A VFDT algorithm optimization and application thereof in data stream classification

Shuangying Jia
Xijing University, Xi'an, Shaanxi, China, 710123

Corresponding author and e-mail: Shuangying Jia, 925297483@qq.com

Abstract. Data stream mining is a new technology for dynamically extracting feature patterns from data streams. Its core technology is data stream mining algorithms. Data stream mining algorithms are divided into clustering and classification algorithms. In data stream classification algorithms, VFDT (The Concept-adapting Very Fast Decision Tree algorithm is an effective classification decision tree algorithm. The algorithm dynamically constructs a decision tree based on the improvement of the Hoeffding tree. With the inflow of data, new branches are continuously added and outdated Branch. However, the algorithm has the problem of concept drift, and some nodes may no longer meet the Hoeffding bound, which affects the classification mining effect. This paper proposes an optimization algorithm (VFDT), which reduces the impact of concept drift and network noise by adding sliding window technology and fuzzy technology to the VFDT classification mining algorithm. Experimental results show that the algorithm can effectively improve the accuracy of stream data classification mining. Reduce classification errors.

1. Introduction
Classification or prediction is one of the basic problems of data mining. There are many mature algorithms for the classification and mining of traditional data sets. Data flow classification and frequent pattern mining cannot use static traditional mining algorithms. Since class definitions change with time, it is necessary to consider the characteristics of data flow changes and delete outdated class definition patterns in time.

The VFDT algorithm is based on the improvement of the Hoeffding tree to dynamically construct the decision tree. The Hoeffding boundary is used to ensure that the data used to construct each subtree has sufficient information, and as data flows in, new branches are continuously added and obsolete branches are cut out. However, the algorithm has a concept drift problem, and some nodes may no longer satisfy the Hoeffding boundary.

2. Performance analysis of VFDT algorithm
2.1. Related definitions and theorems
Definition 1. Definition of data set: Let data division $D$ be the tuple training set of class labels. Assuming that the class label attribute has $m$ different classes $C_i (i=1,...,m)$, let $C_{i,D}$ be the set of tuples of class $D$ in $C_i$, and $|D|$ and $|C_{i,D}|$ are the number of tuples in $D$ and $C_{i,D}$ respectively.

Definition 2. Data flow: Let $t$ denote any timestamp and $x_t$ denote the data vector arriving at time $t$, then the data flow can be expressed as $\{..., x_{t-1}, x_t, x_{t+1}, ...\}$.

Definition 3. Information gain: The average amount of information required to classify the tuples in $D$ is given by:
\[ \text{Info}(D) = - \sum_{i=1}^{m} p_i \log_2 (p_i) \] \hspace{2cm} (1)

Where \( p_i \) is the probability that any tuple in \( D \) belongs to class \( C_i \) and is estimated using \( |C_i|/|D| \).

Suppose that tuples in \( D \) are divided by attribute \( A \), where \( A \) has \( v \) different values \( \{a_1, a_2, ..., a_v\} \) according to the observations of the training data. \( v \) different values corresponding to attribute \( A \) may divide \( D \) into \( v \) subsets \( \{D_1, D_2, ..., D_v\} \), where \( D_j \) represents a tuple with an \( a_j \) value on \( A \).

The desired information based on the classification of tuples in \( D \) divided by \( A \) is given by:

\[ \text{Info}_A(D) = \sum_{j=1}^{v} \left| \frac{D_j}{D} \right| \times \text{Info}(D_j) \] \hspace{2cm} (2)

Then the information gain divided by attributes is:

\[ \text{Gain}(A) = \text{Info}(D) - \text{Info}_A(D) \] \hspace{2cm} (3)

Theorem 1: Let \( \{X_i\}_{i=1}^{m} \) be \( m \) random variables, \( 0 \leq X_i \leq r \), let \( \overline{X} = \frac{1}{m} \sum_{i} X_i \), \( \overline{X} \)'s mathematical expectation be \( \mu \). For a given \( \varepsilon > 0 \), there are:

\[ \Pr(|\overline{X} - \mu| \geq \varepsilon) \leq 2^{-2m\varepsilon^2/r^2} \] \hspace{2cm} (4)

2.2. Principle of VFDT algorithm

G. Hulten et al. Made some modifications to the Hoefdding tree algorithm and proposed the VFDT (Very Fast Decision Tree) algorithm [5] to improve speed and memory utilization. These modifications include more proactively breaking near draws during attribute selection, calculating information gain after training many samples, removing the least promising leaf node's active state once memory is insufficient, removing poorly split attributes, and improving initialization methods.

VFDT works well with convection data, and is much better than traditional classification methods in terms of speed and accuracy. However, it still cannot handle the concept drift problem in the data stream.

Both the Hoefdding tree and the VFDT algorithm are designed for discrete attributes, and do not consider the selection of continuous attribute split points. The phenomenon that the sample data has multiple continuous attribute fields is widespread in streaming data. This point is connected to the network. This is particularly evident in the data flow.

2.3. Analysis of VFDT algorithm

The goal of the decision tree algorithm is to derive a classification model from the existing training samples and correctly classify the new test samples. VFDT is a method based on Hoeifdding inequality to establish a classification decision tree for data stream mining environment. It is generated by continuously replacing leaf nodes with branch nodes, and the sample attributes studied are discrete attributes.

The execution process of the algorithm:

1) The construction of the fast decision tree VFDT starts from the root node, which is the initial leaf node. If \( S \) is a data stream sequence, it contains potentially infinite number of sample data. The error parameter \( \delta \) is initially given by the user. The different attribute fields of the sample are represented by the attribute set \( \{X_1, X_2, ..., X_k\} \), and \( k \) indicates the number of attributes.

2) When the sample data flows into the VFDT system in sequence, initially all the sample data are gathered at the root node of the decision tree. As the sample data of the root node increases, \( n_t \) is used to indicate the total number of samples flowing in from 0 to \( t \).

3) Use the information gain as the attribute selection metric. When the number of samples gathered at the root node at time is \( n_t \), calculate the information gain of each attribute according to equation (1.3). If the average information gain \( \text{Gain}(X_i) \) of the attribute \( X_i \) is the largest, the average information gain of the attribute \( X_i \) is the next, let:
\[
\Delta \text{Gain} = \text{Gain}(X_a) - \text{Gain}(X_b)
\]  

(5)

(4) If Eq. (1.5) is satisfied, the root node will grow a child node according to the best split attribute \( X_a \), and delete the attribute \( X_a \) in the candidate attribute set in its child node, this process is performed recursively. Due to the potential unlimited nature of the data stream, the decision tree will grow unlimitedly without restrictions. If the maximum depth of the tree or other metrics are determined, VFDT will incrementally update the decision tree with the latest sample data. To maintain the accuracy of its judgment.

(5) Based on the Hoeffding tree algorithm, VFDT makes improvements in memory optimization. When the current data fills the memory space, the VFDT system will temporarily remove the space used by the child nodes that have the least impact on the classification decision. For a child node that has temporarily lost its activity, if its classification accuracy is higher than the current active node later, it will resume its activity again.

(6) When the difference between the average information gain of the best split attribute and the next best split attribute is small, the traditional Hoeffding tree algorithm will spend a lot of time on attribute selection. The VFDT algorithm introduces a limit parameter \( \tau \) (provided by the user). If \( \Delta \text{Gain} \leq \epsilon < \tau \), the attribute with the largest gain is selected as the split attribute.

(7) Optimization of processing speed. In step (3), the traditional Hoeffding tree algorithm will perform a split attribute test when each sample arrives, which will greatly affect the calculation efficiency of the system. The VFDT system introduces a minimum sample number \( n_{\min} \) of splitting attributes. The test is performed when the number of samples reaching the node is an integer multiple of \( n_{\min} \). Because the VFDT algorithm does not consider the concept drift problem. Therefore, when the VFDT algorithm is used to directly classify the network data stream with widespread concept drift, a large deviation will occur. In addition, as time goes by, concept drift occurs, and a large number of outdated examples will accumulate in the VFDT tree, making the VFDT tree very bloated.

3. Optimization method of VFDT algorithm

For the concept drift problem, refer to the CVFDT algorithm. During the growth of the original decision tree, if any nodes are classified incorrectly, an alternative subtree \( T_{alt} \) is derived next to the corresponding node. When the replacement subtree grows enough to accurately classify the newly arrived sample, the corresponding subtree in the original tree will be replaced.

The execution process of the optimized algorithm:

(1) The core decision tree of the optimized algorithm is still the Hoeffding tree mentioned by the VFDT algorithm. The reason is that the Hoeffding decision tree can rely on the principle of the Hoeffding boundary, replace infinite samples with small samples to construct an efficient incremental decision tree, and only need to scan the data stream once, which meets our application requirements.

(2) Introduce a response mechanism for newly arrived samples when concept drift occurs. In the algorithm, a sliding window \( W \) is added, and when a new sample arrives, it is added to the sliding window. The task of the sliding window is to respond to the characteristics of the new sample by increasing the count in the corresponding node of the decision tree when a new sample arrives, while reducing the corresponding count of the old sample or the outdated sample to maintain an up-to-date classification model, rather than one When a new sample arrives, a new model is created.

(3) The optimized algorithm improves the sliding window and introduces an impact factor \( \beta \in [0,1] \) for each data stream sample. The value of \( \beta \) is used to judge the classification utility of the decision tree, and affect the sample count value in the sliding window according to the different values, and then dynamically adjust the size of the sliding window. In the improved sliding window, our count of flowing samples is based on the count \( n_{\beta} \) of the impact factor. When there is a deviation in the classification of a sample in the decision tree, the degree to which the influence factor \( 0 \leq \beta < 1 \) of the sample in the sliding window tends to 0 is proportional to the distance between the deviation node and the root node (root). You can use the tree depth \( l \) (The number of layers with deviations), that is \( \beta \propto l \). On the contrary, when the actual classification is the correct classification, \( K \). We set a threshold \( \epsilon \) (such as 0.5) in the sliding window, let \( |W| \) be the sample count in the sliding window, \( |W| \in [0, n_{\beta}] \) and specify: when
When \(|w| = n_{ijk} \beta \leq \zeta\) \quad (6)

When satisfied, it means that there is an obvious concept drift, lock the wrong node, construct an alternative subtree and reduce the size of the sliding window until the following formula holds;

\[|w| > \zeta\] \quad (7)

When \(|w| = n_{ijk} \beta > \zeta\) \quad (8)

When it is established and remains stable for at least \(T\) (artificially defined) window periods, the size of the sliding window is recursively expanded in increments of \(\frac{1}{2}|W|\) until it reaches the critical point of (4.7).

(4) The optimized algorithm will judge the effectiveness of the split based on the number of effective samples in the sliding window, that is: if the formula (2.1) is satisfied, the fuzzy information of each attribute in the node with severe splitting deviation in the sliding window will be recalculated Gain. If a certain internal node is categorized downwards and the influence factor \(\beta\) of the corresponding sample point in the sliding window is larger than the previous point, the concept point tends to 0, indicating that a conceptual drift has occurred in classifying the sample point. This shows that there is a deviation in the attribute test of the node, and a substitute subtree will be generated at this time. If the fuzzy information gain of a new sample attribute \(X_{\text{new}}\) is higher than the current split attribute \(X_{\text{curr}}\), the optimized algorithm will generate an alternative subtree with the attribute \(X_{\text{new}}\) as the root node at the corresponding node.

(5) The judgment of the utility of attribute splitting will improve the method in the VFDT algorithm, and use the method of fuzzy information gain to periodically detect the optimal splitting attribute. Due to the large amount of noisy data or non-deterministic information in the actual network data stream, even for some discrete attribute fields, the traditional steep attribute splitting method will lead to unclear classification boundaries. Therefore, the improved algorithm uses fuzzy information gain as the attribute selection metric for both discrete attributes and continuous attributes.

(6) Substitute the control of the growth process of the subtree.

To improve memory utilization, the replacement subtree also needs to be controlled in size and properly pruned. The judgment criterion for pruning is based on the increment of classification accuracy between the atomic tree and the sub-tree. The optimized algorithm stipulates that if the replacement subtree satisfies formula (2.4), it will be retained; otherwise, the replacement subtree will be deleted. Where \(i\) is the node number in the original tree and the replacement tree, and \(\lambda\) is the replacement subtree retention threshold.

\[
\Delta_{\text{accuracy}} \left( \frac{l'_{\text{test}} + l'_{\text{alt}}}{l'_{\text{test}}(\text{accuracy})} \right) \geq \lambda
\]

In the above formula, accuracy can be defined by the ratio of the number of samples correctly classified by the node to the total samples flowing through the node, namely:

\[
l'_{\text{test}}(\text{accuracy}) = \frac{n'_{\text{right}}}{n'_{\text{all}}}
\]

4. Algorithm evaluation

4.1. Introduction of experimental data set

The MIT Lincoln Laboratory simulated the typical local area network system of the US Air Force over a period of 9 weeks and collected about 5 million records of TCP connection data. The data set contains various types of intrusion data that are widely used in the military environment (the specific data is omitted). Use this data to simulate and verify the data before and after optimization.

4.2. Simulation experiment results and analysis

4.2.1. Comparison of classification accuracy. The simulation results show that with the increase of classification attributes, the classification accuracy of the two systems shows an upward trend, which means
that whether it is rule pattern matching or dynamic decision tree classification, the classification accuracy is dependent on the number of rules or attribute fields Increase. The effect diagram after MATLAB [10] simulation is shown in Figure 1.

Figure 1. Experimental comparison of the two algorithms and the change of sample concept drift percentage.

Analysis of the results: The figure shows that the classification accuracy of the Snort system is higher than the optimized algorithm system at the beginning, and the latter gradually surpasses the former as the number of classification attributes increases. The reason for this is that the Snort system method is mainly based on the pattern matching algorithm, and the establishment and update of the rule base requires manual maintenance by domain experts. At the beginning, because each data flow rule in the Snort system was compiled by experts, the accuracy of the existing data flow detection can reach 100%, but because the system does not have self-learning function, over time And the increase of selected attribute fields, many hidden unknown data types cannot be found in time, and the rule base cannot be dynamically updated in real time; and the optimized algorithm is a dynamic incremental decision tree algorithm based on data mining. The rules can be said, but the construction of the rule base requires no or little human intervention. As the number of incoming samples increases, the rules will be established one by one, and the classification decision tree will become more accurate with the increase of classification attribute fields; as can be seen from Figure 3.1, with the increase of classification attributes, samples with concept drift The number will increase, the concept drift of the optimized system VFDT algorithm is significantly better than the system before the improvement.

4.2.2. Analysis of algorithm performance after optimization. The part of the core decision tree in the optimized algorithm is still based on the Hoeffding inequality. The Hoeffding bound is used to achieve efficient incremental classification with the smallest possible sample, and the total memory requirement is \(O(ndvc)\). Where \(n\) is the total number of nodes in the original decision tree and replacement tree, \(d\) is the number of attributes, \(v\) is the maximum number of attribute values, and \(c\) is the number of classes. Due to the introduction of sliding window technology, all samples in the sliding window can be put into memory and the window mechanism can be used to judge the classification effect of the system on peripheral data in real time. Therefore, the optimized algorithm is an efficient statistical method based on memory, and its efficiency has nothing to do with the total number of samples. In addition, the introduction of sliding windows also makes the classification model generated by the optimized system always adapt to the current situation, and improves the impact of concept drift on the previous classification model. The time complexity required by the sliding window to process each sample data is \(O(l_dvc)\), where \(l_\gamma\) is the depth (number of layers) when the sample was finally classified in the original decision tree.
Figure 2. Comparison of algorithm running time.

Figure 3. Comparison of algorithm memory usage.

Result analysis: According to the analysis results of the algorithm running time and memory usage, the Snort algorithm and the optimization algorithm have little difference in performance when the test samples are small (≤ 100000); but as the number of samples further increases, the optimization algorithm is significantly better than the former in both time and space. The memory usage of the optimized algorithm is approaching 41.7%, and Snort is approaching 54.9%.

5. Conclusions
Classification or prediction is one of the basic problems of data mining. There are many mature algorithms for the classification and mining of traditional data sets. Data flow classification and frequent pattern mining cannot use static traditional mining algorithms. Since class definitions change with time, it is necessary to consider the characteristics of data flow changes and delete outdated class definition patterns in time. This paper reduces the impact of concept drift and network noise by adding improved sliding window technology and fuzzy technology to the traditional streaming data classification and mining algorithm VFDT. Experimental results show that the algorithm can effectively improve the accuracy of streaming data classification and mining, reduce classification errors, and improve the rationality and practicability of the algorithm.

References
[1] Jiawei Han. Data mining concepts and technologies [M]. Beijing: Machinery Industry Press, 2008. 3-5, 14-18, 25, 188, 192, 316, 307-320.
[2] Xu Yonghong, Yang Yun, Cao Lixin, etc. Design of Intrusion Detection System Based on Data Mining [J]. Computer Engineering and Application, 2002, 11 (24): 24.
[3] Li Guohui, Chen Hui. Mining frequent patterns in arbitrary sliding time windows of data streams [J]. Journal of Software, 2008, 19 (10): 2585-2596.
[4] Jawerth B, Sweldens W. An overview of wavelet based multi-resolution analyses [J]. SIAM Rev, 1994, 3G (3): 377-412.
[5] Wang Tao, Li Zhoujun, Hu Xiaohua, etc. An efficient incremental decision tree classification algorithm for data stream mining [J]. Journal of Computer Science, 2007, 30 (28): 1244-1249.
[6] G. Hulten, L Spencer, P Domingos. Mining time-changing data stream [R]. International Conference on Knowledge Discovery and Data Mining, 2001.
[7] L Breiman, J H Friedman, RA Olshen. Classification and Regression Trees [J]. CA, 1984: 125-163.
[8] P Domingos, G Hulten. Mining high-speed data streams [R]. International Conference on Knowledge Discovery and Data Mining, 2000.
[9] Fayyad U M, Irani K B. On the handling of continuous-valued attributes in decision tree generation on learning [J]. Machine Learning, 1992, 9: 87-102.
[10] Zou Tao. Research on key technologies of intelligent network intrusion detection system [D]. Hunan: National University of Defense Technology, 2004.65-67.
[11] Chen Jie. MATLAB Collection [M]. Beijing: Electronic Industry Press, 2007.269-290.