GP Boosting Classification on Concept Drifting Data Streams

Dirisala J. Nagendra Kumar[1], J.V.R. Murthy[2],
Suresh Chandra Satapathy[3], and S.V.V.S.R. Kumar Pullela[4]

[1] BVRICE, Bhimavaram, India
[2] JNTUCE, Kakinada, India
[3] ANITS, Visakhapatnam, India
[4] VS Lakshmi Engg. College, Kakinada
{nagendrakumardj,mjonnalagedda,
sureshsatapathy,ravipullela}@gmail.com

Abstract. Genetic Programming is an evolutionary soft computing approach. Data streams are the order of the day input sources. In general, data streams exhibit a peculiar behavior of drifting the concepts as time passes by. Here is a study of GP Classifier on Concept Drifting Data Streams. GP classifier performance is compared to that of other state-of-the-art data mining and stream classification approaches. Boosting is a machine learning meta-algorithm for performing supervised learning. A weak learner is defined to be a classifier which is only slightly correlated with the true classification. In contrast, a strong learner is a classifier that is arbitrarily well-correlated with the true classification. Boosting combines a set of weak learners to create a strong learner. It is observed that the Boosting GP approach is beating Boosting Naïve Bayes classification on Concept Drifting Data Streams. Hence it is found that GP is a competent algorithm for Concept Drifting Data Stream classification.

Keywords: Genetic Programming, Classification, Multi-class, Boosting, Data Stream, Stream Mining, Concept Drifting Data Stream.

1 Introduction

Many organizations are being dumped with tremendous amount of continuous flow of data, due to a sequence of events from different locations of the organization. Telephone records, credit/debit card transactions, sensor networks, network event logs, web log data, online sales transactions are some examples of data streams. Traditional approach for mining data is known as batch processing, as it assumes data as a static entity. Now data streams stresses the need of online and incremental data mining techniques, of course should be able to deal with concepts drifts in some cases.

Classification is a major data mining technique [6][13]. Genetic Programming (GP) is one of the famous classification techniques, which has its roots in Genetic Algorithms (GA) [7-9]. Bagging and Boosting are two meta learners in data mining [6][12][13].

S.C. Satapathy et al. (Eds.): Proceedings of the InConINDIA 2012, AISC 132, pp. 265–272.
springerlink.com

Data stream classification is studied in [15-20]. Mohammad M. Masud et. al. studied Mine Class Algorithm for automatic detection of a novel class in presence of concept-drift [15]. Gianluigi Folino et. al. has studied a StreamGP approach with adaptive boosting ensemble algorithm for classifying homogeneous distributed data streams [16]. Hussein A. Abbass et. al. made a detailed study of online adaption in learning classifier systems for stream data mining based on Genetic Algorithms [17]. Yi Zhnag and Xiaoming Jin built an ensemble classification technique on data streams [18]. New ensemble methods for evolving data streams are studied by Albert Bifet et. al.[20]. Wenyan Wu and Le Gruenwald studied various issues involved in simultaneous mining of multiple data streams [19].

Most of the work on classification concentrates on binary classification problems. Traditionally Maximum Likelihood Classifier (MLC) [10], Bayesian networks [10], and Neural networks (NN) [11] are the most successful approaches for multi-class classification.

Genetic Programming (GP) is a stochastic approach, derived from Genetic Algorithms (GA), to solve various computer related problems by automatically constructing programs simulating the biological evolution [8]. GP is a nice approach for solving the binary and multi-class classification problems. It guarantees good classification accuracy if enough training time is given to evolve a higher accuracy GP classifier [2]. An attempt is made to reduce this training time to a reasonable degree. The goals that are tried to meet are simplicity, scalability, and high accuracy. The GP classifier has to find fitness for all fitness cases, which may not be stored in main memory for larger datasets. In order to achieve scalability, the size of training data set sampled at a time is restricted to a portion of main memory available. Topon Kumar Paul, and Hitoshi Iba [5] implemented the ensemble approach of Boosting based on GP and called it "a majority voting genetic programming classifier".

T. Loveard and V. Ciesielski [1] proposed five alternative methods to perform GP-based multi-class classification, viz., Binary decomposition, Static range selection, dynamic range selection, class enumeration and evidence accumulation.

J. K. Kishore et al.[2] modeled the n-class pattern classification problem as an n two-class problems. A Genetic programming classifier expression (GPCE) is evolved as a discriminant function for each class. Each GPCE recognizes data samples belonging to its own class and rejects samples belonging to other classes. In [2]-[4], [14] the authors designed a classifier with n binary-trees for the n-class classification problem.

D.P. Muni et al. [3] improved the approach of J.K. Kishore et al.[2] by generating the classifier in one pass. D.P. Muni et al. extended their earlier work to suit for feature selection (FS) in [4], proposing a wrapper approach for FS.

Topan Kumar Paul, and Hitoshi Iba [5] proposed a majority voting technique, which evolves multiple GP rules and apply those rules to test samples to determine their labels and count their votes in favor of a particular class. Then the sample is assigned to the class that gets the highest number of votes in favor of it.

T. Loveard and V. Ciesielski[1] used the total training set as exemplar set. In [3], [4], D.P. Muni et al. used step-wise learning, which takes a smaller exemplar set initially, and gradually increases the exemplar set to the whole training set.

2 Data Streams

The recent advances in hardware and software have enabled the capture of various measurements of data in a wide range of fields. These measurements are generated continuously and in a very high fluctuating data rates. Examples include sensor networks, web logs, and computer network traffic. The storage, querying and mining of such data sets are highly computationally challenging tasks. Mining data streams is concerned with extracting knowledge structures represented in models and patterns in non stopping streams of information. The research in data stream mining has gained a high attention due to the importance of its applications and the increasing generation of streaming information. Applications of data stream analysis can vary from critical scientific and astronomical applications to important business and financial ones. Algorithms, systems and frameworks that address streaming challenges have been developed over the past decade. There is a real need inspired by the potential applications in astronomy and scientific laboratories as well as business applications.

3 GP Boosting Approach for Data Stream Classification

Once a GP classification program predicting the class labels is built, it can be used directly for classification, or combine the GP programs into a more efficient solution. There are several ways to combine classifiers. For example, one can use a voting system for the results of several classifiers.

In the case of a classification problem with n classes, there are several approaches to build classifiers. The three most common approaches are:

1. Develop a single classifier that gives, as output, the class of the new sample as input.
2. Develop n classifiers. Each classifier is responsible for recognizing a particular class.
3. Develop a classifier for each pair of classes. Each classifier is responsible to decide between two classes in particular.

The method 2, n-classifier approach, is the best approach [23]. Thus a variant of method 2 is used in this study. With this method, the classifiers obtained by the GP must have some type of output value. Two approaches were again proposed:

1. Binary classifier

If the classifier result is 0, it predicts that the sample is not part of the class, or if 1 predicts that the sample belongs to the class.

2. Classifier with Continuous Output

The result is a decimal value (eg between 0.0 and 1.0) that represents the confidence with which the classifier links the sample with the designated class.

When a new sample is introduced, each classifier must predict whether the sample belongs to the class for which it was trained. The combined classifier has the output value that determines the largest class of the new sample. In the event of a tie, the classifier that has the highest probability will be identified as the class of new sample.

The present work integrates the boosting meta learner with the evolutionary process of GP. Boosting algorithm is applied on n-class GP classifiers. Here each classifier predicts the confidence with which the classifier is assigned the class. Several studies on the implementation of a method for boosting the GP have reported significant gains in terms of classifier accuracy and computation time of the algorithm [21-22]. The integration of the principles of boosting even within the GP process allows greater economy of resources. Here is the pseudo code of the Boosting GP approach adapted here:

C = number of classes of the problem
P = number of necessary programs for boosting
Training set, T = all training data available
N = total number of samples in T
For all C_j (j = 1 to C)
    Empty the GP population, POP
    Initialize the weight of each sample W with $W_i = 1/N$
    For all P_k (k = 1 to P)
        If POP is empty, fill POP with a new set of programs.
        Changing a program that recognizes the class C (the calculation of fitness uses the weight W_i of each sample to classify), using T and POP.
        Calculate the error of the best program, E_{jk} on the training set, a factor α_{jk} and then the weight of each sample W_i using AdaBoost method.
    End for P
End for C

A sample can be classified using the strongest response in a weighted sum of the outputs of programs by class (using equation 1).

$$\max \left(\sum_{k=1}^{P} \left(a_{jk} * \alpha_{jk} \right) \right) \tag{1}$$

By the end of the routine, P*C programs (where P programs for each class of the C classes) are obtained. The classification score for class j is obtained by the weighted sum by α_{jk} output of each program jk. The class that scores the highest indicates the class of the given sample:

4 Fitness Function for GP Boosting Approach

The fitness is the measure of GP program performance in the prediction of output values from input samples. It is therefore an indication of relevance of the program for classifying the samples in training dataset. Fitness is a numeric value, allowing us to compare the performance of programs. Fitness is used to select programs in the population to transform further.

Fitness function is the result of classifications on the training data. This function compares the value of predicted class and actual class provided in the training data. The fitness function depends on the approach used in the combination of classifiers.

1. For a single classifier approach, fitness is simply the number of correct predictions of the program. This value can be normalized (between 0.0 and 1.0) by dividing the number of matching samples in the data set by the total number of samples in training dataset.
2. In the case of an approach of n-class classifier, the calculation of the fitness depends on the classifier chosen:

a) Binary Classifier: It is as in the single classifier approach.
b) Classifier with Continuous Output: The output of the program P is a value of limited trust between -1.0 and 1.0. Fitness is calculated from the sum of S values of confidence of P_i for each sample, depending on the class C provided by the training data set.

$$S = \sum_j P(i) * C(i) \tag{2}$$

C (i) is 1.0 if the sample i belongs to the class recognized and -1.0 otherwise. Finally, fitness is the sum of S values, normalized between 0.0 and 1.0.
c) Classifier output continues to boosting algorithm built: The technique is essentially the same as (b), but the weight W of training samples is taken into account:

$$S = \sum_j P(i) * W(i) * C(i) \tag{3}$$

As the weight of the samples is also normalized (total weight is 1.0), the sum S can be normalized in the same way as in (b). So for a classification problem, the more fit, the more the program is effective. A perfect prediction rate is obtained when the fit is 1.0. Here 2(c) approach is followed. Every Genetic Programming approach needs some parameters to be specified. In this approach, the GP parameters used are given in Table 1.

Table 1. The default GP parameters used for GP Classifier Construction

Parameter	Values
Population size	100
Maximum depth	5
Stopping Criteria	Fitness=99%, Max. Generations=100, Max. Time=5 min.
Population Initialization	Ramped-half-and-half
Selection	Roulette wheel
GP operator proportions	Crossover=90%, Mutation=7%, New Program=3%

5 Results

The data on which the classifiers are executed are 2-class and 5-class Concept Drift Random trees each with 10 Million rows and the evaluation is through interleaved test then train evaluation. The result of GP classification on the above datasets is as follows:

Table 2. The time taken and classifier accuracy % of various classifiers on 2-class Concept Drift Random trees

Classifier	Functions	Time in sec.	Accuracy %
AdaBoost M1+ NB	**--**	1h4m55s	72.02
GP	+, -, *, /	59m38s	73.01
GP	+,-,*,/,If, <, >	1h4m34s	73.85
GP	If, <, >	31m18s	57.82
GP	If,<,>,!,&,|	30m32s	71.17

Table 3. The time taken and classifier accuracy % of various classifiers on 5-class Concept Drift Random trees

Classifier	Functions	Time in sec.	Accuracy %
Adaboost M1+ NB	--	1h40m19s	54.20
GP	+, -, *, /	1h42m37s	54.73
GP	+,-,*,/,If,<,>	2h5m12s	45.31
GP	If, <, >	50m28s	37.54
GP	If, <, >, !, &, |	1h20m11s	43.43

In case of the above 2-class Random tree dataset, Boosting GP with +,-,*, /, If, <, > functions classifying with 73.85% accuracy is better than that of the combination of AdaBoostM1 and Naïve Bayes Classification with 72.02% accuracy. And for 5-class Random tree dataset, Boosting GP with functions +, -, *, / classifying with 54.73% accuracy is above that of the combination of AdaBoostM1 and Naïve Bayes classifier with 54.20% accuracy. The only disadvantage is that there is no single combination of GP functions and parameters suitable for all datasets. Hence in general, this Boosting GP is a good candidate for Concept Drifting Stream classification and is suitable for further work.

6 Conclusions

It is found that Boosting GP Classifier is a competent approach for classifying concept drifting data streams. The issue is changing accuracies of GP classifier with functions and GP parameters. The next goal is to improve the GP approach in two respects: accuracy and reducing execution time. Trying various proportions of GP functions like crossover, mutation, selection, and etc, may result in better accurate GP programs. Applying some statistical methods like Principal Component Analysis(PCA) as preprocessing step and applying some clustering, like Expectation Maximization clustering, may make this approach faster. The further research work will be in the above direction.

References

1. Loveard, T., Ciesielski, V.: Representing classification problems in genetic programming. In: Proc. Congr. Evolutionary Computation, May 27-30, pp. 1070–1077 (2001)
2. Kishore, J.K., Patnaik, L.M., Mani, V., Agrawal, V.K.: Application of genetic programming for multicategory pattern classification. IEEE Transaction on Evolutionary Computation 4, 242–258 (2000)
3. Muni, D.P., Pal, N.R., Das, J.: A novel approach for designing classifiers using genetic programming. IEEE Trans. Evolut. Comput. 8(2), 183–196 (2004)
4. Muni, D.P., Pal, N.R., Das, J.: Genetic programming for simultaneous feature selection and classifier design. Systems, Man, and Cybernetics, IEEE Transactions on Systems, Man, and Cybernetics, Part B 36(1), 106–117 (2006)
5. Paul, T.K., Iba, H.: Prediction of Cancer class with Majority Voting Genetic Programming Classifier Using Gene Expression Data. 2009 IEEE/ACM Trans. on Computational Biology and Bioinformatics 6(2), 363–367 (2009)
6. Han, J., Kamber, M.: Data Mining Concepts and Techniques, 2nd edn. Elsevier (2006)
7. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading (1989)
8. Koza, J.R.: Genetic Programming: On the programming of Computers by Means of Natural Selection. M.I.T. Press, Cambridge (1992)
9. Poli, R., Langdon, W.B., McPhee, N.F.: A field guide to Genetic Programming (March 2008), http://www.gp-field-guide.org.uk
10. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. John Wiley and Sons (2001)
11. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning internal representation by error propagation. In: Rumelhart, D.E., McClelland, J.L. (eds.) Parallel Distributed Processing. MIT Press (1986)
12. Breiman, L.: Bagging predictors. Machine Learning 24, 123–140 (1996)
13. Tan, P.-N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Person Education (2006)
14. Nagendra Kumar, D.J., Satapathy, S.C., Murthy, J.V.R.: A scalable genetic programming multi-class ensemble classifier. In: World Congress on Nature & Biologically Inspired Computing, NaBIC 2009, pp. 1201–1206 (2009), doi:10.1109/NABIC.2009.5393788
15. Masud, M.M., Gao, J., Khan, L., Han, J., Thuraisingham, B.: Integrating Novel Class Detection with Classification for Concept-Drifting Data Streams. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) ECML PKDD 2009. LNCS, vol. 5782, pp. 79–94. Springer, Heidelberg (2009)
16. Folino, G., Pizzuti, C., Spezzano, G.: An Adaptive Distributed Ensemble Approach to Mine Concept-Drifting Data Streams. In: ICTAI 2007 Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence, vol. 02 (2007)
17. Abbass, H.A., Bacardit, J., Butz, M.V., Llorà, X.: Online Adaptation in Learning Classifier Systems: Stream Data Mining (2004)
18. Zhang, Y., Jin, X.: An automatic construction and organization strategy for ensemble learning on data streams. ACM SIGMOD Record Homepage archive 35(3) (September 2006)
19. Wu, W., Gruenwald, L.: Research issues in mining multiple data streams. In: Stream KDD 2010 Proceedings of the First International Workshop on Novel Data Stream Pattern Mining Techniques (2010)

20. Bifet, A., Holmes, G., Pfahringer, B., Kirkby, R., Gavaldà, R.: New ensemble methods for evolving data streams. In: 15th ACM SIGKDD Intl. Conference on Knowledge Discovery and Data Mining (KDD 2009), Paris, France (June 2009)
21. Folino, G., Pizzuti, C., Spezzano, G.: Boosting Technique for Combining Cellular GP Classifiers. In: Keijzer, M., O'Reilly, U.-M., Lucas, S., Costa, E., Soule, T. (eds.) EuroGP 2004. LNCS, vol. 3003, pp. 47–56. Springer, Heidelberg (2004)
22. Paris, G., Robilliard, D., Fonlupt, C.: Genetic Programming with Boosting for Ambiguities in Regression Problems. In: Ryan, C., Soule, T., Keijzer, M., Tsang, E.P.K., Poli, R., Costa, E. (eds.) EuroGP 2003. LNCS, vol. 2610, Springer, Heidelberg (2003)
23. Teredesai, A., Govindaraju, V.: Issues in Evolving GP based Classifiers for a Pattern Recognition Task. In: Proceedings of the 2004 IEEE Congress on Evolutionary Computation, pp. 509–515. IEEE Press (2004)