Log Anomaly Detection Based on Bi-LSTM Feature Extraction

1st Yue Zhang
School of Electrical and Information Engineering
1st *Zhengzhou University*
Zhengzhou, China
zhangyue_o@163.com

2nd Dong Zhang
System Software R&D Dept.
2nd *Inspur Electronic Information Industry Co.,Ltd.*
Jinan, China
zhangdong@inspur.com

3rd Feng Guo
System Software R&D Dept.
Inspur Electronic Information Industry Co.,Ltd.
Jinan, China
guofengbj@inspur.com

4th Xiaotong Wang
System Software R&D Dept.
Inspur Electronic Information Industry Co.,Ltd.
Zhengzhou, China
wangxiaotong@inspur.com

5th Yihai Duan
System Software R&D Dept.
Inspur Electronic Information Industry Co.,Ltd.
Zhengzhou, China
duanyihai@inspur.com

6th Xiaonan Zhang
Compute Science and Technology
Harbin Institute of Technology,Weihai
Weihai, China
2200400405@stu.hit.edu.cn

2022 5th International Conference on Data Science and Information Technology (DSIT) | 978-1-6654-9868-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/DSIT55514.2022.9943901

***Abstract*—With the development of the Internet and the digital age continues to advance, traditional operation and maintenance methods can no longer meet the maintenance of large-scale distributed systems, AIOps emerges as the times require. In AIOps, finding anomalies is a top priority. PCA (Principal Component Analysis) is a traditional machine learning algorithm, which is used for log anomaly detection using the dimensionality reduction feature of PCA algorithm, the advantages of this detection method are: fast detection and simple model. However, the anomaly detection effect is not satisfactory, because when extracting the features of log data, the sequence features and semantic features cannot be sufficiently mined, resulting in a lower accuracy rate. For this problem, we propose a feature extraction improvement model. This model is based on Bi-LSTM (Bidirectional Long Short Term Memory) deep network to improve the feature extraction method of PCA model, extracting semantic features of log sequences by exploiting the ability of long short-term memory network to grasp contextual information, and feed the vector representation matrix containing the semantic features into the PCA anomaly detection network for anomaly detection. The model is validated on HDFS open source dataset and Inspur server dataset, and the results show that the F1-measure of this model is 36.3% higher than that of the traditional PCA model, and 1.1% higher than that of the single Bi-LSTM model.**

Keywords—anomaly detection, Bi-LSTM neural network, PCA, log feature extraction, AIOps

I. INTRODUCTION

As the development needs of various industries become more extensive and require more diverse services, modern systems are increasingly moving toward large-scale distributed systems. Distributed systems can provide greater system capacity, higher scalability, modular operations, etc. to satisfy increasing business volume. However, the distributed system architecture is huge and the modules call each other, resulting in errors in one node that may affect the service of the whole system. High complexity of manual maintenance system, and need to spend a lot of time and energy to ensure the normal operation of the system, so this approach is increasingly unable to meet the maintenance of large-scale distributed systems, developing AIOps is the key to solving the problem. In AIOps, the most fundamental thing is to detect anomalies. A large number of logs are generated during the system operation, which record the operation information of each node, network connection status and important events occurring in the system, and analyzing whether there are anomalies in the system through logs has become a hot topic of research in recent years.

Many studies have been devoted to automated log analysis techniques, integrating statistical analysis and machine learning into log anomaly detection techniques. Typical algorithms include PCA [1], Log Clustering [2], Decision tree [3], etc. These algorithms calculate the count frequency of log sequences and obtain the event count matrix when feature extraction is performed on log sequences. The limitation of the event count matrix is that it does not contain semantic relationships between contexts, and cannot sufficiently extract log sequence features, leading to inability to accurately determine anomalies in the anomaly detection phase. Currently, scholars are working on using deep learning in anomaly detection efforts. In the field of natural language processing, RNN (Recurrent Neural Network) has achieved better results by combining contextual information in sequences through the characteristics of its self-circulation, and can handle tasks such as sequence annotation, text classification, sentence relationship determination, and machine translation. LSTM (Long Short Term Memory) [4] is improved on the basis of RNN, which is able to capture the dependence of sequence for a longer time range and solve the gradient disappearance problem of RNN, and is widely used in text

Funding agency: This project is supported by the Ministry of Industry and Information Technology Software Special Project (2105-370171-07-02-860873). Project affiliation: Inspur Electronic Information Industry Co., Ltd.

Authorized licensed use limited to: Malardalen University. Downloaded on February 13,2023 at 10:52:50 UTC from IEEE Xplore. Restrictions apply.

processing. Du et al. [5] first used LSTM networks in the work of log anomaly detection by proposing the DeepLog model. This model is an unsupervised prediction-based model that has been widely used in subsequent studies.

Comprehensive analysis of the above, in order to improve the feature extraction ability of traditional anomaly detection models and to take advantage of deep learning, we propose a model based on Bi-LSTM network fused with PCA algorithm. In the feature extraction stage, the model uses the ability of the bidirectional long short-term memory network to grasp contextual information, obtains semantic relationships between sequences, and represents log sequences as a feature matrix containing semantic relationships. The PCA algorithm is used in the anomaly detection stage to map the extracted feature matrix to a low-dimensional feature space and determine anomalies by calculating the deviation of each data point from other data in different dimensions of the feature space. The main work of this paper:

(1) We propose an anomaly detection model of Bi-LSTM network fused with PCA algorithm, which improves the feature extraction mode of traditional anomaly detection methods;

(2) Applying this model to AIOps scenarios, do anomaly detection on Inspur sever logs, and good results are obtained, which verifies the adaptability of the model;

(3) Compared with the traditional PCA anomaly detection model, the present model has significantly improved F1 measure on the HDFS dataset.

II. Related Work

In AIOps, the most fundamental thing is to find anomalies, which can be detected through the metric data and log data generated during the system operation. Metric data is time-series data, also known as KPI series, and usually contains business metrics and performance metrics. Xu et al. [6] proposed the Donut algorithm, which is an unsupervised KPI anomaly detection method based on VAE. The algorithm was applied to the KPI researched by a top global Internet company and obtained a high F-score. Su et al. [7] proposed a hybrid model of KPI anomaly detection based on GRU-GAN, using GRU network as the generator and discriminator of GAN network, through adversarial training, obtained the time correlation and data distribution of KPI, and better performance results are achieved in the KPI anomaly detection task. Compared with metric data, log data is unstructured data, and its complexity is higher. Anomaly detection of log data generally includes three steps: log parsing, feature extraction, and anomaly detection. The work of this paper is to perform anomaly detection based on log data.

A. Log parsing

The main job of log parsing is to transform unstructured raw logs into structured log templates, and several log parsing tools exist. He et al. [8] proposed the Drain log parser, which can accurately and efficiently parse raw log messages in a streaming manner with online parsing capabilities by building a fixed-depth tree to extract templates and assign them to log groups of leaf nodes. The Spell method proposed by Du et al. [9] divides logs into different groups by computing the longest common subsequence of multiple log messages and extracts templates to achieve online parsing in a streaming manner. In paper [10], Zhu et al. conducted comparative experiments on 13 dominant log parsing methods, using them on 16 different raw log datasets to verify the effectiveness of log parsing methods in terms of accuracy, robustness, and efficiency, and to demonstrate the effectiveness of the Drain method. Therefore, considering the merits and demerits of several log parsers, we use Drain as the benchmark for log parsing in our experiments.

B. Feature extraction and anomaly detection

Before performing anomaly detection on a log template, it needs to encode the log template in text form into a vector form that can be used as input to the anomaly detection network. In this step, algorithms or networks can be used to make the transformed vectors contain semantic information, also called feature extraction, to improve the accuracy and efficiency of anomaly detection.

One way to encode text into feature vectors is to generate event count matrices [11], first divide logs into different groups to generate log sequences, then calculate the number of occurrences of each log template in the log sequence, convert log sequences into event count vectors according to the number of occurrences, and finally concatenate the count vectors of all sequences to form event count matrices. One-hot is a relatively simple way of encoding words, where all the different words in the text are formed into a vocabulary, and the words are encoded into a vector of the size of the vocabulary according to the index of the word in the vocabulary, with the index position of the word in the vector being 1 and the rest being 0. This encoding is simple and efficient when the vocabulary is short, but in real situations the vocabulary is large, resulting in a high vector dimension and sparse encoding that is not easy to compute. Word2vec is a more efficient method for extracting text features, which concept is to map one-hot encoded sparse matrices onto a low-dimensional space, and new word vectors can be computed using the Skip-gram model or the CBOW model.

Some traditional machine learning anomaly detection models [11], such as PCA and SVM , first generate an event count matrix before performing anomaly detection. Meng et al. [12] proposed the LogAnomaly model, which improves on the feature extraction approach used by Deeplog by proposing template2vec encoding based on word2vec, which takes synonyms and antonyms in log templates into account and encodes log templates as vectors containing positive and negative words and contextual semantic features, similar to Deeplog using LSTM networks as a baseline for unsupervised anomaly detection based on prediction. Lu et al. [13] first used CNN in log anomaly detection work, proposing logkey2vec to encode log templates as feature vectors, using three convolution kernels of different sizes to convolve the feature vectors, filtering out weakly correlated features through a maximum pooling layer, and finally connecting fully connected layers to produce prediction results, a supervised model. Sun et al. [14] proposed a CNN-BiLSTM model, which combines the advantages of convolutional neural networks and recurrent neural networks, using CNN to extract space features of log parameters and Bi-LSTM to extract time features of log templates, achieving a high accuracy rate. Zhang et al. [15]

Authorized licensed use limited to: Malardalen University. Downloaded on February 13,2023 at 10:52:50 UTC from IEEE Xplore. Restrictions apply.

proposed the LogAttn model, which combines an auto-encoder with an attention mechanism, an unsupervised log anomaly detection method, which achieved the best comprehensive performance on three real datasets; Li et al. [16] proposed an anomaly detection method based on Bi-LSTM, which combines Bert natural language processing model and Bi-LSTM model based on attention mechanism, and applied the method to HDFS and OpenStack datasets, making the accuracy and recall has been greatly improved.

III. METHOD

A. Model Framework

The anomaly detection model proposed in this paper belongs to the field of supervised learning. The framework of this model is shown in Fig.1 and consists of three main parts, log parsing, feature extraction and anomaly detection.

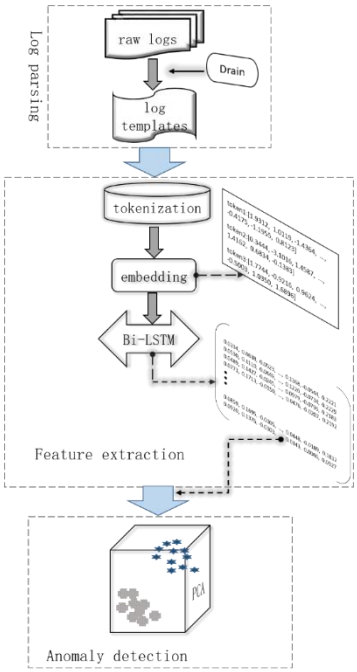

Fig. 1. Model Framework

B. Bi-LSTM based feature extraction method

1) Log templates tokenization processing

a) Sliding window processing

The HDFS dataset divides all log templates extracted by the log parser into different groups according to block id. The Inspur server dataset is grouped according to the logs generated by different servers, and a group is called a log template sequence. Given a sequence of log templates $K = \{k_1, k_2, k_3, \dots, k_n\}$, using a sliding window of size w and step s to divide the log template sequence into different windows $\{w_1, w_2, w_3, \dots\}$.Where w_i denotes the set of w log templates $\{k_1, k_2, k_3, \dots, k_w\}$.

b) tokenization processing

The main job of this section is to convert the log template k_i in window w_i into a series of tokens, as shown in Fig.2. First extract all the tokens in the log templates and compose a vocabulary, assign a unique id to each token in the vocabulary, and represent each token in the templates with the corresponding id in the vocabulary. Since the length of all log templates (i.e. the number of tokens contained in each template) is not exactly the same, resulting in different lengths of the template vectors represented using ids, which is not conducive to subsequent neural network computation, so using padding to represent the log templates as vectors of the same length. In experiments, we padding the vector into 50 dimensions.

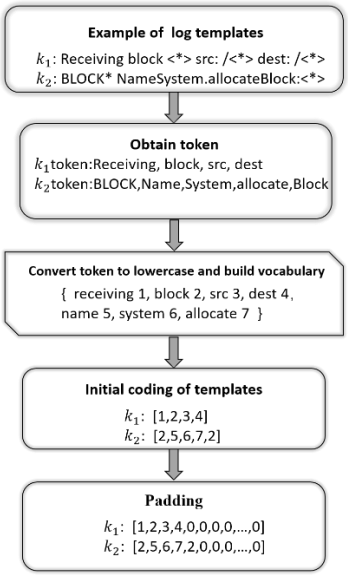

Fig. 2. Tokenization processing

2) Bi-LSTM neural network model

After the handling in subsection *1)*, each window w_i is converted into a $w \times 50$-dimensional vector. Feed the $w \times 50$ dimensional vector into the neural network model to further extract the semantic features and obtain the vector representation matrix.

It first goes through an embedding layer, which embeds the token id in each template vector into the d-dimensional space, and the embedding process uses a random initialization of the d-dimensional vector. After word embedding each template is represented as a $50 \times d$ dimensional initial embedding matrix and used as input to the Bi-LSTM network. Training classification tasks continuously optimize the model parameters to refine the word embedding matrix and obtain a vector representation matrix of log template sequences containing semantic features.

The LSTM network is basically an RNN network with update gates, forgetting gates and output gates added to the RNN network, using these gating units to control what information is updated, discarded and output during backward transmission, enabling it to capture a longer range of dependencies, solving the problem of RNN networks not being able to "remember" for long periods of time. In addition, when the RNN network back-propagation, the gradient decreases exponentially as the layers increase, resulting in the problem of gradient disappearance, while the LSTM network can avoid the gradient disappearance under the control of the gating unit.

Authorized licensed use limited to: Malardalen University. Downloaded on February 13,2023 at 10:52:50 UTC from IEEE Xplore. Restrictions apply.

The Bi-LSTM network consists of a forward loop layer, which extracts sequence information from left to right in the positive direction, and a reverse loop layer, which extracts sequence information from right to left in the negative direction. The bidirectional LSTM network fully learns the semantic features of the contextual sequences from both directions. As shown in Fig.3, the input is a sequence $\{x_1, x_2, \dots, x_{10}\}$ containing 10 template vector. First compute the hidden and cell states $\overrightarrow{h_1}, \overrightarrow{c_1}$for the forward loop of this sequence, then compute the forward $\overrightarrow{h_2}, \overrightarrow{c_2}$ all the way to $\overrightarrow{h_9}, \overrightarrow{c_9}$. The reverse loop starts with the calculation of $\overleftarrow{h_9}, \overleftarrow{c_9}$ and proceeds in reverse up to $\overleftarrow{h_1}, \overleftarrow{c_1}$; $\hat{y}_1$ to $\hat{y}_{10}$ indicates the output of the LSTM block at the corresponding time.

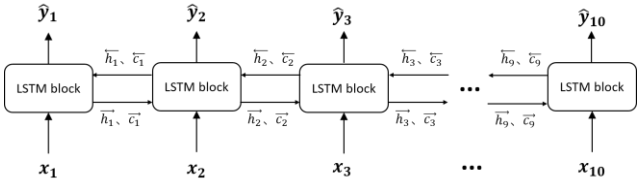

Fig. 3. Bi-LSTM network

The computation of both forward and reverse loops is the forward propagation process of a neural network. When forward propagation is performed in an LSTM network, the calculation is given by:

Cell state candidates:

$$\tilde{c}_t = tanh[(w_{ch} * h_{t-1} + w_{cx} * x_t) + b_c] \quad (1)$$

Update gate:

$$U = \sigma[(w_{uh} * h_{t-1} + w_{ux} * x_t) + b_u] \quad (2)$$

Forget gate:

$$F = \sigma[(w_{fh} * h_{t-1} + w_{fx} * x_t) + b_f] \quad (3)$$

Output gate:

$$O = \sigma[(w_{oh} * h_{t-1} + w_{ox} * x_t) + b_o] \quad (4)$$

Cell state:

$$c_t = U * \tilde{c}_t + F * c_{t-1} \quad (5)$$

Hidden state:

$$h_t = O * tanh(c_t) \quad (6)$$

Where σ is the sigmoid activation function, x_t is the input at time t, b_c, b_u, b_f, b_o are the bias vectors of the corresponding functions, $w_{ch}, w_{cx}, w_{uh}, w_{ux}, w_{fh}, w_{fx}, w_{oh}, w_{ox}$ are the weight matrices for calculating the corresponding values, for example, w_{ch} denotes the weight matrix between the cell state candidate c and the hidden state h when calculating it. Denoting the output at moment t of the forward loop as $\vec{y}_t$ and the output at moment t of the reverse loop as $\overleftarrow{y}_t$, we have:

$$\vec{y}_t = g(w_{\vec{y}}[\vec{c}_{t-1}, \vec{h}_{t-1}, x_t] + b_{\vec{y}}) \quad (7)$$

$$\overleftarrow{y}_t = g(w_{\overleftarrow{y}}[\overleftarrow{c}_{t-1}, \overleftarrow{h}_{t-1}, x_t] + b_{\overleftarrow{y}}) \quad (8)$$

Where g is the activation function, $w_{\vec{y}}, w_{\overleftarrow{y}}, b_{\vec{y}}, b_{\overleftarrow{y}}$ correspond to the weight matrices and bias vectors of the forward and reverse recurrent networks respectively. Then the actual output at time t: $\hat{y}_t = \vec{y}_t + \overleftarrow{y}_t$。

In the Experiment, we stack two layers of Bi-LSTM as the backbone network for feature extraction, as shown in Fig. 4. The input sequence vector is $\{x_1, x_2, \dots, x_{10}\}$, h_0, c_0 are the initial inputs of the hidden and the cell states, h_{10}, c_{10} are the final outputs of the hidden and cell states; the superscripts 1 and 2 of $h_n^1, h_n^2, c_n^1, c_n^2$ correspond to the first and second layer respectively; output is the predicted result of the two-layer Bi-LSTM network, then attaching it to a fully connected layer as a prediction layer, using the cross-entropy loss function as a benchmark for the network back-propagation learning parameters, and updating the parameters using the Adam Optimizer. The training process continuously optimizes the model parameters to reduce the loss value, and takes the h_{10} with the lowest loss value as the final vector representation of this sequence.

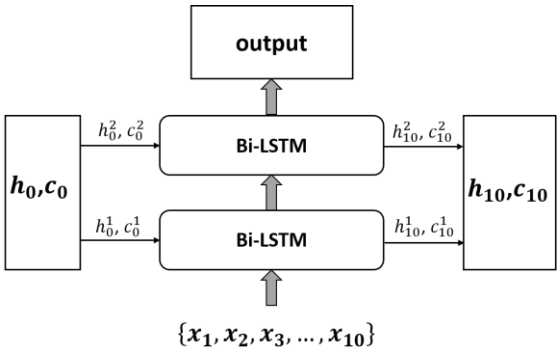

Fig. 4. Two-layer Bi-LSTM network

When the batchsize is set to 2048, the embedding vector d is set to 64, the window size w is set to 10, and the hidden layer size is set to 128, the data dimensional transformations during training are shown in Table I.

TABLE I. DIMENSIONAL TRANSFORMATION

Layer	Output
input:(2048,10)	-
tokenization	(2048,10,50)
embedding	(2048,10,50,64)
dimensional reduction	(2048,10,64)
two Bi-LSTM layers	output: (2048,10,256) h_{10}: (2048,4,128) c_{10}: (2048,4,128)

C. *Anomaly detection*

The input of the PCA network is the vector representation matrix h_n via Bi-LSTM feature extraction network, and the output is the normal anomaly classification of the sequence. The PCA network maps the vector representation matrix to m-dimensional feature space, then calculate the degree of deviation

Authorized licensed use limited to: Malardalen University. Downloaded on February 13,2023 at 10:52:50 UTC from IEEE Xplore. Restrictions apply.

r for each data sample x_i in different dimensions of the feature space. For a certain feature space vector c_j, the degree of deviation is calculated using the following formula:

$$r_{ij} = \frac{(x_i{}^T \cdot c_j)^2}{\lambda_j} \tag{9}$$

Where λ_j corresponds to the feature value of the dimension j feature space vector and acts as a normalization. For sample x_i, calculate the deviation degree of this sample in m feature spaces, and sum up all the deviations to get the final anomaly score of this sample sequence. That is, the anomaly score

$$S(x_i) = \sum_{j=1}^{m} r_{ij} \tag{10}$$

The optimal anomaly threshold t is calculated using the Q statistic in the training process, and when the anomaly score $S(x_i) > t$, label the sample x_i as anomaly; otherwise, label it as normal.

IV. Experimental analysis

A. Dataset

Using the public HDFS dataset [17] and the Inspur server dataset to do the evaluation of the model proposed in this paper. The HDFS dataset is a set of logs generated by the Hadoop distributed system during its operation, containing 11,175,629 logs, each contains a unique identifier block id. The logs can be divided into 575,061 tasks according to the identifiers in the logs, of which 16,838 task blocks are marked as anomalies. The Inspur server dataset is a set of logs generated during the operation of multiple servers within the Inspur Group, containing a total of 5,000 logs, 25 of which are marked as anomaly logs by O&M experts.

B. Evaluation Metrics

Using Accuracy, Precision, Recall and F1 measure to assess the quality of the model, the formulas are as follows:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{11}$$

$$Precision = \frac{TP}{TP+FP} \tag{12}$$

$$Recall = \frac{TP}{TP+FN} \tag{13}$$

$$F1\ measure = 2 \times \frac{Precision \times Recall}{Precision+Recall} \tag{14}$$

Using anomaly samples as positive samples, TP denotes the number of samples that are anomaly themselves and predicted to be anomaly, TN denotes the number of samples that are normal themselves and predicted to be normal, FP denotes the number of samples that are normal themselves but predicted to be anomaly, and FN denotes the number of samples that are anomaly themselves but predicted to be normal. From the formula, Accuracy characterizes the proportion of overall correct predictions of the model, Precision reflects the percentage of successfully detected anomalies to all predicted as anomalies in the sample, Recall reflects the percentage of successfully detected anomalies to all actual anomalies, F1 measure is the summed average of Precision and Recall, with a higher value indicating more better of the model.

C. Comparison experiments

Experiment I Compare the anomaly detection ability of PCA, Bi-LSTM and BiLSTM-PCA models based on HDFS datasets. In the experiment, extracting the event count matrix of the log sequence is used as the input of the PCA model, the output of the Bi-LSTM model is connected to a fully connected layer as a prediction layer, and then the classification result is output through the softmax layer. To improve the efficiency of the experiment, we take out 7,940 task blocks containing 104,815 logs in the original HDFS dataset, and the experimental results are shown in Fig. 5.

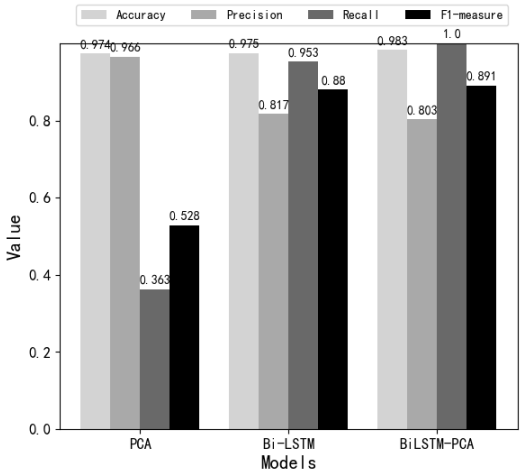

Fig. 5. HDFS log anomaly detection results

Observing Fig. 5, comparing the detection results of PCA and BiLSTM-PCA models, it can be found that the overall effect of this model is significantly better than PCA model, especially Recall is 63.7% higher than PCA model, indicating that the false negative rate of PCA model detection is higher. That is, the number of incorrectly detected the anomaly samples as normal samples is high, while the present model with Recall of 1 is able to detect the anomaly accurately. The results demonstrate that the Bi-LSTM network is able to extract more accurate semantic features compared to the event count matrix. Compared with the traditional PCA model, the Bi-LSTM network based on deep learning has better detection effect, and its Accuracy, Recall, and F1-measure are higher than those of the PCA model, indicating that the deep network Bi-LSTM is effective in the field of anomaly detection. Comparing the Bi-LSTM model with this model, it is found that its Precision is 1.4% higher than this model, but the Accuracy, Recall and F1-measure are lower than this model by 0.8%, 4.7%, and 1.1%, respectively. Compared with the Bi-LSTM model, the anomaly detection ability of the model is slightly improved, because this model uses the PCA network in prediction, which is more advantageous than the fully connected layer.

Experiment II Based on the Inspur server dataset to verify the adaptation capability of PCA, Bi-LSTM and BiLSTM-PCA models, the experimental results are shown in Fig. 6.

Authorized licensed use limited to: Malardalen University. Downloaded on February 13,2023 at 10:52:50 UTC from IEEE Xplore. Restrictions apply.

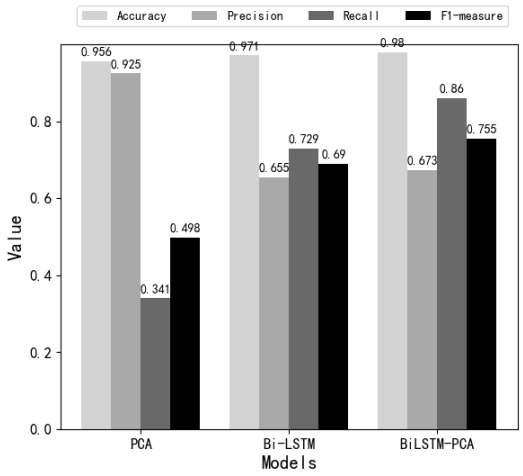

Fig. 6. Inspur server log anomaly detection results

As seen from the above figure, the detection results of this model, Bi-LSTM model and PCA model have decreased due to the more diverse templates and complex forms of tasks extracted from the Inspur server dataset compared to the HDFS dataset, while the block id in the HDFS dataset distinguishes different tasks, making the logs in the log sequence more relevant. But on the whole, the detection effect of this model is higher than that of the PCA model and the Bi-LSTM model, which indicates that this model can adapt to the anomaly detection of different log data.

V. CONCLUSION

In this paper, we propose a log anomaly detection method based on Bi-LSTM feature extraction, which improves the event count matrix of the traditional PCA network in the feature extraction stage, uses a two-layer Bi-LSTM deep network to extract the semantic features of sequences, generates a vector representation matrix as the input of the PCA network, and uses the dimensionality reduction property of the PCA network to classify log sequences as normal or abnormal. The model achieves good results on the HDFS dataset. The comparison experiments in Section IV show that the anomaly detection capability of this model is significantly better than that of the PCA model, which proves the effectiveness of the feature extraction improvement; the model also shows superiority on the Inspur server dataset, which proves the adaptability of the model.

Future research directions: After log parsing the raw logs, we can get the structured log templates and the parameters contained in the logs, but we do not consider the effect of parameters on log abnormalities in the process of doing anomaly detection, so in the future work, we will consider the detection of log parameters, and when the log template and the corresponding parameters are normal, the log is normal.

REFERENCES

[1] W. Xu, L. Huang, A. Fox, D. Patterson, and M. I. Jordon, "Detecting large-scale system problems by mining console logs," In SOSP'09: Proc. of the ACM Symposium on Operating Systems Principles, 2009, pp. 117-131.

[2] Q. Lin, H. Zhang, J.G. Lou, Y. Zhang, and X. Chen, "Log clustering based problem identification for online service systems," In ICSE'16:Proc. of the 38th International Conference on Software Engineering, 2016, pp. 102-111.

[3] M. Chen, A. X. Zheng, J. Lloyd, M. I. Jordan, and E. Brewer, "Failure diagnosis using decision trees," In ICAC'04: Proc. of the 1st International Conference on Autonomic Computing, 2004, pp. 36-43.

[4] Sepp. H, and Jürgen. S, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp.1735-1780, 1997.

[5] M. Du, F. Li, G. Zheng, and V. Srikumar, "Deeplog: Anomaly detection and diagnosis from system logs through deep learning," In Proc. of CCS'17: ACM. Conf. Computer. Commun. Secur., 2017, pp. 1285-1298.

[6] H. W. Xu, W. X. Chen, N. W. Zhao, Z. Y. Li, J. H. Bu, and Z. H. Li, et al. "Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications," In Proc. of WWW'18: World Wide Web Conf., 2018, pp. 187-196.

[7] H. Su, Q. He, and B.Guo, "KPI anomaly detection method for Data Center AIOps based on GRU-GAN," In ICICSE'21: Proc. of 10th International Conference on Internet Computing for Science and Engineering., 2021, pp. 23-29.

[8] P. He, J. Zhu, Z. Zheng and M. R. Lyu, "Drain: An Online Log Parsing Approach with Fixed Depth Tree," In Proc. of ICWS'17: IEEE. Int. Conf. Web. Serv., 2017, pp. 33-40.

[9] M. Du, and F. Li, "Spell: Streaming parsing of system event logs," In Proc. of ICDM'16: IEEE. Int. Conf. Data. Min., 2016, pp. 859-864.

[10] J. Zhu, S. He, J. Liu, P. He, Q. Xie, Z. Zheng, and M. R. Lyu, "Tools and benchmarks for automated log parsing," In Proc. of ICSE-SEIP'19: IEEE/ACM. Int. Conf. Softw. Eng.: Softw. Eng. Pract., 2019, pp.121-130.

[11] S. He, J. Zhu, P. He, and M. R. Lyu, "Experience report: System log analysis for anomaly detection," In Proc. of ISSRE'16: Int. Symp. Softw. Reliab. Eng., 2016, pp. 207-218.

[12] W. Meng, Y. Liu, Y. Zhu, S. Zhang, D. Pei, and Y. Liu, et al. "LogAnomaly: Unsupervised Detection of Sequential and Quantitative Anomalies in Unstructured Logs," In Proc. of IJCAI'19: Int. Joint. Conf. Artif. Intell., 2019, pp. 4739-4745.

[13] S. Lu, X. Wei, Y. Li, and L. Wang, "Detecting Anomaly in Big Data System Logs Using Convolutional Neural Network," In Proc. of DASC/PICom/DataCom/CyberSciTec'18: IEEE. Int. Conf. Dependable, Auton. Secur. Comput., IEEE Int. Conf. Pervasive Intell. Comput., IEEE Int. Conf. Big Data Intell. Comput. IEEE Cyber Sci. Technol. Congr., 2018, pp. 151-158.

[14] J. Sun, J. H. Zhang, Y. J. Bu, B. Chen, N. Hu, and F. Y. Wang, "Log Anomaly Detection Method Based on CNN-BiLSTM Model," Computer Engineering [J], 2021, pp. 1-10, DOI:10.19678/j.issn.1000-3428.0061750.[in Chinese]

[15] L. M. Zhang, W. Z. Li, Z. J. Zhang, Q. N. Lu, C. Hou, and P. Hu, et al. "LogAttn: Unsupervised Log Anomaly Detection with an AutoEncoder Based Attention Mechanism," In KSEM'21: Proc. of the 14th International Conference on Knowledge Science, Engineering and Management, 2021, pp. 222-235.

[16] X. Q. Li, W. N. Niu, X. S. Zhang, R. Z. Zhang, Z. Q. Yu, and Z. M. Li, "Improving Performance of Log Anomaly Detection with Semantic and Time Features Based on BiLSTM-Attention," In Proc. of CECIT'21: Int. Conf. Electron., Commun. Inf. Technol., 2021, pp. 661-666.

[17] "HDFS," Accessed: 2020, https://github.com/logpai/loghub/tree/master/HDFS.

Authorized licensed use limited to: Malardalen University. Downloaded on February 13,2023 at 10:52:50 UTC from IEEE Xplore. Restrictions apply.