Deep Learning Neural Networks with Auto-adjustable Attention Mechanism for Server Fault Diagnosis Under Log Data

Yiyang Xiong(✉), Yajuan Qiao, Shilei Dong, Xuezhi Zhang, and Hua Tan

China Telecom Corporation Limited Research Institute, Beijing, China
xiongyy1@chinatelecom.cn

Abstract. Log data generated by the server describe the system operation information in detail. As a consequence, using algorithms to analyze the log data and realize self-help fault diagnosis has become a research hotspot. There are two kinds of popular automatic fault diagnosis. The first is the traditional machine learning algorithms such as random forest. They can quickly and accurately give the fault diagnosis results after learning the manually extracted feature information. The other is the algorithm using deep neural network, which can complete fault diagnosis end-to-end and obtain better results than the first when the amount of data is large enough. However, these two algorithms have obvious shortcomings: the log file is a time sequence, but machine learning algorithm is unable to capture sequence information. The deep learning algorithm based on recurrent neural network can capture the sequence information, but it lacks the understanding of the system, so the usable conditions are relatively harsh (a large amount of labeled data is required). Therefore, we innovatively combine the advantages of the two algorithms. Firstly, the feature is extracted by machine learning algorithm. Then, the attention mechanism layer is added to the neural network, and the output of machine learning is used as the basis for modifying the attention mechanism parameters. Our algorithm has been verified on the public data set, and achieves an improvement of nearly 3% in F1.

Keywords: Deep learning · Attention mechanism · Fault diagnosis · Log data

1 Introduction

The development of new technologies such as cloud computing, blockchain and sixth generation mobile communication has driven the new demand for servers. In addition, servers are also evolving towards intensive and integrated management. Virtualized server resources bring more efficient resource utilization, but also bring great difficulties to fault diagnosis: too complex internal structure makes it difficult for developers to debug and test accurately. Therefore, how to quickly judge the time and type of fault through intelligent technology is a very meaningful research field in the future.

Log data is the text data generated by the printout code embedded in the program. It records the key information such as variables and execution status when the program is

© The Author(s), under exclusive license to Springer Nature Singapore Pte Ltd. 2023
Y. Wang et al. (Eds.): ICSINC 2022, LNEE 996, pp. 72–79, 2023.
https://doi.org/10.1007/978-981-19-9968-0_9

running. Through it, technicians can locate abnormal requests, track program execution logic, and perform more fine-grained fault diagnosis. As text data, it can not only provide information for Feature Engineering of machine learning, but also belongs to natural language processing widely used in deep learning. Therefore, this paper adopts it as the research goal of fault diagnosis.

Many scholars have used data mining technology to realize automatic log analysis, such as system anomaly detection [1], program verification [2] and system security [3]. Machine learning algorithm represented by random forest can quickly and accurately extract log information and locate faults with the help of artificial feature extraction [4]. However, such algorithms usually cannot capture the timing information of log data. By all appearances, the timing information of the log is very important. For example, many errors will lead to major faults only if they accumulate gradually in a short time.

On the other hand, Log data can be converted into text data in a unified format, and deep learning is the best method to solve the text problem [5]. In addition, deep neural network represented by recurrent neural network [6] and Seq2Seq [7] can extract the timing information of text, Therefore, processing log data through deep learning algorithm may be the best way. Some researchers have applied the deep learning model to the fault diagnosis of log data. They regard the log sequence as text data, and each log corresponds to a word. By training the marked abnormal log and normal log model, their method has achieved one of the best experimental results at present. However, this method relies heavily on the quality and quantity of labeled data and lacks interpretability.

The proposal of attention mechanism leads the current trend of in-depth learning research. It can give different weights to texts in different positions, and has a strong advantage when the text is long [8]. It is easy to think that different behavior data in the log have different effects on whether the fault occurs. Therefore, can we propose an attention mechanism based on the characteristics of log data and give different weights to the data whose type are different on fault diagnosis?

The main contribution of this paper is to add an attention mechanism layer to the neural network based on the features of log data, and the parameter calculation of attention mechanism is realized by the random forest algorithm which is very powerful in feature extraction. The algorithm proposed in this paper achieves a performance improvement of more than 3% over a single neural network algorithm on a public data set.

2 Model Architecture

2.1 Overview

Our algorithm is mainly composed of two parts. One part is the pre-processing module. Its purpose is to convert the log data into a vector that can be understood by the neural network. In addition, it will give the importance parameters of different log. It consists of two parts: convolution and feature engineering. The second part is the neural network module. Because the number of input and output of Seq2Seq can be adjusted at any time, it can adapt to the volatility generation of log data, so we use it as the basic framework of neural network. The difference between our Seq2Seq and other neural network is the addition of attention mechanism, which can be adjusted in time according to the changes of log data, and its parameters are determined by the previous feature engineering.

The innovations of this paper are as follows: 1. Machine learning is used for feature extraction, and the feature is sent to the subsequent deep neural network model as the importance label of each data. 2. Attention mechanism is added to Seq2Seq model, and this parameter is determined by feature engineering. 3. Type coding is added in the word embedding part. The core idea of our innovation is to let the network understand more text information in advance, which is realized by integrating the advantages of machine learning and deep learning respectively.

2.2 Pre-processing

The details of our pre-processing are shown in Fig. 1. Next, we will elaborate it in this section according to the picture.

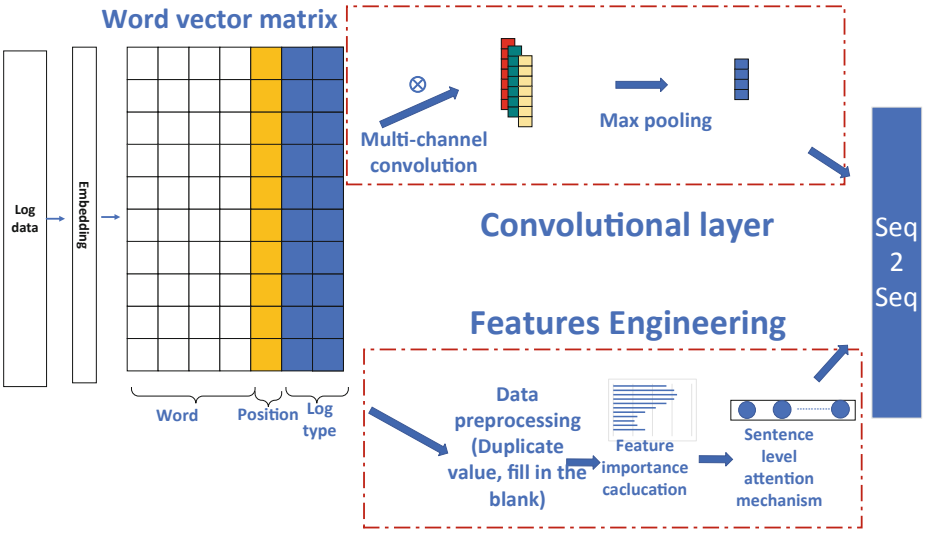

Fig. 1. The input of our preprocessing is log data, and the output is the vector that the neural network can understand and the parameters required by the attention mechanism. It is composed of word embedding, convolution and feature engineering.

Word Embedding. Log belong to text data that is produced continuously, and computer, especially neural network, cannot understand character information. Therefore, in order to make the computer understand the log data, we must vectorize it firstly. Word vectors can be obtained directly from the corpus constructed by other scholars, or they can be obtained by network training as unknown parameters [9]. Because our problem is aimed at the server log data, there are few associated corpora, so we adopt the method obtained entirely by network training. In addition, due to there is little prior information, the convergence speed may be very slow. In order to make the algorithm understand the log data faster, we add location coding and type coding in the embedding layer, which record the timestamp and type of the log respectively. Among them, adding type coding is a innovation proposed in this paper.

Convolution Layer. In order to understand the text information better, we design the convolution kernel with reference to textCNN. Because each row of our word vector matrix represents a log information, we adopt one-dimensional convolution, that is, the data has only one axis, the width of the convolution kernel is equal to the width of the word vector matrix, and will only move in the length direction of the matrix. In this way, our model will analyze data based on a single log. The length of our convolution kernel adopts three types: 3, 4 and 5, so that it can mine the context information of log data from multiple angles and capture a large number of local features. However, its performance ability will be significantly weakened on more than 5 long texts, so more temporal feature extraction will be reflected in the subsequent Seq2Seq network. After obtaining the convolution results, since Seq2Seq with massive parameters will be used later, in order to reduce the system complexity, we added a pooling layer, which can greatly compress the length of word vector and convert the variable length log data obtained from different convolution cores into the same length for subsequent processing.

Features Engineering. Our feature project refers to the routine commonly used in Kaggle competition [10]. Firstly, we need to use Python's pandas to preprocess the data. Because the data has been vectorized, we abandon the complex and difficult to explain steps such as feature construction. Our main task is to remove duplicate values and fill in missing values. Then, we calculate the feature importance through Python's random forest algorithm Xgboost, which can learn the importance of each log data through continuous learning and serial construction of branches. The core of our innovative idea is to give bigger weight to the more important log data that can affect server fault diagnosis, while the log data that has less relationship with fault diagnosis has less weight. Through importance calculation, we can know which log data is more important. Finally, in order to facilitate processing and match the batch size of Seq2Seq, we convert the log data into sentences and send them to the neural network. Therefore, our importance parameters are constructed in sentences (that is fixed sequence length of 30 words). Each circle in Fig. 1 represents the importance parameters of all log data of a sentence.

2.3 Seq2Seq with Attention Mechanism

Our neural network adopts Seq2Seq architecture, which has an encoder, a decoder and a semantic vector. Among them, our core innovation is that the importance calculated by feature engineering has become the parameter of neural network attention mechanism, which changes the network parameters by adjusting the semantic vector C. The specific structure is shown in Fig. 2.

Compared with classical RNN, Seq2Seq can realize the unequal length of input sequence and output sequence, so it can easily realize the transformation between sequences. Therefore, our model can well adapt to the fluctuation of data volume caused by log data generation.

As mentioned earlier, we use convolution layer to extract local information, but due to the limitation of the size of convolution kernel, once the distance between texts exceeds 5, the relationship between them cannot be obtained, and Seq2Seq solves this problem well. The Seq2Seq model can be divided into three parts: encoder, decoder and

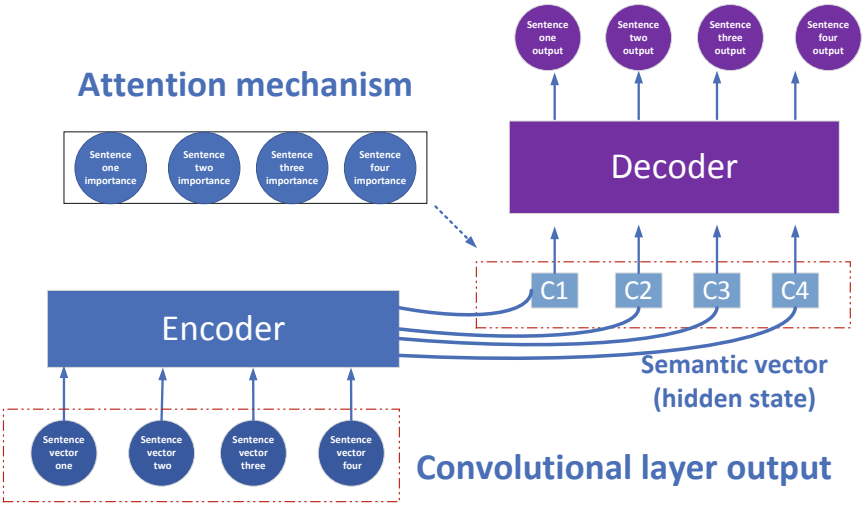

Fig. 2. Seq2Seq model with attention mechanism, its encoder and decoder adopt RNN architecture. Its encoder is input by the result of the convolution layer, the semantic vector is adjusted by the attention mechanism, and its output is executed by the decoder.

intermediate vector. The encoder is responsible for encoding the input sequence and calculating the feature tensor, and the decoder is responsible for receiving the feature tensor and outputting the target sequence. The output result of Seq2Seq at each time will be constrained by all input sequences. It can be seen essentially as a conditional language model, as shown in formula (1):

$$P(Y|X) = P(y_1|x)P(y_2|y_1, x)\ldots P(y_m|y_1, \ldots, y_{m-1}, x) \tag{1}$$

Our Seq2Seq encoder uses the recurrent neural network to calculate the hidden state of the input sequence (usually the last hidden state will be retained). The hidden state will be transmitted to the decoder after passing through a full connection layer. The input of our encoder is the vector obtained after convolution and pooling of the word vector matrix, which is sent to the encoder with a fixed length (a group of 30 log information).

The decoder also adopts the recurrent neural network. During training, it will accept the hidden state provided by the encoder and the output vector of the convolution layer as the input. During prediction, because there is no target sequence, it will accept the output of the decoder at the previous time and the hidden state provided by the encoder as the input, and output the probability distribution predicted at the next time.

The hidden state of the encoder is also called semantic vector because it saves the encoded semantic information. The attention mechanism allows the decoder to selectively obtain the semantic vector information of the encoder. Our selective acquisition here is reflected in giving different weights to different log data: give a larger weight to the log data that is very important in fault diagnosis, and give a smaller weight to the log data that is less important in fault diagnosis. This importance is obtained by the feature engineering of preprocessing. Our attention mechanism formula is described as follows:

$$a = SoftMax(score(Q, K)) \tag{2}$$

where Q, K represent the relationship of a query key value pair. Then, we weight the attention mechanism according to the importance calculated in the earlier stage. The common weighting methods of attention mechanism include multiplicative model, additive model and linear model [11]. Here we adopt the multiplicative model, that is, weighting in the form of multiplication. The formula is as follows:

$$c = \sum_i a_i v_i \tag{3}$$

where a is obtained by formula (2), and v is the matrix composed of log importance parameters calculated in the earlier stage.

3 Experiment

This experiment uses Jupyter notebook as the development tool, python as the language, and NVDIA RTX 3080 for model training and testing. We use the open source log alarm data set of Alibaba Tianchi platform as the experimental object, and judge the performance of different models through the accurate fault diagnosis.

Each log data we obtain has the following information: server number, log recording time, log text information, service mode, failure time, and failure type. In order to focus on the problem concerned in this article: fault diagnosis based on log information, we only record the text information, the sequence of faults and the corresponding fault types. Table 1 intercepts some information of the data we use.

Table 1. Table captions should be placed above the tables.

Time	Log message	Fault type
2020/10/9 8:32	System Boot Initiated BIOS_Boot_Up|State Asserted	CPU-1 fault
2020/8/9 18:44	Memory DIMM031|Presence Detected|Asserted	Memory fault
2020/4/17 14:42	Drive Slot DISK3|Drive Present|Asserted	CPU-2 fault
2020/4/11 8:03	Slot/Connector Port1 Link Down|Slot is Disabled	Other fault

Our experiment uses F1 value as the evaluation index [12]. Its calculation is obtained from the recall rate and precision rate.

Then, due to the variety of log data, in order to facilitate processing and give the importance parameters in batches, we use the random forest model to extract its features. First, we cluster it: after training with labeled samples, our random forest model can return the proximity between samples (the higher the frequency of two samples at the same node, the closer they will be). Then, we set the number of final clusters. This experiment is set to 4, so tens of thousands of log data can be roughly divided into four categories. Finally, we calculate the correlation between it and the classification results according to the four categories, and get the importance parameters of different types of log data. The results are shown in the Fig. 3.

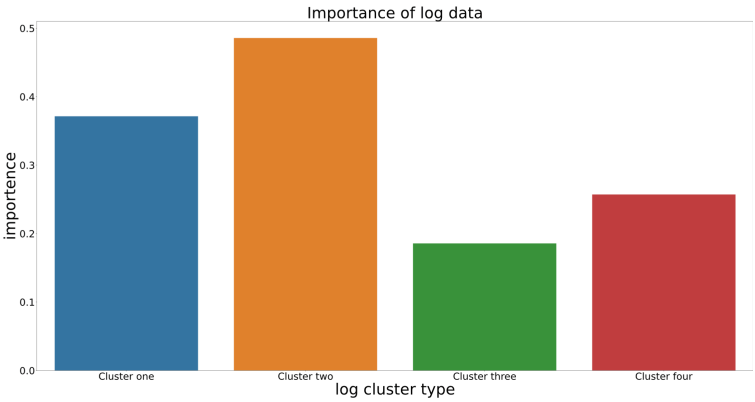

Fig. 3. Importance parameters of different clusters

After obtaining the importance parameters, we can get the experimental results through convolution and subsequent design of Seq2Seq2. We compare this processing system with random forest, logistic regression and Seq2Seq algorithm. The experimental comparison results are as follows (see Table 2):

Table 2. Comparison of results of four models.

Model	Precision	Recall	F1
Logistic regression	0.569	0.541	0.555
LightGBM	0.715	0.665	0.689
Seq2Seq alone	0.756	0.721	0.738
Our model	0.776	0.739	0.757

4 Conclusion

The algorithm proposed in this paper starts from making the neural network learn more features in advance, and realizes the innovation by modifying the attention mechanism parameters and obtaining the parameters through feature engineering. Compared with other neural network algorithm, this algorithm has better interpretability, because it gives different neural networks different degrees of attention according to the types of log data, which is similar to the work of manual processing log faults.

This paper also has some defects in some work: for example, clustering is relatively simple: hundreds of log data types are clustered into four categories, so the importance calculation is also relatively rough. In addition, this method should also add more prior knowledge about fault diagnosis in feature engineering, rather than relying only on random forest model.

Acknowledgment. This article is supported by "6G overall technology research" (Project No. 2020YFB1806601).

References

1. Titonis, T.H., Manohar-Alers, N.R., Wysopal, C.J.: Automated behavioral and static analysis using an instrumented sandbox and machine learning classification for mobile security. U.S. Patent 9,672,355 (2017)
2. Beschastnikh, I., et al.: Leveraging existing instrumentation to automatically infer invariant-constrained models. In: Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering, pp. 267–277 (2011)
3. Oprea, A., et al.: Detection of early-stage enterprise infection by mining large-scale log data. In: 2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, pp. 45–56. IEEE (2015). Author, F.: Article title. Journal **2**(5), 99–110 (2016)
4. Machine Learning: An Artificial Intelligence Approach. Springer Science & Business Media (2013)
5. Liang, H., et al.: Text feature extraction based on deep learning: a review. EURASIP J. Wirel. Commun. Netw. **2017**(1), 1–12 (2017)
6. Zaremba, W., Sutskever, I., Vinyals, O.: Recurrent neural network regularization. arXiv preprint arXiv:1409.2329 (2014)
7. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. Adv. Neural Inf. Process. Syst. 27 (2014)
8. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
9. Xue, G., et al.: Dynamic network embedding survey. Neurocomputing **472**, 212–223 (2022)
10. Bojer, C.S., Meldgaard, J.P.: Kaggle forecasting competitions: an overlooked learning opportunity. Int. J. Forecast. **37**(2), 587–603 (2021)
11. Niu, Z., Zhong, G., Yu, H.: A review on the attention mechanism of deep learning. Neurocomputing **452**, 48–62 (2021)
12. Chicco, D., Jurman, G.: The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. BMC Genomics **21**(1), 1–13 (2020)