On Improving the Design of Multiclass Classifiers

M.N.S.S.K.Pavan Kumar and C.V.Jawahar
Center for Visual Information Technology
International Institute of Information Technology
Hyderabad - 500019
{pavan@students.,jawahar@}iiit.net

Abstract

The paper gives a general formulation for multiclass classifier design using pairwise classifiers. Probabilistic error analysis of a Decision Directed Acyclic Graph(DDAG) Architecture has been presented, basing on apriori probabilities of the classes, and misclassification probabilities of the classifiers. A design of a DDAG has been proposed using a greedy algorithm in a special case, and a branch and bound approach in a general case.

1 Introduction

Multi class classification has lately received a lot of attention. Some of the few popular methods being one-vs-rest, one-v-one classifiers. This paper addresses the issue of computing probabilistic error estimates of a given learning architecture of multiclass classifier from fundamental principles.

Many algorithms have been proposed for two class classification problems [1]. Similar approaches could not be directly extended to the multiclass classifiers with much success. Hence, many approaches for multi class classification are based on building two class classifiers for the given data, and then combining the decisions of two class classifiers to emulate multiclass classification. There are many ways in which a given N class problem containing classes $\omega = \{\omega_1, ..., \omega_N\}$can be posed as a combination of several two class problems. Most of the popular algorithms fall under two classes – one-vs-rest and one-v-one. In one-vs-rest approaches, there are N classifiers, each trained as positive for the ω_i and negative for all the other classes $\omega - \{\omega_i\}$. One-vs-rest classifiers have been difficult to analyze theoretically, and also it is complex to train the individual classifiers when working on a large number of classes. Methods to combine classifiers and classifier decisions have been discussed in [2, 3, 4]. Much of the work on classifier combinations is concentrated on majority voting classifiers. Error analysis of a DDAG architecture using SVMs as pairwise classifiers is discussed in [5]. The current paper discusses a general formulation of the architecture dependence of multiclass classifiers and presents an algorithm to design effective classifiers using the DDAG architecture.

2 Data, Classifier and Architecture

The performance of a multiclass classifier can be affected by three components – data, classifier and the architecture, unlike the two-class case, where the problem is just data dependent. In this paper, an analysis is made at the architecture level, fixing the information obtained from data and the individual classifiers. For example, the information that is obtained from the training data can be the apriori probabilities P_i for each class ω_i, and from individual classifiers, q_{ij}, the probability of misclassification between classes ω_i, ω_j.

3 Problem Formulation

Let $\Omega = \{\Omega_1, ..., \Omega_N\}$ be a set of classes in a classification problem, and $n(\Omega) = N$. Let the set $P = \{P_1, ..., P_N\}$ represent the apriori probabilities of the classes and $Q = \{Q_1, ..., Q_N\}$ be the misclassification probabilities of the classes in C Let there be M classifiers whose decisions are to be combined in some specified manner to arrive at a class label $\Omega_i \in \Omega$ for the given input sample. Let the decisions of these M classifiers be combined using an arrangement $A \in \mathcal{A}$, where $\mathcal{A}$ is the set of all possible arrangements for a given set of classifiers under some constraints. Let us call the set $\mathcal{A}$ of arrangements following some constraints as an architecture. It can be observed that in general, Q_i does not only depend upon the misclassification probabilities of the individual q_i $1 \leq i \leq M$, but also on the architecture $\mathcal{A}$, and the arrangement $A \in \mathcal{A}$ which is used for classifying the classes. By fixing q_is and $\mathcal{A}$ for a given set of classes, an optimal way of arranging them can be found out. For example, $\mathcal{A}$ can represent a all possible arrangements of the $M = N(N-1)/2$ classifiers in the form of a DAG. A specific arrangement $A \in \mathcal{A}$ can be a particular DAG in which the M classifiers are arranged. The overall misclassification probability of a specific arrangement A in an architecture $\mathcal{A}$ can be expressed as

$$J_{\mathcal{A}}(A) = \sum_{i=1}^{n(\Omega)} P_i Q_i$$

Let $J^*_{\mathcal{A}}$ represent the value least error achievable by all possible arrangement in $\mathcal{A}$. The objective is to find out $A \in \mathcal{A}$, such that $J(A) = J^*_(\mathcal{A})$. Also given two architectures $\mathcal{A}$ and $\mathcal{B}$, sometimes it is possible to show that $J^*_{\mathcal{A}} \leq J^*_{\mathcal{B}}$, where it is advisable to use the optimal classifier that can be found from the architecture $\mathcal{A}$. The formulation also allows the M classifiers used in the decision to be of different type as the information that is needed is the individual misclassification probabilities of classifiers from which Q_is can be computed. This gives a framework for computing the total misclassification probability of an arrangement of different classifiers and finding a best way of combining their decisions, within a given architecture $\mathcal{A}$. The following section shows this analysis on a DAG architecture.

4 Error analysis of a DAG

A Rooted Binary DAG is a generalized representation of the decision trees. An example DAG for six classes can be seen in the Figure 1. Given an input sample x, it is first given to the root node of the DAG for decision. Depending on the decision at that node, the sample takes follows to either left subgraph or to the right subgraph. A decision is taken at this node considering as the root node. A sequence of decisions is taken until a leaf node is reached, where x is assigned the label corresponding to the leaf. A DAG architecture, with nodes as pairwise classifiers using SVM is first proposed in [5].

In a given set of M classifiers, it is not necessary that each classifier has to classify all the classes. This is particularly true when pairwise classifiers are used. A relevant classifier to a class ω_i is defined as the classifier which is trained to classify ω_i, and is irrelevant otherwise. A probabilistic error analysis of a DDAG architecture on N classes and using $M = N(N-1)/2$ classifiers is presented below. Each sample x given to a classifier using DAG architecture, has to follow a path, decided by the decision at each node, to reach a leaf

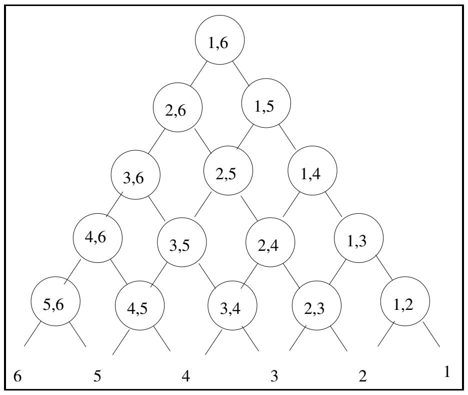

Figure 1: A 6 class DAG arrangement of pairwise classifiers

node. This path is called the evaluation path. The evaluation path, can be considered as a series connection of $N-1$ classifiers, where all the nodes need not be relevant classifiers, but once a relevant classifier is reached, all the nodes are relevant from then. It is necessary that in an evaluation path, all the relevant nodes to give correct answer for the sample to be correctly classified. A *relevant path length* is defined as the number of relevant classifiers in a path.

4.1 The Equal Probabilities Case

This section presents a probabilistic error analysis of a DDAG in a simple case, and an algorithm to design the DDAG by minimizing the error term. Let each relevant classifier on the evaluation path misclassify a given sample with a probability q. The number of evaluation paths possible for each class,i.e paths to the leaf node, are in binomial distribution. For any general node, the number paths to a node l in m^{th}layer are ${}^{m-1}C_{l-1}$. The probability of error of a class is calculated as a product of the misclassification probabilities of relevant nodes in the path, and number of such paths possible, with different relevant path lengths. Given a class i, the relevant path length j of that class can vary between 1 and $max(N-i, i-1)$. A class ω_i can be reached in $R(i,j)$ ways, where j is the relevant path length. $R(i,j)$ is given by

$$R(i,j) =^{N-j-2} C_{i-2} +^{N-j-2} C_{i-j-1}$$

${}^nC_r$ is defined to be zero when $n < r$ or $r < 0$. The correct classification probability for a class i, Q_i is given by

$$(1-Q_i) = \frac{1}{2^N} \sum_{j=1}^{max(N-i,i-1)} R(i,j) \frac{2^{N-j-1}}{(1-q)^j} \qquad (1)$$

The optimization in this case becomes maximization of

$$J = \sum_{i=1}^{N} P_i(1 - Q_i)$$

Algorithm A greedy method can be applied to solve the maximization problem. We are given N classes with apriori probabilities P. Each time a class is selected, a profit of $P_i * (1 - Q_i)$ is earned. The function can be maximized by choosing two classes with largest P_i at each step and giving them the least available relevant path lengths. It can be observed from the following lemmas show that if relevant path length increases, Q_i increases.

Lemma 1 Let x,y be maximum possible relevant path lengths of two classes ω_i, ω_j, and if

$$max(N - i, i - 1) > max(N - j, j - 1) \tag{2}$$

then $Q_i > Q_j$.

proof The lemma states that the classes having maximum relevant path length higher, have a higher chance of misclassification. Without loss of generality, assume that the $x = y + 1$. The lemma is proved by showing that $Q_i - Q_j$ is positive in this case. $Q_i - Q_j =$

$$\frac{1}{2^N}\sum_{j=1}^{y}\frac{R(y+1,j)(1-q)^j}{2^{N-j-1}} - \frac{1}{2^n}\sum_{j=1}^{y-1}\frac{R(y,j)(1-q)^j}{2^{N-j-1}}$$

$$= R(y+1,y)\frac{(1-q)^y}{2^{2N-y-1}} + \sum_{j=1}^{y-1}\frac{(R(y+1,j) - R(y,j))(1-q)^j}{2^{2N-j-1}}$$

Since $R(y+1,j) - R(y,j)$ is positive when condition (2) is satisfied, $Q_i - Q_j$ is positive.

4.1.1 Proof Of Correctness

Lemma 2 Let X be a class list, with aprioris P_i, There exists an optimal class order, where classes x, y of least apriori probability have the highest relevant path length.

Proof Let Y be the class order, where the classes a, b have the highest relevant path length, and let $P_b > P_a > P_y > P_x$. Let Y' be a solution where the classes a, b have a lesser path length than the classes x, y.

$$\begin{aligned} & J(Y) = (P_b + P_a)Q_n + (Py + Px)Q_{n-1} \\ & J(Y') = (P_y + P_x)Q_n + (Pb + Pa)Q_{n-1} \\ & J(Y) - J(Y') \\ = \; & (P_b + P_a)(Q_n - Q_{n-1}) + (Py + Px)(Q_{n-1} - Q_n) \\ = \; & ((P_b + P_a) - (Py + Px))(Q_{n-1} - Q_n) \le 0 \end{aligned} \tag{3}$$

This is always negative because $(P_b + P_a) - (P_y + P_x)$ is positive from the assumption, and $Q_{n-1} - Q_n$ is negative from lemma 2. Clearly, $(P_y + P_x) - (P_b + P_a)$ is negative, from the assumption. This means, $J(Y) < J(Y')$. This shows that Y is a better solution than Y'.

Lemma 3 Let Y be the complete order of the classes. Let the last two classes, i.e the ones with highest probabilities be removed. The remaining order Y' is optimal for the remaining classes. In the algorithm, pairs are being selected from a sorted set. Even if a pair, is removed the tree still remains optimal, because the remaining class order is sorted on probabilities.

The greedy algorithm thus produces the optimal order of classes, the proof of which is direct from lemma 3 and lemma 4.

An Example Let P={0.3,0.1,0.2,0.4} be the apriori probabilities for a 4 class problem with classes $\Omega = \{\omega_1, \omega_2, \omega_3, \omega_4\}$ one optimal DAG order for this should be the dag with the class order $\{\omega_1, \omega_2, \omega_3, \omega_4\}$. It can be observed that this ordering has the least misclassification probability when compared to others. The value of the objective function obtained for the class order $\{\omega_1, \omega_4, \omega_3, \omega_2\}$ is 0.8892 and while for the one class order $\{\omega_1, \omega_2, \omega_3, \omega_4\}$ is 0.8028.

4.2 The Unequal Probabilities Case

The above problem got extremely simplified because of the assumption of equal misclassification probabilities for all the classifier pairs. If that constraint is relaxed, the problem no longer remains greedy. A new recursive formulation of the problem and its solution is shown in this section. As an example, in Figure **??**, if a sample x is given to the node $1, 6$, then two cases arise. Case 1 : The sample x belongs to either class 1 or class 6. This can happen with a probability of $P_1 + P_6$. In this case the node is classified by the $1, 6$ classifier with an error probability q_{16}. Case 2 : The sample x belongs to neither 1 nor 6. In this case a random class is chosen. This can happen with a probability $1 - (P_1 + P_6)$. In either case, the classification probability depends on the classification probability of the sub problems, i.e the Sub-DAGs from classes 1,5 and 2,6. Let i, j be the

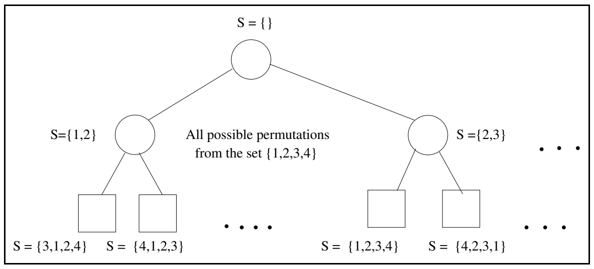

Figure 2: Branch and Bound Algorithm for creating optimal DAG

indices of the classes we are introducing in the DAG, i.e a root node corresponding to classes ω_i, ω_j is added, along with the other nodes that have to be added along with it. Let ρ_{ij} represent the sum of apriori probabilities with respect to the subproblem. Let ψ be the solution that is already available before the addition of i, j classifier. Let ψ_r, ψ_l be the rightmost and leftmost classes in the order ψ. Now, the objective function $J(i,j)$ in terms of ρ^i_{ij}, ρ^j_{ij} can be written as,

$$\begin{aligned} J(i,j) = {} & [(1-q_{ij})\rho^i_{ij} + \frac{1}{2}(1-\rho_{ij})]J(i,\psi_r) \\ & +[(1-q_{ij})\rho^j_{ij} + \frac{1}{2}(1-\rho_{ij})]J(\psi_l,j) \end{aligned} \tag{4}$$

where

$$\rho^i_{ij} = \frac{P_i}{\sum_{k=i}^{j} P_k}$$

and

$$\rho_{ij} = \rho^i_{ij} + \rho^j_{ij}$$

Using this formulation, the objective is to find the class order which minimizes this expression.

Algorithm The objective here is to maximize J, which represents the correct classification probability of the DAG. The problem is solved using the depth first branch and bound algorithm, by posing it as a minimization problem of $-J$. In this algorithm, the optimal solution minimizing J, can be reached by following a series of decisions. A depth first state space tree generation for a sample four class problem is shown in Figure 2. The S in the figure represents the solution obtained till that point in the search. The initial node represents the state where no decisions have been taken. At this point, a decision is made in the form of choosing an ordered pair of classes from the list of N classes available, and can be done in ${}^nP_2$ ways. This is represented as the second level in the tree shown in the figure 2. Selection of a node classifying between two classes ω_i, ω_j, restricts the search to the selection of only the nodes classifying between the remaining classes in the subproblem. Each node in the tree of figure 2, shows a sub problem, where a decision has to be taken from the classifiers classifying between the remaining classes. This progresses recursively, until a solution is reached, which are shown as square boxes in the figure. When the algorithm reaches the first solution node, the value of the objective function is recorded. This is used as the bounding criterion for the searches proceeding from other nodes to the answer nodes, and is updated when a better solution is found. It can be easily observed that the J is a decreasing function of number of classes, i.e as more number of classes are added to the set, the classification accuracy of the whole class set reduces. i.e -J is an increasing function. This algorithm yields in an optimal arrangements of the DAG. The final arrangement of the DAG obtained is optimal.

5 Conclusion

The performance of an ensemble of pairwise classifiers to emulate multiclass classification system depends on parameters from data, individual pairwise classifiers, and the architecture used. A novel formulation for design of multiclass classifier architectures is presented basing on this idea. An algorithm for designing optimal multiclass classifiers using the DDAG architecture are proposed.

References

[1] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification and Scene Analysis.* Wiley, 1973.

[2] Sharkey AJC and Sharkey NE, "Combining artificial neural nets: ensemble and modular multi-net systems," *Springer-Vorlag, Berlin*, 1999.

[3] H. J. Kang , K. Kim, and J. H. Kim , "A framework for combining of multiple classifiers at an abstract level," in *Engineering Applications of Artificial Intelligence*, vol. 10,4, pp. 379–385, 1997.

[4] T. K. Ho, J. J. Hull, and S. N. Srihari, "Decision combination in multiple classifier systems," in *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 16, pp. 66–75, 1994.

[5] John.C.Platt, N. Cristianini, and J. Shawe-Taylor, "Large margin dags for multi-class classification," in *Advances in Neural Information Processing Systems 12*, pp. 547–553, 2000.