Optimizing Twins Decision Tree Classification, Using Genetic Algorithms

Farid Seifi(farid@comp.iust.ac.ir), Mohammad Reza Kangavari(kangavari@iust.ac.ir)
Department of Computer Engineering, Iran University of Science and Technology, Iran.

Hamed Ahmadi(hank.ahmadi@gmail.com)
Department of Automated Systems Software, National Aerospace University of Kharkiv, UKRAINE.

Ehsan Lotfi(esilotf@gmail.com), Sanaz Imaniyan, Somayeh Lagzian(shiva.sehat@gmail.com)
Department of Computer Engineering, Islamic Azad University of Mashhad, Iran.

Abstract— **Decision tree classification is one of the most practical and effective methods which is used in inductive learning. Many different approaches, which are usually used for decision making and prediction, have been invented to construct decision tree classifiers. These approaches try to optimize parameters such as accuracy, speed of classification, size of constructed trees, learning speed, and the amount of used memory. There is a trade off between these parameters. That is to say that optimization of one may cause obstruction in the other, hence all existing approaches try to establish equilibrium.**

In this study, considering the effect of the whole data set on class assigning of any data, we propose a new approach to construct not perfectly accurate, but less complex trees in a short time, using small amount of memory. To achieve this purpose, a multi-step process has been used. We trace the training data set twice in any step, from the beginning to the end and vice versa, to extract the class pattern for attribute selection. Using the selected attribute, we make new branches in the tree. After making branches, the selected attribute and some records of training data set are deleted at the end of any step. This process continues alternatively in several steps for remaining data and attributes until the tree is completely constructed. In order to have an optimized tree the parameters which we use in this algorithm are optimized using genetic algorithms.

In order to compare this new approach with previous ones we used some known data sets which have been used in different researches. This approach has been compared with others based on the classification accuracy and also the decision tree size. Experimental results show that it is efficient to use this approach particularly in cases of massive data sets, memory restrictions or short learning time.

I. INTRODUCTION

Decision tree classification is one of the most practical and effective methods which is used in inductive learning. Today, the main effort of scientists in this field is to construct decision trees with good accuracy [12], small-size tree, short learning-time and a small amount of memory usage. There is a trade off between these parameters. That is to say that optimization of one may cause obstruction in the other, hence all existing approaches try to establish equilibrium. There are two major branches in the field of classification, some approaches try to handle data streams [3, 4] and others work on static data sets.

It is also possible to classify the existing approaches into two categories:

1. Self-determined (independent) approaches
2. Human Intelligence-dependent approaches

In the first category, the decision tree construction algorithm receives the training data set as an input and constructs the decision tree as an output without any interaction with the user. In the second one, which is based on a multidimensional visualization technique, the data set is visualized in a comprehensible manner enabling the user to construct a decision tree [1]. The user interference in decision tree construction brings about advantages and disadvantages. The small size of the constructed tree is surely an advantage and the long waiting time for user's interaction and visualization restrictions in visualizing massive data sets are definitely disadvantages.

In the current study we introduce a new self-determined approach which has not only a suitable accuracy, but also a small-size tree which is constructed in a short time, using a small amount of memory. In this study we use static data sets. The rest of this paper is organized as follows. In section II we mention the classification goal and some problems which must be solved. Section III presents the motivation of this study. We introduce our approach to decision tree construction in section IV. The genetic algorithm which is used to finding the optimum values for classification parameters are also discussed in this section. The comparison of efficiency is discussed in section V. Section VI summarizes this paper and outlines several issues for future research.

II. DECISION TREE CLASSIFICATION

The goal of classification is, based on the attribute values of any new phenomenon, assigning a class to that phenomenon through defined classes. In fact, this work is predicting the class of new phenomenon based on the

attributes [7], [8]. There are different methods for this prediction of which decision tree classification is one [5], [10]. In the first stage of this method, a training data set is used; the rules and relations between attribute values and classes are extracted and applied in a decision tree. In the second stage, test data have to be used to estimate the accuracy of constructed decision tree. This tree could be used for classification of real-world data.

There are two main problems [5] in decision tree construction whose different solutions lead to various classification methods. One of these problems is the attribute selection for making new branches in the tree. The pruning is the other one which has to be done to omit and decrease the tree nodes.

III. Motivation

In the Human Intelligence-dependent classification methods, first the training data is sorted and visualized by using each attribute separately. These visualizations show class patterns. Through particular standards, the user is, by observing the various class patterns, to choose one. The chosen attribute, which is used for sorting the training data and making the selected pattern, is the selected attribute for classification in this stage. By observing the selected pattern, the user is, then, charged to select some domains of that attribute for making new branches [1], [2], [11]. In the following stages, the remaining data and attributes are processed in same manner. As mentioned above, human intelligence-dependent approaches have some disadvantages.

Many self-determined approaches have been constructed, but most of them concentrate on optimization of classification accuracy. Therefore, they almost make large decision trees, which are inappropriate for classification of massive datasets.

Thus, we tried to propose a self-determined approach that

- Does not have the disadvantages of Human Intelligence-dependent approaches.
- Has a very small size tree that makes it usable for the classification of massive data sets.
- Extracts the class pattern.
- Selects an attribute for making new branches itself, by considering different patterns.
- Makes new branches based on selected pattern and attribute.
- Prunes and optimizes the tree by using the accuracy of branches.

This method has been implemented by MATLAB and applied on some major datasets which their main features have been introduced in section V. We optimized the parameters of this method, using genetic algorithm. This algorithm is also mentioned in section V. This approach has been compared with others in that section based on the classification accuracy and also the decision tree size.

IV. Twins Decision-Tree Classification (TDC)

We give a detailed introduction of our new approach in this section. First of all, we discuss the decision tree learning process and after that we present the pruning and optimization operations used in this study.

A. Learning and Constructing the Decision Tree

The decision tree construction is done in various steps in TDC. In every step we use an attribute for making new branches in the tree. After making branches, the selected attribute and some records of training data set are deleted at the end of any step. This process continues alternatively in several steps for remaining data and attributes until the tree is completely constructed. The details on how to extend the tree in any step are as follows. Any step has two stages. The first stage, which is selecting one attribute for making new branches, includes:

- Sorting the data set using each attribute separately.
- Extracting the worth of the attributes by considering the balance of relations between those attributes and classes every time that the data set is sorted. This part is discussed in detail in subsection B.
- Selecting the attribute with the highest worth in order to extend the tree. This part is discussed in detail in subsection B.

The second stage is constructing new branches and deleting some samples from training data set. This stage includes:

- Specifying the domains of the selected attribute mapped to any class. Using this information to grow new branches in the tree. This part is explained in subsection D.
- Using the training data set in order to estimate the accuracy of each new branch.
- Discarding some part of the training data set in order to prevent the influence of some redundant data on creating probable new branches. These redundant data are defined those that can be classified using current new constructed branches. This also helps decreasing the data set. Note that we only discard the parts of data which can be accurately classified at least with one of the branches.

Fig. 1 shows the pseudo code of this process.

B. Determining balance of relations between any attribute and classes and selecting the best attribute

First of all, we examine the effect of changes in the values of each attribute on class patterns and then extract the existing patterns.

```
Training-Data = {a1, a2 ...}
While AttributeNum>0 Do
    For Each Attribute ATT_i Do
1.      SortedData=SortTrainingDataBy ( ATT_i)
2.      AttAdvantages[i] = DefineAdvantage ( ATT_i, SortedData)
    End For
3. SelectedAtt = CompareAdvantages (AttAdvantages)
4. NewBranches = MakeNewBranches (SelectedAtt)
5. EstimateAccuracy (NewBranches)
6. DelDataByNewBranches ()
    DelAtt (SelectedAtt)
End While
```

Fig. 1. Pseudo code of TDC's decision tree construction.

To extract these patterns, we take into account the effects of the whole data set on class assignment. An array with the length equal to the number of classes is defined for any sample of data, filled with zero as a default value.

$$Array(1:recordnum, 1:classnum) = 0 \quad (1)$$

We trace the training data set from beginning to the end and set the values of each data sample's array, by considering its class and the values in the array of previous data sample.

For this initialization the array of previous data sample is copied into the current array and then weight1[1] is added to that element of the array which is mapped to the current data sample's class and weight2 is subtracted from others.

$$Array1(i,:) = Array1(i-1,:)$$
$$Array1(i, \mathrm{Re}c_i.class) += weight1$$
$$For\ each\ j <> \mathrm{Re}c_i.class\ \ do\ \ Array1(i,j) -= weight2$$

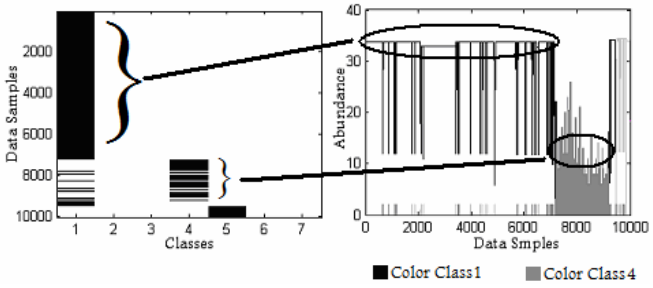
We repeat the same operation on another array[2] and in the opposite direction. It is worth to say that these twins operations made us to name this approach as TWINS D-TREE CLASSIFICATION. We defined *Aboundance_array for any data sample*, which obtained by adding array1 to array2l, as in

$$Aboundance\_Array = Array1 + Array2 . \quad (3)$$

The abundance array contains information about the number of each class around any data sample. On the other hand, it contains the class pattern. Fig. 2 shows how class abundance is related to the class patterns.

Since the class pattern is extracted when the data set is sorted by a specific attribute, it is possible to compare different attributes by comparing their class patterns which are stored in arrays. Fig. 3 shows the extracted class patterns for two different attributes in Shuttle data set.

[1] Weight1 and weght2 depends on the characteristics of data set.
[2] Named as array2



(a) (b)

Fig. 2. (a) The class pattern obtained by using an attribute. (b) The class abundance around any data sample.

To compare the attributes to select the best, the abundance array which has already been defined should be prepared for any attribute. Simply, the attribute which has the maximum value in the array elements, has the highest worth to extend the tree and is selected for extending the tree.

C. *Optimum Value for Weight1 and Weight2*

To discover the optimum value for Weight1 and Weight2 a method of genetic algorithms is hired. Chromosome consists of two gens, W1 and W2. Basic population of chromosomes is defined by data set characteristics. Each genetic algorithm must have a fitness evaluation function. In this case chromosomes which make higher Abundance array for an attribute can fit. Therefore an array of each chromosome and its related abundance is made.

In current population there are numbers of individual chromosomes without any information of probability for any of them, so tournament selection must be hired [14]. Each generation consists of n top chromosomes of the former generation and new chromosomes made by tournament selection of the selected n top chromosomes of former generation [13].

Termination of this process is when the n top chromosomes repeated for j times. In other words it is when a flat land is reached. Between these top chromosomes just one must be selected. W1 and W2 are gens of selected chromosome.

```
Repeat
    Generation=Makenegeneration(TopN)
    ChromArray=Makearray(Generation)
    TopN=SelecttopChromosoms(ChromArray,J)
   if TopN=XtopN then
        Counter++
     else
        Counter=0
       Xtopn=TopN
    Endelse
 Until Counter>J
```

Fig. 3. Pseudo code of the genetic algorithm which used for finding optimum values of Weight1 and Weight2.

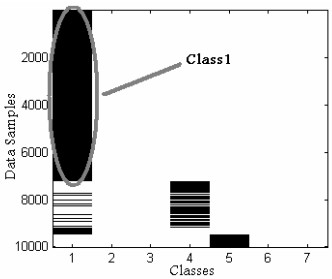


Fig. 4. Contiguous data samples which had taken the same class

An important point which should be mentioned is that gens of the chromosome are from different types so they should not be mixed during the process of tournament selection, the mixture must be done just between same gens of different chromosomes (W1or W2.)

D. Constructing New Branches based on Selected Attribute

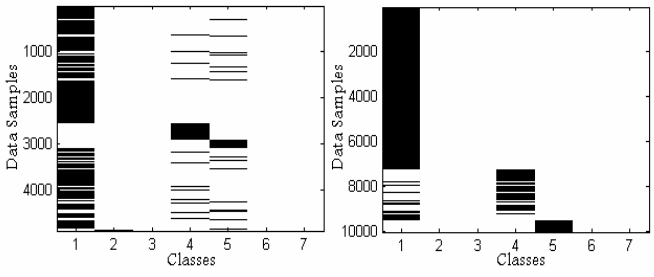
New branches are grown in the tree by using the abundance arrays which are initialized by sorting the training data set with the selected attribute. The maximum value in the array elements of any data sample shows whose abundance is the highest (most).

Therefore it is possible to assign a class to any data sample. Since the class assignment to any data sample is based on the class abundance around it the contiguous samples may take the same class. Fig. 4 shows contiguous data samples with the same assigned class. So it is possible to find a sub domain from the selected attribute which all data samples in that domain has the same assigned class. We use this attribute and class and ranges of domain to make a new branch in the tree. This can be done with all domains to make new branches.

It is noteworthy that some of these branches are not suitable and must be ignored. The pruning strategy and optimization methods used for this purpose are discussed in the next section.

E. Optimizing the Tree

As mentioned before some part of training data set is ignored after making branches. This operation prevents the influence of such data on constructing unreal branches and also decreases the data set.



(a) (b)

Fig. 5. (a) The extracted class pattern using 6th attribute. (b) The extracted class pattern using 9th attribute in shuttle data set.

TABLE 1

MAIN FEATURES OF SATIMAGE, SEGMENT, AND SHUTTLE DATASETS

Data Set	Classes	Inputs	Train Set Size	Train Set Size
Satimage	7	36	4293	1430
Segment	7	19	698	1397
Shuttle	5	9	34803	14500

Therefore, this ignorance not only decreases the tree size but also increases the accuracy of the tree and the learning speed. In fact, the classification time of massive data sets may decrease if we reduce the tree size [6].

There are many different methods for reducing the tree size referred to as pruning. In this approach we prune the branches which have a low level of accuracy or the ones which have been made by the effect of a short data set. The deletion of surplus branches increases the tree efficiency; however, it may put a paltry negative effect on the accuracy.

Finally, we sort the different branches based on their accuracy after pruning the tree. The most accurate branches are placed in the upper levels of the tree. As a result, there will be less chance for inaccurate branches to interfere with classification. This operation greatly promotes accuracy.

V. COMPARISON OF EFFICIENCY

A. Data Sets

In order to compare this new approach with the previous ones we used some known data sets such as Shuttle, Satimage, and segment data sets which have been used in different researches. The Satimage and Segment data sets are categorized in medium size datasets and the Shuttle data set is categorized in large ones. We used them to show the efficiency of our approach in case of massive data sets. The main features of these data sets are summarized in table1.

B. Results

There are various parameters which may affect efficiency of decision trees. Listed below are a few of them [9].

- Accuracy. This is the reliability of the rule and one of the most important parameters which is used for comparing different approaches. This parameter is relevant to correct classifications.
- Classification Speed. In some circumstances, the speed of the classifier is a major issue. A classifier that is 90% accurate may be preferred over one that is 95% accurate if it is say 100 times faster in testing.
- Comprehensibility. If it is a human operator that must apply the classification procedure, the procedure must be easily understood or else mistakes will be made in applying the rule.

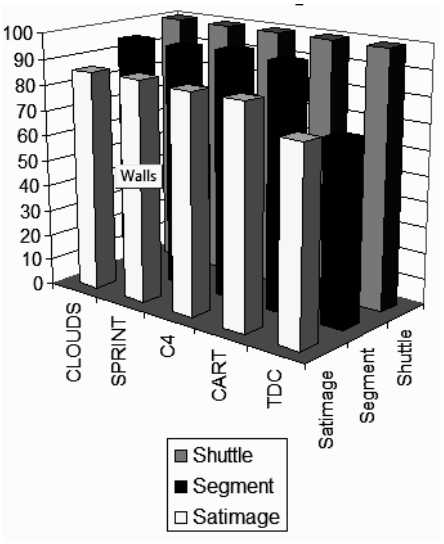


Fig. 6. Testing accuracy of TDC and other approaches.

- Learning time. This parameter is the time which is taken for learning and constructing the decision trees. Different approaches try to shorten the time.

There is a trade off between these parameters. Hence all existing approaches try to establish equilibrium. In this study the equilibrium was established such that it would make this approach efficient enough to be used particularly in cases of massive data sets, memory restrictions or short learning time.

Fig. 6 shows the accuracy of different approaches. This is obvious that this approach is not absolutely accurate but it has an acceptable accuracy.

Fig. 7 shows the tree size of different approaches. This is clear that our approach has a very small size tree that makes it usable for the classification of massive data sets.

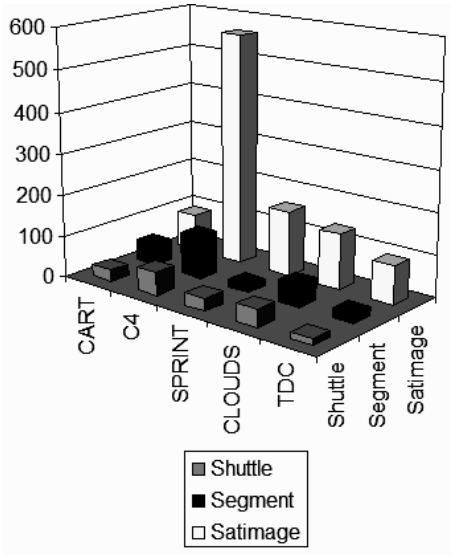


Fig. 7. Tree size of TDC and other approaches.

The number of operations that should be done for decision tree learning is a function of $n^2 * m^3$ in TDC, which *n* and *m* stand for the number of attributes and data samples alternatively. Therefore this algorithm has polynomial time complexity.

The more the size of the decision tree grows, the more the number of operations, which has to be done for classification, increases. Therefore, the tree simplicity of this approach leads to reduction of the classification time.

The main amount of memory used, except the one which is used for storing training and test data sets, is an array with the size of $m * k * 3$ which k stands for the classes.

VI. Conclusion and Future Work

Considering the experimental results, shown in Fig. 6, the accuracy of the constructed decision trees are not better than the previous ones. It is worth to say that the aim of this study is not to improve the accuracy, but to make simpler decision trees which have fewer branches, use short learning time and a small amount of memory. Optimization of the parameters which are used in decision tree construction algorithm can help us to increase the accuracy of this approach. We explained how to use genetic algorithm to optimize the parameters and therefore increase the accuracy of the trees.

Fig. 7 shows that the trees constructed by using this algorithm, are simpler than others. This may result in a shorter classification time, because a smaller tree reduces the number of comparisons that have to be done for classification. This reduction is useful in case of massive data and makes it possible to classify large data sets. As well therefore, we propose this approach for the classification of massive data sets.

As mentioned above, this algorithm is of polynomial time complexity classes. Considering this characteristic we recommend this algorithm in cases in which we have short learning time.

Because of the small amount of memory usage, we can use this approach for the time we have memory restrictions.
For the future work, the classification accuracy needs to be improved. This is clear that if the training data set is not uniformly distributed, classic approaches may construct inaccurate trees. We propose a solution to this problem which can be used in TDC. Instead of using the abundance array which contains the class patterns we could use the differential of it. This may lead to ignoring local dispersion in the data set and making the parts of the data which have less congestion useful in extending the tree. Negative effects though it might have on the other parameters, the accuracy is augmented.

References

[1] M. Ankerst, C. Elsen, M. Ester, and H.-P. Kriegel, "Visual classification: An interactive approach to decision tree construction,"

Proc. 5th Intl. Conf. on Knowledge Discovery and Data Mining (KDD '99), pp. 392–396, 1999.

[2] M. Ankerst, M. Ester, and H.-P. Kriegel, "Towards an effective cooperation of the user and the computer for classification," Proc. 6th Intl. Conf. on Knowledge Discovery and Data Mining (KDD '00), 2000.

[3] Qiang Ding, Qin Ding, William Perrizo, "Decision tree classification of spatial data streams using Peano Count Trees," SAC, 413-417, 2002.

[4] João Gama, Pedro Medas, Pedro Pereira Rodrigues, "Learning decision trees from dynamic data streams," SAC, 573-577, 2005.

[5] J. Gehrke, V Ganti, R. Ramakrishnan, and W.-Y. Loh, "BOAT - Optimistic decision tree construction," In A. Delis, C. Faloutsos, and S. Ghandeharizadeh, editors, Proc. ACM SIGMOD'99, Philadephia, USA, ACM, pages 169-l 80, 1999.

[6] J. Gehrke, R. Ramakrishnan, and V. Ganti, "RainForest – A Framework for fast decision tree construction of large datasets," In A. Gupta, 0. Shmueli, and J. Widom, editors, Proc. VLDB '98, New York, USA, pp. 416-427, 1998.

[7] Nicholas R. Howe, Toni M. Rath, and R. Manmatha, "Boosted decision trees for word recognition in handwritten document retrieval," SIGIR, PP. 377-383, 2005.

[8] Patrick Knab, Martin Pinzger, and Abraham Bernstein, "Predicting defect densities in source code files with decision tree learners," MSR, pp. 119-125, 2006.

[9] D. Michie, D.J. Spiegelhalter and C.C. Taylor, "Machine learning, neural and statistical classification," Ellis Horwood, 1994.

[10] Kai-Uwe Sattler, Oliver Dunemann, "SQL database primitives for decision tree classifiers," CIKM, pp. 379-386, 2001.

[11] Soon Tee Teoh, Kwan-Liu Ma, "PaintingClass: interactive construction, visualization and exploration of decision trees," KDD, pp. 667-672, 2003.

[12] Md. Zahidul Islam, Ljiljana Brankovic: A framework for privacy preserving classification in data mining. ACSW Frontiers: 163-168, 2004.

[13] Reeves, Colin R., Rowe, Jonathan E: Genetic algorithms-Principles and Perspectives: A guid to GA theory. Springer, ISBN: 978-1-4020-7240-6, 2002.

[14] Melanie Mitchell: An Introduction to Genetic Algorithms, MIT press, 1998.