \( y = [y_1 \ldots y_m]^t \) with \( y_i \geq 0 \) and \( \sum_{i=1}^m y_i = 1 \) and let \( t \in \mathbb{R}^m \) be the desired target. If \( \text{mse}(x) < 0.5/m \) then \( x \) was correctly classified, where \( \text{mse}(x) = \sum_{i=1}^m (t_i - y_i)^2 \) is the mean squared error of \( x \).

**Proof.** Assume, for simplicity, that the class of \( x \) is \( \omega_m \), so \( t = [0 \ 0 \ldots 1]^t \). If \( y = h(x) \) corresponds to a correct output then \( y_m > y_1, i = 1, \ldots, m - 1 \). Obviously, if \( y_m > 0.5 \) then the output is correct, since by hypothesis \( y_i \geq 0 \) and \( \sum_{i=1}^m y_i = 1 \). For a given pattern \( x \), if \( y_m \geq 0.5 \) then \( \text{mse}(x) \) is maximum when \( y_m = 0.5 \), since for any different value, both parts of Eq (5) are simultaneously minimized.

\[
\text{mse}(x) = \frac{1}{m} \left[ \sum_{i=1}^{m-1} y_i^2 + (1 - y_m)^2 \right]
\]

When \( y_m = 0.5 \) the maximum \( \text{mse}(x) \) is obtained by the solution of the quadratic programming problem in Eq(6).

\[
\begin{align*}
\max & \quad y_1^2 + \ldots + y_{m-1}^2 \\
\text{s.t.} & \quad y_1 + \ldots + y_{m-1} = 0, 5 \\
& \quad y \geq 0
\end{align*}
\]
This problem has \( m - 1 \) solutions, that consist of any vector of the form \( y = [0 \ldots 0.5 \ldots 0.5]^t \). In this case, \( mse(x) = \frac{0.5}{m} \). So, for a given pattern \( x \), if \( mse(x) < \frac{0.5}{m} \) then it was correctly learned. Conversely, observe that if \( mse(x) \geq \frac{0.5}{m} \) it does not necessarily means that the pattern was not learned.

\[
\square
\]

### 2.3. The Partitioning Algorithm

Based on Lemma 1, a partition \( R_m \) can be established by the following procedure: let \( r \) be the radius of a hipersphere \( \mathcal{H} \subset \mathbb{R}^p \) with center in \( u_m \). \( R_m \subset \mathcal{H} \) is the subspace \( \{x \in \mathcal{H} \land mse(x) > \frac{0.5}{m}\} \). If \( r \) is small enough there is no superposition between partitions. As \( r \) grows, the subspace becomes difficult to be learned by a new classifier. In the experiments performed the value of \( r \) was the mean distance between \( x \in \mathcal{M} \) and \( u_m \), where \( \mathcal{M} \) is the set of not-learned patterns in the trained data. This approach favors the generation of superimposing partitions, which can be learned by a new classifier.

### 3. The Decision Making System

The Bayesian network is the decision making system that combines the multiple classifier outputs in order to produce an unique prediction. There are two major procedures involved with a Bayesian network: learning and inference. Bayesian network learning consists of obtaining the subjacent graph, or Bayesian network structure \( B_2 \), as well as a set of conditional probabilities between adjacent nodes, named \( B_P \). Bayesian network inference consists of evaluating for each node a vector of conditional probabilities, named belief vector, in response to evidence, that is, a set of nodes of known states.

The proposed partitioning algorithm originates a tree for which each node corresponds to a subspace and to a classifier. That structure, named classifiers diagram, guides the construction of \( B_S \). The Bayesian network graph reproduces the hierarchic structure of the classifiers diagram, meaning that the deeper is the node in the classifiers diagram, the more specialized it is. The Bayesian network captures this property. In addition, since classifiers produce individual estimates of a pattern class, they are associated to leaf or terminal nodes in the Bayesian network graph. The combining nodes, that summarizes predictions from classifiers, are associated to a non-terminal nodes. In order to preserve the hierarchic structure of a classifiers diagram, one non-terminal node should be introduced at each diagram level (Fig 1).

\[
\text{Classifiers diagram}
\]

\[
\text{Bayesian network}
\]

### 4. Experimental Results

The experiments consisted of isolated handwritten digits recognition of the NIST database ([3]). This is a large base formed by 341858 binary images with resolution of 300 dpi, distributed in three subgroups: hsf_0123 (223123 images) used for training and validation and hsf_4 (58646 images) and hsf_7 (60089 images) used for test.
The proposed method, named BNE (Bayesian Network Ensemble), used linear Perceptron and 3-NN as component classifiers. It was compared with three other classification schemes: a) a 5-NN; b) a MLP network (375-64-10); and c) Boosting with 5 MLPs with the same topology programmed with the Torch 3 library (Collobert [1]). In these experiments 60000 images were used for training and the whole hsf7 subset were used for testing. In order to compare quantitavely those methods a Kappa coefficient was calculated, largely used for comparing classifiers in image processing literature. The results are shown in Table 4.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Rec. (%)</th>
<th>Kappa (%)</th>
<th>σ² (10⁻²)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLP</td>
<td>97.94</td>
<td>97.7</td>
<td>3.39</td>
</tr>
<tr>
<td>Boosting</td>
<td>98.08</td>
<td>97.86</td>
<td>2.46</td>
</tr>
<tr>
<td>5-NN</td>
<td>98.57</td>
<td>98.4</td>
<td>2.9</td>
</tr>
<tr>
<td>BNE</td>
<td><strong>98.89</strong></td>
<td><strong>98.72</strong></td>
<td><strong>2.25</strong></td>
</tr>
</tbody>
</table>

**Table 1. Recognition’s rates**

The numbers in Table 4 reveal that the proposed system presents the best performance, being suitable for applications that deal with large datasets like the NIST database.

Another experiment was developed considering the expanded database, using 190,000 images for training. In this second experiment the BNE was trained in 6 hours, while a neural network with the same topology as the one used in past experiment has consumed more than 24 hours. The recognition rate was also better when using the BNE (99.04%). This is an expected behaviour, because in a very large database there are a large number of redundant patterns that are neglected in the posterior phases of the training process. Those patterns are not ignored when used for training a conventional neural network.

5. Conclusion

This work presents a method for combining classifiers whose outputs are probabilities. The classifiers are locally specialized. They are trained with samples of local subspaces that result of feature space segmentation. The main idea consists on identifying a classification system with a plastic architecture that depends on the problem. The complexity of the classification system grows interactively depending on the problem it is applied to. A weak classifier is used as an initial guess. If it is not good enough newbies classifiers are designed for solving ambiguities that the original one could not learn. The rule adopted for identifying the neighborhoods associated to local classifiers is based on an original development stated on Lemma 1.

There is also an important contribution raised in this work. Since the individual classifiers are weak, the complexity of the system is shared with the combiner. This means that there are few free parameters that need to be adjusted because the architecture of the combiner is dynamically modeled based on the problem. Nevertheless, since a Bayesian network is a tool used for estimating a complex and factored probability distribution, the overall output system minimizes the classification empirical error.

Experimental results reveal that the proposed method is suitable for training very large databases. In an OCR application it has been achieved a high recognition rate and the overall time used for training was considerably faster than other methods used for comparison.

The method was developed in standard C++ language using a modular and object-oriented framework. The modular structure favor using other kind of classifiers that will be investigated in future work.

References