Parallel algorithm of flow data anomaly detection based on isolated forest

YueLiu
College of Computer and Information
Hohai University
Nanjing, China
997248890@qq.com

Yuansheng Lou
College of Computer and Information
Hohai University
Nanjing, China
wise.lou@163.com

Sipei Huang
College of Computer and Information
Hohai University
Nanjing, China
1992559195@qq.com

***Abstract*—The isolated forest algorithm is improved and applied to the hydrological field. The parallel anomaly detection algorithm (Flink-iForest) is proposed. At the same time, the k-means algorithm is combined to solve the problem of Flink-iForest threshold division and improve the stability of anomaly detection results. Through various experiments and real hydrological data, first of all, the Flink-iForest algorithm is verified in terms of accuracy, efficiency and scalability, and compared with the standard SKlearn-iForest and PIFH algorithm; finally, the effectiveness and efficiency of Flink-iForest algorithm are proved by experiments.**

Keywords-component; isolated forest algorithm; anomaly detection; Flink; parallelization

I. INTRODUCTION

In the field of hydrology, hydrological data is a time series data which increases with time. Hydrological data contains a lot of useful information. It can not only be used for the prediction and early warning of future hydrological data, but also the hidden abnormal data will affect the analysis and decision-making. However, with the data soaring, new requirements are put forward for anomaly detection technology in the field of hydrology.

In recent years, there have been some research results on how to detect a large number of hydrological data in real time. Wu Yafei [1] proposed a time series anomaly detection algorithm based on Hadoop anomaly detection distributed computing, which solved the problem that the traditional detection algorithm could not integrate local solutions, and then solved the memory bottleneck problem. However, the efficiency of the algorithm still needs to be improved, and it is not suitable for the flow data environment; Tian Lu [2] proposed a stream data mining algorithm based on spark streaming for the first time. By setting the size of the sliding window to select the current user data stream, and by using the stream clustering algorithm to get the current power consumption behavior mode of the user, fast detection can be realized. However, spark streaming is based on micro batch operation, which does not really realize the real-time processing of the convection data, and cannot accurately consume the convection data once, which will inevitably lead to the accuracy of the results; Song Po [3] first proposed a real-time network data flow anomaly detection algorithm based on storm, which solved the problem of feature selection and added a new evaluation standard, and also achieved the parallel processing of flow data. However, the same problem still exists, which can not guarantee the accurate one-time consumption of flow data, and the accuracy of the algorithm is slightly inferior to other parallel algorithms. Therefore, this paper proposes to further excavate the outliers in hydrological time series and explore the parallel anomaly detection algorithm based on Apache Flink platform.

II. APACHE FLINK

Apache Flink is an open source flow processing framework developed by the Apache Software Foundation. Its core is Cloth data processing engine. Flink executes arbitrary stream data programs in data parallel and pipeline mode, and Flink's pipeline runtime system can execute batch processing and stream processing programs. In addition, Flink's runtime itself supports the execution of iterative algorithms. Apache Flink's data flow programming model provides single event-at-a-time processing on limited and infinite datasets. At the basic level, the Flink program consists of flows and transformations.

III. ISOLATED FOREST ALGORITHM

The isolated forest algorithm mainly catches the small proportion of abnormal points in the whole data set and these points will deviate from the characteristics of the data center. For these isolated points, iForest has a very efficient strategy.

In a data space, the random hyperplane is used to segment it, and a single segmentation can be divided into two data subspace, the same step, continues to segment each subspace until there is only one data point in each space.

978-1-7281-8288-9/20/$31.00 ©2020 IEEE
DOI 10.1109/AIEA51086.2020.00035

The segmentation method of data space is the most eye-catching part of iForest. Because each segmentation is completely random, Monte Carlo method can be used to get a convergence value. iForest is composed of n itrees (binary trees).

In the detection phase, let every data traverse the iTree in the forest, we can get the node location of each data in each tree, and calculate the height of each data in each tree according to the node location. As shown in Figure 1, if the node is farther away from the root node, it means that this data is very likely to be a normal value, otherwise, it is very likely to be an abnormal value, and the red point is much more likely to be an abnormal value than the blue point.

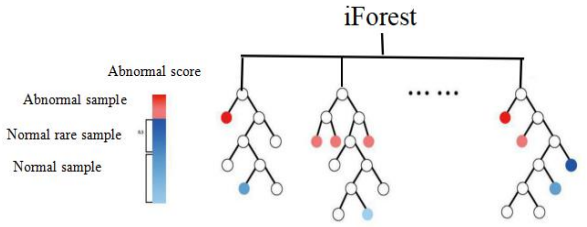

Figure 1. abnormal scores of isolated forest

IV. THE IMPROVEMENT OF ISOLATED FOREST ALGORITHM

A. Problems of isolated forest algorithm in hydrological field

Firstly, only a subset of the training set needs to be randomly selected to build iTree. Secondly, the iForest algorithm has linear time complexity, low constant and low memory requirements. Finally, the iForest algorithm is based on the idea of integration. Even if the efficiency of some itrees is not very high, the integration algorithm can always transform weak algorithm into strong algorithm.

iForest algorithm is a very promising anomaly detection algorithm, which has significant advantages. However, in the actual environment requirements, the isolated forest algorithm still needs several improvements:

- All the iForest algorithms described above are based on the static data set of single machine, and the data scale is limited and controllable. In the actual environment, the sensor data of hydrological test station will be updated every 5 seconds, so the data source is continuous, uncontrollable and unpredictable. Then whether the iForest algorithm can guarantee its detection accuracy in the actual hydrological time series data. According to this situation, a new anomaly detection framework of iForest algorithm for streaming data is proposed.
- In some special application scenarios, many operations need to protect the load balance and stability of the system, if the experiment can only be carried out in a stand-alone environment. Then the memory or disk of the machine will surely be full one day, and the computing efficiency of a single machine is limited, so a new requirement is proposed, and the iForest algorithm based on the flow environment is parallelized on the Flink platform. This paper will also demonstrate that the improved iForest algorithm framework and other improved algorithms have a greater improvement in accuracy and efficiency through experiments.
- Complete randomness brings advantages to iForest algorithm, but also leads to some defects. For example, the threshold value of each calculation exception may change, which will lead to the decrease of the accuracy of the detection results or the phenomenon of misjudgment.

B. Research on parallel algorithm of isolated forest based on Flink

1) General framework of anomaly detection of hydrological flow data

In the Flink platform, the first two modules are: 1) stream handler; 2) window handler. The stream handler applies a mapping function to each stream object and sends it to the window. The window processor runs an outlier detection algorithm in each slide. In the map function of the flow handler, each hydrological data point is sent to a single partition window. The window has its own storage record state, which is persistent in the slide. In summary, the content of a window at any point in time contains all the active points and their metadata. The former keeps the active point in its partition and allows some temporary replication, as described below, while the latter keeps the final metadata in each slide. The metadata also includes information about outliers. Because they have evolved for new points and old points in each slide, the window state is completely updated, so the rolling window semantics is applied. Here, the rolling window is used as a meta window.

When the hydrological flow data reaches the sliding window, the initial detector is trained by iForest algorithm based on the multi window data. In the test phase, the exception detector is used to detect each instance in the sliding window, and judge whether it is an exception point according to the exception score of the instance. After an example in a sliding window is completed, the statistical results show that if the anomaly rate of the sliding window is less than the threshold u, the concept drift does not occur at this time, and the trained anomaly detector will not change. Otherwise, it means that the concept has changed, the trained anomaly detector needs to be modified and retrained, and then the detector is updated and retrained according to all instances in the current sliding window, so as to discard the previous detector.

2) Adaptive threshold determination of exceptions

This paper proposes a method based on k-means clustering to improve the iForest algorithm, and can adaptively divide the abnormal threshold according to the iForest model trained in real time. The anomaly score calculated by the Flink-IForest algorithm is a binary classification problem, then the parameter k value in the experiment is taken as 2.

V. PARALLEL IMPLEMENTATION OF IMPROVED ISOLATED FOREST ALGORITHM IN FLINK PLATFORM

A. Construction of parallel training model

The process of ParameterMap operation on sampled data is parallel, and Flink's map() operator performs parallel calculation. There are characteristic sampling in map (), iTree height limit calculation, recursive iTree construction and so on.

Flink-IForest will accept a parameter (maxFeatures) for feature sampling. The next step is more important, because the height of the tree will directly affect the performance of subsequent anomaly detection. The height of the tree must be restricted so that the algorithm ends early. The height of the tree will be controlled by maxDepth. When the code is implemented, an integer number of log2 (downward) is taken according to the number of data in each sample subspace. This number is the maximum height limitPath of the tree, and the final height is the smaller of the two.

Constructing iTree is the key part of realizing Flink-iForest. It is a recursive process. When a leaf node is generated, the recursive cycle exits, and the leaf node stores the specific data instances. More importantly, the training data is divided and encapsulated into the internal nodes of iTree, and the left and right subtrees of internal nodes are recursively generated.As shown in Figure 2.

B. Implementation of parallel anomaly detection

The Flink-iForest algorithm implemented in this chapter can reflect the use of the version after Flink 1.8.0 in the phase of parallel exception detection and detection. In the process of parallel detection, iForest model will be forwarded to each TaskManager through broadcast, and each piece of data in the sliding window will traverse all the itrees for exception evaluation, And anomaly detection is carried out by the calculated anomaly score and adaptive threshold.

Here, we use the ScalarFunction of Flink Table, through which we can get instances with different eigenvalues, and then through iForestModel, we can get the avgHeight of the instances in the tree, and score the exceptions according to the avgHeight.

Vector container gets the eigenvector of the data instance. First, the average height of iTree is calculated by the number of samples; then the average value of the sum of the height of each tree in the forest is calculated by iForestModel . If the leaf node of the tree is encountered in the process of traversal, the height of the data in the tree will be returned directly; if the leaf node is not encountered, the characteristic value of it will be judged; if it is greater than the characteristic index of the internal node, the right child will be accessed recursively; otherwise, the left child will be accessed recursively. Finally, we need to implement the transform() method.

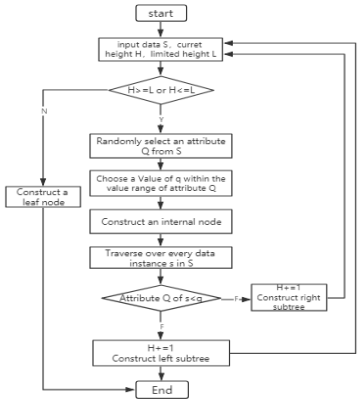

Figure 2. iTree Construction flow chart

C. Experimental results and analysis

Figure 3 shows the daily average water level of Liuhe hydrological station in Chuhe River Basin, with obvious abnormal points.

1) Performance comparison experiment

Firstly, the AUC value of the algorithm is compared from the accuracy of the algorithm. In the single core configuration environment, compare the stand-alone iForest[4]、PIFH、SKlearn-iForest and Flink-iForest in the original paper, and the specific results are shown in Figure 4.

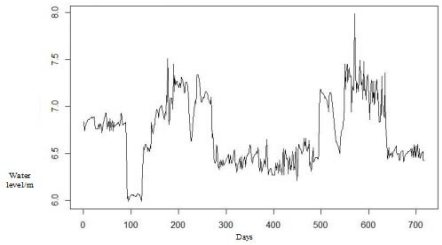

Figure 3. Daily average water level of Liuhe Station in Chuhe River Basin

It can be seen from the experimental result comparison chart that the improved Flink-IForest algorithm in this paper has a lower AUC value in the specific scenario of hydrological data than the original paper [4], but it is lower than that of PIFH and SKlearn-IForest. Higher AUC values may be limited by insufficient anomaly detection in the algorithm itself. The algorithm implemented in this paper meets the requirements of anomaly detection in terms of accuracy.

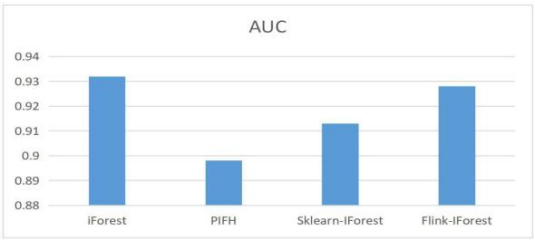

Figure 4. AUC value comparison of different algorithms

Then we need to compare the execution time of the algorithm. The first group of experiments is: under the same CPU condition, the running memory of SKlearn-iForest is limited to 64G; the second group of experiments is: when Flink-iForest algorithm submits a task, it sets a single node to 2G memory, and the number of CPU cores is from 1 core to 4 cores. The third group of experiments: PIFH algorithm uses the same experimental operation and environment as Flink-iForest algorithm, and the experimental results are shown in Figure5.

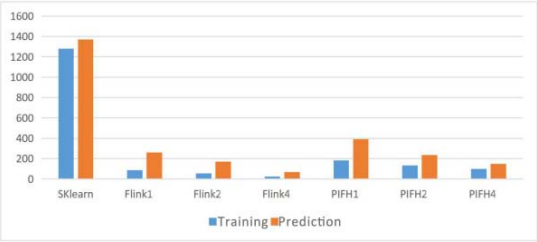

Figure 5. Algorithm performance comparison chart

The experimental results show that in a single-core environment, the time cost of the SKlearn-IForest algorithm in the training phase is 15 times that of the Flink-IForest algorithm. In the prediction phase, the time overhead SKlearn-IForest algorithm is 5 times that of Flink-IForest. The PIFH algorithm obviously saves more time than the SKlearn-IForest algorithm, but there is still a gap with Flink-IForest. As the degree of parallelism continues to increase, the advantages of Flink-IForest are increasing. In the 4-core case, Flink-IForest's prediction phase takes 68 seconds, which is three times faster than the single-core case. The detection time of the PIFH algorithm is only 2 times faster than the single-core, and it takes more time than Flink-IForest.

To sum up, the single version of Flink-iForest algorithm performs better than the standard SKlearn-IForest algorithm and PIFH algorithm. With the increasing parallel of algorithms, the advantages of Flink-iForest algorithm are more prominent.

2) Expansibility experiment

The parallelism of the Flink-IForest algorithm can increase as the number of TaskManagers increases. Continue to use the same streaming data set, the memory on the Flink platform is set to 2G for each running node, and the number of cores (parallelism) continues to increase from 1 core to 4 cores. Experimental parameters: 100 iTrees, the sample subspace size is 256. In the case of increasing parallelism, the time of the Flink-IForest algorithm during the training phase and the execution time (seconds) of the anomaly detection phase decrease rapidly, and the results are shown in Figure 6.

From the graph, it can be found that the training and detection time decline very fast, which can reach the logarithmic decline speed. It shows that Flink-iForest algorithm has more and more performance advantages in the situation of increasing parallelism, and meets the requirements of algorithm in big data environment.

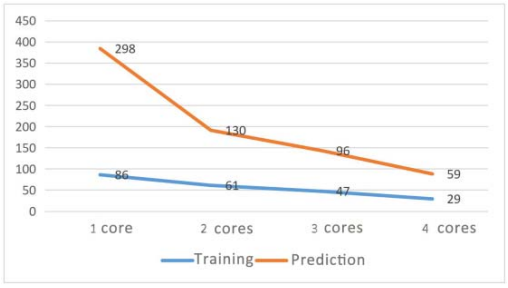

Figure 6. expansion curve of Flink-iForest algorithm

VI. SUMMARY

In this paper, anomaly detection is carried out for hydrological time series data. This paper mainly introduces the parallel implementation of iForest algorithm based on Flink. Firstly, it introduces the characteristics of Flink and the improvement of iForest algorithm in the field of hydrology. Then it realizes the training model and the parallel process. Finally, the validity and efficiency of Flink-iForest algorithm are proved by experiments.

References

[1] Wu Yafei. Time series anomaly detection based on distributed computing[D]. Shanghai Jiaotong University, 2016.

[2] Tian Lu. Research on online detection of power anomaly based on Spark Streaming[D]. North China Electric Power University (Beijing), 2019.K. Elissa, “Title of paper if known,” unpublished.

[3] Song Bo. Research on anomaly detection technology of real-time network data stream based on Storm[D]. Harbin Engineering University, 2016.

[4] Liu F T, Kai M T, Zhou Z H. Isolation Forest[C]. Eighth IEEE International Conference on Data Mining. IEEE, 2008: 413-422.

[5] Alcalde-Barros A, García-Gil, Diego, García, Salvador, et al. DPASF: A Flink Library for Streaming Data preprocessing[J]. 2018.

[6] Hou Yongxu, Duan Lei, Qin Jianglong, et al. Design of parallel anomaly detection based on Isolation Forest[J]. Computer Engineering and Science, 2017, 39(2): 2

[7] Deng Jianming. Discussion on the way to maximize the value of hydrological big data[J]. Water Resources Development Research, 2015(05):55-58.