Improved algorithm of decision tree based on neural network

Min Zhang, Hongwei Peng, Xiaoling Yan
College of Information Engineering, Dalian University, Dalian, 116000, China
2306118105@qq.com
azhangmin@dlu.edu.cn

Abstract. Decision trees have been applied to solve many data mining problems due to their superior learning and classification capabilities, and have achieved good results. However, for dealing with big data and complex model problems, decision trees show insufficient accuracy and overfitting. In order to solve these problems, neural network is introduced as a decision tree node, and an improved algorithm based on neural network decision tree is proposed. In the neural network decision tree, the classifier learning consists of two stages: the first stage uses a heuristic algorithm with reduced uncertainty to divide the big data, and stops the growth of the decision tree until the node dividing ability is below a certain threshold; in the second stage, the neural network is used to classify the decision-making leaf node with generalization ability. The experimental results show that compared with the traditional classification learning algorithm, the algorithm has a higher accuracy rate and it can determine the size of decision tree through structural adaptation for the classification problem of identifying big data and complex patterns.

1. Introduction
With the development of science and technology, learning from big data has become inevitable in many fields, such as machine learning, pattern recognition, medical diagnosis, and image processing. Decision tree learning has the advantages of simple implementation, few parameters, low calculation and can flexibly handle various big data. For decision trees, the scale of the tree reflects the generalization ability to a certain extent. The larger the scale of the tree, the more complex the rules extracted from the tree. If the rules are too complex, it will lead to overfitting problems [1]. Without affecting the classification accuracy, it is very important to make the optimized decision tree as small as possible. Neural network has been proven to be an effective learning method for performing classification tasks, especially when high-dimensional data is input and the relationship between input and output is complex, neural networks shows good performance [2]. Studies have shown that the representation ability of neural network models will increase exponentially with the increase of depth, thereby improving classification ability or prediction accuracy. However, this process will consume a lot of training times.

In recent years, scholars have proposed many ensemble learning algorithms about neural networks and decision trees. Reference [3] proposed to preprocess the relationship between each attribute and the target attribute through a neural network, and then establish a derivative relationship between each attribute and the classification result to establish a tree, but the algorithm time complexity is high. Reference [4] proposed a hybrid learning model of BP algorithm based on C4.5 algorithm and optimization to solve the problem of difficult selection of BP neural network input parameters and hidden layer nodes, it can not solve the multi-classification problem because it is a binary tree.
Reference [5] proposed an extreme learning machine tree (PEP-ELM-Tree) model, however the algorithm uses information gain in node splitting, which tends to be biased towards the attributes of selecting more branches and leads to overfitting.

To solve these problems, this paper proposes an improved algorithm based on neural network decision tree (ADT). For a sample set, using decision tree to "roughly" divide dataset when the division reaches a certain level, using neural network to replace the output category of the decision tree. ADT can adaptively determine the size of the decision tree through the structure of the classification problem of big data and complex patterns.

2. Related work

2.1. Decision tree

Decision tree is an inductive learning method. It adopts a top-down recursive method. Starting from the root node, selecting attributes on each node according to a given measurement method and build branches down according to the possible values of the corresponding attributes. Until the training set has no remaining attributes to further divide the samples [6]. A path from root to leaf corresponds to a conjunctive rule, and the whole decision tree corresponds to a set of disjunctive expression rules. Common measures include information gain, information gain rate, Gini index, chi-square test, etc. The specific steps of the decision tree algorithm are as follows [7].

(1) Create the root.
(2) If the training sets belongs to a same class, return it as a leaf node and mark it with this class.
(3) If the training sets is empty or there are no remaining attributes to further divide the sample, return it as a leaf node.
(4) According to a certain measurement method, select the best splitting attribute and establish a branch.
(5) For remaining attributes, call (2)-(4) recursively, selecting the best split attribute, creating sub nodes on the training sets, and further divide the subset.
(6) Pruning.

2.2. Neural Network

Neural network is a computing model, which is composed of a large number of interconnected neurons. Neural network structure includes input layer, hidden layer, and output layer. The input layer is responsible for receiving external data; the hidden layer is responsible for processing information and constantly adjusting the connection attributes between neurons; the output layer is responsible for outputting the calculation results. In the learning stage, by adjusting the weight of the neural network, the error between the predicted sample and the actual sample is gradually reduced to achieve the best fitting result [8]. The feedforward neural network training process is as follows.

(1) Input the training set, the maximum number of iterations T, and initialize $W, b$.
(2) Calculate the input and output of each neuron in the hidden layer.
(3) Use the expected output and actual output of the network to calculate the loss function.
(4) Calculate the partial derivative of each layer parameter and update the parameters.
(5) Determine whether the network error meets the requirements. When the error reaches the preset accuracy or the number of learning times is greater than the set maximum iterations, the algorithm ends. Otherwise, the next learning sample is selected and returns to step (2).

3. Decision tree algorithm based on Neural Network

The ADT classifier learning is divided into two steps: the first step is to select the best attribute from the training set as the split point and build a tree until the division ability is lower than the given uncertainty coefficient to stop the growth of the decision tree; the second step classify the tree with generalization ability by using neural network.
3.1. Conditional category entropy

Supposing that $\alpha = \{\alpha_1, \alpha_2, \cdots \alpha_n\}$ represents a certain division of training set $D$, then the information entropy of $D$ is:

$$H(D, \alpha) = -\sum_{i=1}^{m} p(\alpha_i, D) \log_2(p(\alpha_i, D))$$

(1)

If $\beta = \{\beta_1, \beta_2, \cdots \beta_n\}$ is another division of training set $D$, then the information entropy is:

$$H(D, \beta) = -\sum_{i=1}^{m} p(\beta_i, D) \log_2(p(\beta_i, D))$$

(2)

Let $\alpha$ and $\beta$ division $D$ then

$$H(D, \alpha \cap \beta) = -\sum_{i=1}^{m} \sum_{j=1}^{m} p(\alpha_i \cap \beta_j, D) \log_2(p(\alpha_i \cap \beta_j, D))$$

(3)

The partition conditional entropy has the following connection.

$$H(D, \beta / \alpha) = H(D, \alpha \cap \beta) - H(D, \alpha)$$

(4)

Define two division distances as:

$$d(\alpha, \beta, D) = H(D, \beta / \alpha) + H(D, \alpha / \beta)$$

(5)

Using normed distance as the attribute measurement method [9], then

$$d_\alpha(\alpha, \beta) = \frac{d(\alpha, \beta, D)}{H(\alpha \cap \beta)} = 2 - \frac{H(\alpha) + H(\beta)}{H(\alpha \cap \beta)}$$

(6)

where $H(\alpha \cap \beta) = H(D, \alpha \cap \beta)$, $0 \leq d_\alpha(\alpha, \beta) \leq 1$

If decision tree completely separates the input samples, that is, the samples mapped to each leaf node belong to the same class, then the entropy of each leaf node is equal to 0, and formula (1) is 0. But for complex problems, it is almost impossible to completely separate training sets (assuming that they can be completely separated, the decision tree is easy to overfit), conditional entropy can be used as a measure of attribute node partition in decision tree.

3.2. Uncertainty coefficient $\varepsilon$

To avoid decision tree getting stuck during training process, we introduced a penalty factor to encourage each internal node to use both left and right subtrees equally. Without this penalty, trees tended to get stuck on plateaus where one or more internal nodes always assigned almost all probabilities to one of its subtrees and the logical gradient of the decision was always very close to zero. The penalty factor of node $i$ is calculated as follows.

$$\alpha_i = \frac{\sum p'(x)p(x)}{\sum p'(x)}$$

(7)

where $p'(x)$ is the path probability from root to node $i$. Then the calculation formula of the node uncertainty coefficient $\varepsilon$ is as follows.

$$\varepsilon = \lambda \sum_{\text{inner nodes}} \left[ \frac{1}{2} \log_2(\alpha_i) + \frac{1}{2} \log_2(1-\alpha_i) \right]$$

(8)

where $\lambda$ is the penalty intensity parameter, which is set before training. The penalty is based on the assumption that use of alternative neural networks will generally be more suitable for any given classification task, and does improve accuracy in practice. The penalty intensity decays exponentially with the depth of the tree node.

3.3. ADT construction algorithm

ADT model is a tree structure, in the root and middle, according to decision tree algorithm; in the leaf nodes, select appropriate features and embed the neural network to implement local decision-making on sample subsets. The specific construction process is as follows.

1. Data preprocessing.
2. Calculate the equivalence class division of each condition attribute, $R = \{\alpha_1, \alpha_2, \cdots \alpha_n\}$ is a set
of conditional attributes.

(3) Decision tree T is represented as four tuple \([U, A, V, f]\), where \(U\) stands for data sets, \(A = C \cup D\), \(C\) and \(D\) are condition attributes and decision attributes, respectively. Calculated the \(K\) value of each attribute, and stored the two conditional attributes with the largest \(K\) value into \(SIG\), where \(D = \{d\}\) represented the decision attribute set.

\[
K(C, D) = \max \text{card} \left\{ X_i : X_i \subseteq Y, \forall j \right\} / \text{card}(U)
\]

(4) If \((\text{card}(C_i) = 1)\) then \(a_f = C_i\) and to step(5);
else if \((\text{card}(C_i) = 2)\) then \(a_f \cup a_s = C_i\) and to step(6);
else calculated \(I(a, D)\) where \(a_i \in C_i\) choose \(a_f, a_s \in C_i\) made \(\min \{|I(a_f, D) - I(a_s, D)|\}

(5) Calculated \(d_y(a_f, a_s), a_i \in C_2\), find \(N = \{a_i : \min d_y(a_f, a_s), a_i \in C_2\}\), if the base of \(N\) is 1, then \(a_s = N\), otherwise calculated \(H(a_i \mid a_f)\), choose \(a_x\) made \(\min H(a_x \mid a_f)\), \(a_s = a_N\)

(6) \(P = a_f \land a_s\) calculated the generalization of \(P\) relative to \(D\), \(GEN_0(P), U = Z_m/1, R = R - P\), Check it as the node and return to step (2).

(7) Calculate the node penalty factor and uncertainty coefficient \(\varepsilon\) according to formulas (7) and (8). When the conditional entropy value of the node is less than \(\varepsilon\), stop the growth of the decision tree.

(8) At the leaf nodes, neural network is trained through back propagation, and the parameters are adjusted by the gradient descent method. The cross entropy loss function is calculated as follows.

\[
\text{loss}(x) = -\frac{1}{n} \sum \left[ y \ln(p) + (1 - y) \ln(1 - p) \right]
\]

where \(x\) represents the sample, \(n\) represents the total number of samples, and \(p\) represents the probability that sample \(i\) is predicted to be positive.

(9) The weight and bias are calculated as follows.

\[
\frac{\partial C}{\partial w} = \frac{1}{n} \sum \sigma(z) - y) \quad (10)
\]

\[
\frac{\partial C}{\partial b} = \frac{1}{n} \sum \sigma(z) - y) \quad (11)
\]

3.4. ADT loss function

Given a training set \((x_i, y_i)\), where \(x_i\) is the sample of training, \(y_i\) is the expected corresponding to the sample and ADT prediction model is \(f(x)\). The purpose of training is to make the expected value equal to the actual value as much as possible, if \(f(x) \neq y\), represents the prediction deviation, and a function is needed to define the loss caused by the deviation. Let \(h(x)\) is the ADT loss function.

\[
h(x) = \int \left[ \sum \text{loss}(x) \right]
\]

Where \(\text{loss}(x) = -\frac{1}{n} \sum \left[ y \ln(p) + (1 - y) \ln(1 - p) \right]\)

4. Experiment

To verify Decision tree based on neural network algorithm proposed in this paper, 15 sets of experiments were designed, 1-14 from UCI public data set and 15 from kaggle data set. The details of each dataset are shown in Table 1. To measure the testing performance, we randomly take 20 percent of data from the complete dataset as the testing datasets. To reflect the performance of the model more objectively, multiple times hold-out methods are used to conduct experiments, and the average of the results of 10 times was taken as the final experimental result, the best result is shown in boldface in the table.
Table 1 Data set description

| No | Dataset                              | #Instance | #Attribute | #Class |
|----|--------------------------------------|-----------|------------|--------|
| 1  | MAGIC Gamma Telescope                | 19020     | 11         | 2      |
| 2  | Pima Indian Diabetes                 | 768       | 8          | 2      |
| 3  | Waveform Noise                       | 5000      | 40         | 3      |
| 4  | Yeast                                | 1484      | 8          | 10     |
| 5  | Credit Approval                      | 690       | 15         | 2      |
| 6  | Heart Disease Data Set               | 303       | 14         | 2      |
| 7  | Autism Screening Adult               | 704       | 21         | 2      |
| 8  | OBS-Network                          | 1075      | 22         | 4      |
| 9  | Banknote Authentication              | 1372      | 5          | 2      |
| 10 | Wireless Indoor Localization         | 2000      | 7          | 4      |
| 11 | HTRU2                                | 17898     | 9          | 2      |
| 12 | Healthy Older People                 | 75128     | 9          | 4      |
| 13 | Abalone                              | 4177      | 8          | 3      |
| 14 | Wine                                 | 178       | 13         | 3      |
| 15 | SARS B-cell Epitope Prediction       | 14387     | 13         | 2      |

For decision trees, the scale of tree reflects the generalization ability of the tree to a certain extent. The larger the scale of tree, the higher accuracy, and the more complex rules extracted from the tree. If the rules are too complex, which will lead to overfitting problem. The purpose of ADT algorithm is to alleviate the problem of decision tree overfitting and long training time of neural network, and improve the generalization ability of the tree.

4.1. Training time comparisons of ADT

Table 2 shows training time of PEP-ELM, ANN, and ADT on 15 data sets. It can be seen from the table that neural network is the most time-consuming. The possible reason is that neural network has many hidden layers and a large number of parameters, which leads to a long training time. ADT takes the shortest time in all of the 15 data sets. Compared with neural network, the accuracy of decision tree based on neural network is improved by 2%-5%, and training time is shortened by 5-10 times.

| No | PEP-ELM  | ANN    | ADT    |
|----|----------|--------|--------|
| 1  | 185.67   | 670.83 | 90.84  |
| 2  | 3.76     | 10.739 | 2.76   |
| 3  | 281.72   | 941.72 | 163.51 |
| 4  | 8.98     | 68.51  | 6.03   |
| 5  | 5.92     | 11.93  | 3.76   |
| 6  | 4.47     | 8.03   | 3.56   |
| 7  | 1.13     | 1.94   | 0.96   |
| 8  | 4.91     | 7.97   | 3.96   |
| 9  | 4.19     | 5.87   | 2.88   |
| 10 | 10.98    | 17.86  | 4.59   |
| 11 | 310.03   | 917.91 | 169.97 |
| 12 | 973.24   | 2476.8 | 256.97 |
| 13 | 31.11    | 109.62 | 21.65  |
| 14 | 0.84     | 1.23   | 0.53   |
| 15 | 226.81   | 742.69 | 100.84 |

4.2. Testing accuracy comparisons of ADT

Table 3 shows the testing accuracies of C4.5, PEP-ELM, ANN and ADT algorithms on 15 data sets. In the 15 data sets, ADT performed well in 13 and the accuracy difference in the remaining 2 was less than 1% and in the big data sets 1, 11, 12, 15, ADT accuracy rate increased by 3%-5%. Combined with Table 2 and
Table 3, ADT reduces the training time while improving the accuracy, and to a certain extent avoids the decision tree overfitting.

**Table 3 Testing accuracies of C4.5, PEP-ELM, ANN, ADT**

| No | C4.5  | PEP-ELM | ANN  | ADT  |
|----|-------|---------|------|------|
| 1  | 0.814 | 0.826   | 0.805| 0.847|
| 2  | 0.706 | 0.745   | 0.739| 0.769|
| 3  | 0.708 | 0.752   | 0.724| 0.751|
| 4  | 0.518 | 0.571   | 0.552| 0.603|
| 5  | 0.831 | 0.834   | 0.876| 0.876|
| 6  | 0.780 | 0.547   | 0.803| 0.856|
| 7  | 0.923 | 0.931   | 0.942| 0.965|
| 8  | 0.959 | 0.973   | **0.980** | 0.973 |
| 9  | 0.873 | 0.843   | 0.875| 0.890|
| 10 | 0.864 | 0.864   | 0.868| 0.889|
| 11 | 0.932 | 0.941   | 0.901| 0.977|
| 12 | 0.973 | 0.974   | 0.977| 0.977|
| 13 | 0.512 | 0.513   | 0.562| 0.653|
| 14 | 0.890 | 0.908   | 0.823| 0.915|
| 15 | 0.698 | 0.811   | 0.698| 0.842|

(0/15) (1/15) (1/15) (13/15)

**4.3. Recall rate comparisons of ADT**

Table 4 shows the recall rates of C4.5, PEP-ELM, ANN, and ADT on 15 data sets. Recall rate is a measure of coverage, that is, how many positive samples are correctly classified as positive samples. As can be seen from the table, ADT performed well in 12 of the 15 data sets. In the remaining three, the gap with other algorithms is less than 1%.

**Table 4 Recall rate of C4.5, PEP-ELM, ANN, ADT**

| No | C4.5  | PEP-ELM | ANN  | ADT  |
|----|-------|---------|------|------|
| 1  | 0.79  | 0.83    | 0.83 | 0.84 |
| 2  | 0.71  | 0.74    | 0.74 | 0.77 |
| 3  | 0.68  | **0.77** | 0.74 | 0.76 |
| 4  | 0.50  | 0.52    | 0.54 | 0.57 |
| 5  | 0.82  | 0.85    | 0.88 | 0.88 |
| 6  | 0.80  | 0.83    | 0.83 | 0.85 |
| 7  | 0.94  | 0.92    | 0.93 | **0.98** |
| 8  | 0.74  | 0.94    | **0.98** | 0.97 |
| 9  | 0.89  | 0.94    | 0.92 | **0.96** |
| 10 | 0.92  | 0.89    | 0.94 | **0.95** |
| 11 | 0.87  | 0.87    | 0.89 | 0.93 |
| 12 | 0.89  | 0.97    | 0.97 | **0.98** |
| 13 | 0.56  | 0.53    | 0.63 | 0.72 |
| 14 | 0.87  | **0.95** | 0.88 | 0.94 |
| 15 | 0.59  | 0.53    | 0.78 | **0.82** |

(0/15) (2/15) (1/15) (12/15)

**4.4. F1 values comparisons of ADT**

Table 5 shows the F1 values of C4.5, PEP-ELM, ANN, and ADT on the 15 data sets. F-measure is the weighted harmonic average of accuracy and recall, when the parameter \( a = 1 \), is the most common \( F_1 = \frac{2 \cdot P \cdot R}{P + R} \), Where P represents the accuracy, R represents the recall rate \(^{[10]}\). As can be seen from the table, ADT performed well in 13 of the 15 data sets. In the remaining two, the gap with other
algorithms is less than 1%, indicating that the algorithm proposed in this paper is more effective.

| No | C4.5 | PEP-ELM | ANN | ADT |
|----|------|---------|-----|-----|
| 1  | 0.77 | 0.81    | 0.84| 0.86|
| 2  | 0.69 | 0.74    | 0.73| 0.76|
| 3  | 0.67 | 0.75    | 0.76| 0.74|
| 4  | 0.48 | 0.56    | 0.53| 0.58|
| 5  | 0.83 | 0.85    | 0.84| 0.87|
| 6  | 0.75 | 0.83    | 0.82| 0.86|
| 7  | 0.94 | 0.93    | 0.97| 0.98|
| 8  | 0.74 | 0.83    | 0.97| 0.97|
| 9  | 0.90 | 0.91    | 0.93| 0.94|
| 10 | 0.85 | 0.86    | 0.86| 0.89|
| 11 | 0.91 | 0.95    | 0.93| 0.98|
| 12 | 0.85 | 0.94    | 0.90| 0.97|
| 13 | 0.85 | 0.81    | 0.89| 0.92|
| 14 | 0.87 | 0.90    | 0.87| 0.92|
| 15 | 0.71 | 0.46    | 0.62| 0.79|

5. Conclusion
This paper proposes an improved algorithm of decision tree based on neural network to solve the problem of DT induction overfitting and the time-consuming problem of neural network. Firstly, used decision tree to divide the big data, when the node division ability is less than the given uncertainty coefficient, neural network is embedded as its leaf nodes. The fusion algorithm combines the advantages of decision trees and neural networks, with high accuracy, short training time and performs well in big data classification. Future work on this topic may include the following two aspects: further research on the incremental mechanism of ADT and induction of ADT with mixed type attributes.

Acknowledgments
This work was financially supported by the key research and development plan of Liaoning Province "Research on Big Data Processing Method of Medical Care Based on Intelligent Perception" (2017104014); "Research on Heart Disease Risk Assessment Model Based on Big Data Analysis", a general project of the Natural Science Foundation of Liaoning Province (20180551247).

References
[1] Junhai Z, Xizhao W, Sufang Z, et al. Tolerance rough fuzzy decision tree[J]. Information Sciences, 2018, 465(69):425-438.
[2] Kolsbjerg E L, Peterson A A, Hammer B. Neural-network-enhanced evolutionary algorithm applied to supported metal nanoparticles[J]. Physical Review B, 2018, 97(19):195-201.
[3] Zhu F. A Classification Algorithm of CART Decision Tree based on MapReduce Attribute Weights[J]. International Journal of Performability Engineering, 2018, 14(75):1-55.
[4] Orshea J D, Crockett K A, Khan W, et al. A hybrid model combining neural networks and decision tree for comprehension detection[C] Royal society open science,2018,43(09):1-7.
[5] Ran Wang, Yu-Lin He, Chi-Yin Chow, Fang-Fang Ou, Jian Zhang. Learning ELM-Tree from big data based on uncertainty reduction[J]. Fuzzy Sets and Systems,2015,258(33):270-274.
[6] Reimers N, Dehghani N. Event Time Extraction with a Decision Tree of Neural Classifiers[J]. Transactions of the Association for Computational Linguistics, 2018, 6(95):77-89.
[7] Diez, Francisco Javier, Luque M, Berme. Decision analysis networks[J]. International Journal
of Approximate Reasoning, 2018, 96(2):83-90.

[8] Wu Zheng, An Hong, et al. Research and Optimization of Fast Convolution Algorithm Winograd on Intel Platform[J]. Computer research and development, 2019, 56(004):825-835.

[9] Bo X U, Xianyong Z, Shan F. Weighted Dependence of Neighborhood Rough Sets and Its Heuristic Reduction Algorithm[J]. Pattern Recognition and Artificial Intelligence, 2018,14(54):132-140.

[10] Kiss G, Naderpour M, Liu J, et al. SoK: Modular and Efficient Private Decision Tree Evaluation[J]. Proceedings on Privacy Enhancing Technologies, 2019,74(2):187-208.