GENETIC PROGRAMMING FOR MULTI-CATEGORY PATTERN CLASSIFICATION

Kaihua Huang

Department of Electrical and Computer Engineering
Mississippi State University
Mississippi State, MS 39762 USA
email: huang@isip.msstate.edu

ABSTRACT

This paper explores the feasibility of applying genetic programming (GP) to multi-category pattern classification problem. Multicategory pattern classification has been done traditionally by using the maximum likelihood classifier (MLC). GP-based techniques have an advantage over statistical methods because they are distribution free, i.e., no prior knowledge is needed about the statistical distribution of the data. GP can discover relationships among observed data and express them mathematically. GP also has the ability to automatically discover the discriminant features for a class. In this paper, a methods for GP-based -class pattern classification are developed. The experiment result for one approach is provided.

1. INTRODUCTION

Classification has been done traditionally by the maximum likelihood classifier (MLC). Although MLC can be used with any likelihood function, a normal distribution is generally assumed for the input data because it leads to minimum classification error. The basic drawback of the maximum likelihood classification is that a distance-based approach for classification is adopted and a normal distribution is assumed for the input data. Moreover, in a multi-category pattern classification problem, apart from assigning a class to a given input feature vector, there is a need to discover the underlying relationship among data and express it in an understandable manner.

Genetic programming (GP) is gaining attention due to its ability to discover the underlying data relationships and express them mathematically. Although GP uses the same principles as genetic algorithms (GAs), it is a symbolic approach to program induction, i.e., it involves the discovery of a highly fit computer program from the space of computer programs that produces a desired output when presented with a particular input. The major considerations in applying GP to multi-category pattern classification are listed below.

- GP-based techniques are data distribution free, i.e., no a priori knowledge is needed about the statistical distribution of the data or no assumption is made as in MLC.
- GP can operate directly on the data in their original form.
- GP can detect the underlying relationship that exists among data, and express it as a mathematical LISP expression. The generated LISP expressions can be used directly in the application environment.
- GP can discover the most important discriminative features of a class.

This paper proposed an approach applying GP in multi-category pattern classification and investigate its feasibility. Section 2 describes the methods used in training process and section 3 touches on an algorithm for evaluation. Experiments and analysis are presented in section 4 and 5 respectively.

2. GP-BASED TRAINNING

This section addresses the questions that arise applying GP to an n-class pattern classification problem during training process. Let

$\mathbf{F} = f_1, f_2, ..., f_n$ be the set of functions
$\mathbf{T} = X_1, X_2, ..., X_n$ be the set of terminals.

The functions in the function set may include:

- Arithmetic operations
- Mathematical functions (such as SINE, COS, EXP, LOG);
- Boolean operators (such as AND, OR, NOT);
- Conditional operators [such as IF LESS THAN OR EQUAL TO (IFLTE)];
- User-defined domain-specific functions.

The set of possible structures, such as computer programs in GP, is the set of all possible compositions of functions that can be composed from **F** and **T**. GP begins with a population of randomly created computer programs. Each computer program represents a potential solution. GP maintains a population of solutions to a given problem. During every generation, the fitness of each solution is evaluated, and for the next generation, the solutions are selected based on their fitness. The choice of functions, terminals, and the fitness function depend upon the problem. The population evolves over a number of generations through the application of variation operators, such as crossover and mutation. A genetic programming classifier expression (**GPCE**) which get best fitness will be generated when the Termination criterion is matched and is evolved as discriminant function for classes.

Suppose we have **m** feature vectors $\mathbf{x_1},\mathbf{x_2},...,\mathbf{x_m}$with dimension of **d** for a **n** class problem. we will face following issues while applying GP:

2.1. Domain-specific functions

There are many choices to define a domain-specific functions for multi-category classification problem. However, two basic rules should be followed. First of all, this function should indicate enough information of training data. Secondly, reduce computational load as much as possible since the domain-specific functions will be used for each GPCE of every generation. The effectiveness of these function will dramatically affect the performance of GP training procedure and the training time as well. We define a domain-specific function as simple as a return operation.

For the static classification problem, the domain-specific function return the $\mathbf{i}^{th}$ element of feature vector $\mathbf{x_j}$.

```
float getFeaElement (float arg1){
gFeaIndex = argument1;
((gFeaIndex > FEA_NUM)||(gFeaIndex < 1)) ?
1:gFeaIndex;
return gFea[gVecIndex][gFeaIndex];
}
```

For the temporal modeling problem, the domain-specific function return the $\mathbf{i}^{th}$ element of feature vector $\mathbf{x_{j,k}}$(the $\mathbf{k}^{th}$ vector in the $\mathbf{j}^{th}$ vector set).

```
float getFeaElement (float arg1,float arg2) {
gFeaIndex= argument1;
gSeqIndex = argument2;
gFeaIndex =
((gFeaIndex > FEA_NUM)||(gFeaIndex < 1)) ?
1:gFeaIndex;
gSeqIndex =
((gSeqIndex >= SEQ_NUM)||(gSeqIndex < 0)) ?
0:gSeqIndex;
return gFea[gVecIndex][gSeqIndex][gFeaIndex];
}
```

2.2. Fitness Measure

GP is guided by the fitness function to search for the most efficient computer program to solve a given problem. A simple measure of fitness has been adopted for the pattern classification problem. Let
f = fitness,
c = number of samples classified correctly,
m = number of samples used for training during evolution.
So, the fitness will be

$$f = 1 - \frac{c}{m} \quad (1)$$

2.3. Classification rule

As we know, the output and input of GPCE and subset of GPCE will be a float number. Intutively,We need to map the output of GPCE to the a class label. To build the range of output value and thresholds of each class, we need to compute the maximum and minim element of all training vectors. Let

l = maximum feature element,
s = minimum feature element,
n = total number of classes

The thresholds of class **i** will be;

$$\left[s + (i-1) \times \frac{l}{n}, s + i \times \frac{l}{n} \right) \quad (2)$$

If a GPCE outputs a result within the thresholds of class i, the corresponding feature vector will be classified as category i.

3. COMMITTEE-BASED EVALUATION

Another thing make GP-based classifier much different from other classifier is that GP classifier can make use of GPCE committee for evaluation. In evaluation, each GPCE from the committee will vote the feature vector for one class with a wight. Since GP simulate process of evolution, it not necessary that all survivals (GPCEs) will be the same, however some of them are close to each other. Therefore, after each training process, we can hopefully get a new CPCE. we can decide whether or not to choose this CPCE by a simple similarity criterion. If this CPCE classifies our training data set with 80% (or any value you can set) labels being the same to any of GPCEs we already selected, we will discard this one.

Also, we can use a weigh for each CPCE during evaluation process.Let

$\mathbf{w_{i,j}}$ = weigh of the $\mathbf{i}^{th}$ GPCE votes for the class j.
$\mathbf{c_{i,j}}$ = number of the $\mathbf{i}^{th}$ GPCE votes feature vectors in training set for j class.
$\mathbf{n_{i,j}}$ = number of the $\mathbf{i}^{th}$ GPCE correctly votes feature vectors in training set for j.

$$w_{i,j} = \frac{c_{i,j}}{n_{i,j}} \quad (3)$$

Let $X = \{x_1, x_2, \ldots, x_m\}$be a given evaluation set of **m** vectors in a **s**-dimensional feature space.After GP-based training process, we got a Committee of **l** GPCEs and a $\boldsymbol{l \times n}$ wight matrix $\mathbf{W} = \{\boldsymbol{w_{i,j}}\}$ for **n** classes. The procedures of the Committee-based evaluation algorithm are described as follows:

Algorithm: (Committee-based evaluation)

1 begin initialize X,W, l GPCEs,
2 for each vector in X
3 for each GPCE in committee
4 Compute the votes of this CPCE
5 select the class label with maximum votes as the label for this feature vector
6 return labels of all evaluation vectors
7 end

4. EXPERIMENTS

Currently, I have finished the baseline experiments on two type of data sets as follows:

Table 1: experiment data sets

	Set 1	**Set 2**
Model	static classification	temporal modeling
Dimension	10	39
Classes	11	5
Training set	528	925*
Test set	379	350*

*: Sets of 5 for each class

I implemented the GP classifier based on GPQuick[2] source code. The parameters for GP are showed as follow:

Table 2: Parameters for the GP

Parameter	**Weightage**
Crossover weightage	0.70
Mutation weightage	0.20
Crossover weightage annealing	0.50
Mutation weighage annealing	1.00
Copy weightage	0.04
Mutation rate	0.60
Crossover rate	o.70
Mutation node	0.435
Mutation constant	0.435
Mutation shrink	0.30
Selection strategy	Tournament
Tournament size	6

The function set I used for GP include all the functions list in section 2 except Mathematical

functions which take much more computational time.The termination criterion for GP-based training is more than 2,000,000 generations or get a error rate less than 10%.But up till now, I haven't get any GPCE with an error rate less than 10% on training set yet. The GP-based training process took quite a lot of time since it try to learn the underlying relation between data set through a way near to random guessing, but from the log file of the training procedure I find that it do evolve in a direction of getting better. I have trained a GPCE more than 3*24 hours but still have a error rate greater than 30%. The current experiment result shows the error rate based on the GPCEs I already obtained from training as follow:

Table 3: . Error rate on test data set

Committee number	Error Rate	
	Set 1	Set 2
1	**68.86**	90.00
7	70.97	72.86
17	72.29	65.74
27	71.76	**64.28**

Because of using GPCEs with high error rate (more than 60%) on the training set 1, the error rate doesn't decrease much as the number of committees increase. But for the data set 2, we can see the there is a trend of decreasing as the number of committee increase.

5. ANALYSIS

Obviously, the experiment result didn't support the GP-based multi-category classifier well.However, behind those numbers, there are several important issues. First of all, these evaluation results are based on high error rate GPCEs because the huge time for training prevents us to get perfect GPCE in a short amount of time. Hence, the time for training becomes the biggest obstacle on the way of applying GP to multi-category pattern recognition. Secondly, the choice of the GP parameters in table 2 was basically empirical. How these parameters will factor the time of training is unknown to us. Thirdly, how to choose the function set and define the domain-specific functions is another important factor might affect the training process.

Although problematically, the GP-based multi-category classifier is still promising. There are quite a lot of improvement can be made for current approach. The present classification rules are not suitable for a GP process. It is not a good way to classify the feature vectors based on a m-m mapping mechanism because the complexity of underlying relationship, which makes GP very hard to learn and results in enormous training time. Future work might lie in applying the two-category classifier technique to GP[3] and build GP-based classifier based on two-category classifier and decision tree since two-category relationship is much easier for the GP to learn. On the other hand, since we can make use of the training data as many times as we want, the training process for GPCEs can be proceeded paralleled in different machine.

Also, the committee-based evaluation is another attractive novel approach although the experiment result didn't provide much support for that. Nevertheless, hopefully, we can expect each GPCE having learned part of the knowledge from the training data and contribute their knowledge during evaluation as long as we get low error rate GPCEs during training process. But how the number of committee will affect the performance of classifier and what is the suitable quota of committee for a certain number of classes are still need to be further study on.

6. CONCLUSIONS

This paper investigate the feasibility of applying GP to multi-category pattern classification problem. A training method and an evaluation algorithm are presented.The experiment results, however frustrating, are provided. Analysis on the current experiment result and algorithm are extended. In summary, GP methods for multi-category classification is quiet promising but still not practical right now.

7. REFERENCE

[1] R. O. Duda, P. E. Hart and D. G. Stork, Pattern Classification, Wiley-Interscience, 2001

[2] A. Singleton, "Genetic programming with C++", Byte, pp. 171-176, Feb. 1994

[3] J. K. Kishore, L. M. Patnaik, V. Mani, V. K. Agrawal, “Application of Genetic Programming for Multicategory Pattern Classification,” Evolutionary Computation, IEEE Transactions on, Volume: 4 Issue: 3, pp.242-258, Sept. 2000