ACCURATE INFERENCE OF UNSEEN COMBINATIONS OF MULTIPLE ROOTCAUSES WITH CLASSIFIER ENSEMBLE

Xuan Zhang[1], Longxiang Xiong[1], Ningyuan Sun[1], Mingxia Wang[1], Hao Tang[1], Yanxing Zhao[2]†*

[1]Network Intelligent Laboratory, School of Electronics and Information Engineering, Beijing Jiaotong Unviersity
[2]Beijing Baolande Software Corporation
20120167, 20120138, 20120104, 20120119, 20120107@bjtu.edu.cn, aaron.zhao@bessystem.com

ABSTRACT

Root cause analysis (RCA) of network faults is crucial to wireless network operation and management. It, however, is challenging, due to diverse feature types, diverse lengths of time slices, simultaneous occurrences of multiple root causes, and lack of training samples. In this paper, we present our solutions for these problems in ICASSP-SPGC-2022 AIOps Challenge in Communication Networks. We first design specific feature engineering method to represent the provided spatial features. Secondly, we conduct time series analysis on training data and propose an efficient method to infer whether a sample includes multiple root causes. Thirdly, in order to solve the lack of training data for unseen combinations of multiple root causes, we propose to ensemble multiple single-root-cause classifiers. Fourthly, we introduce TextCNN into multivariate time series classification to obtain high accuracy. Our approach achieved score 0.93 and ranked 4th place on the leader board. Code is available at https://github.com/zxuan000/SPGC_aiops_bjtu

***Index Terms*—** Root cause analysis, TextCNN, AIOps, Time series analysis

1. INTRODUCTION

In order to ensure mobile networks operate reliably, faults must be diagnosed and located accurately. To locate root causes of faults, one class of existing studies uses causal discovery algorithms to infer dependencies between time series [1,2]. Another class of algorithms uses probabilistic graphical models to represent the conditional correlations between variables [3, 4]. Few studies, however, are conducted in wireless networks.

Root cause analysis (RCA) of wireless network faults is challenging. Specifically, in this competition, we found the following four key challenges by carefully characterizing the provided data:

- **Diverse feature types.** The dataset contains nearly 90 features. In addition to numerical features, it also contains some spatial features that can not be directly used.
- **Diverse lengths of time slices.** The lengths of sample files vary from 1 to several hundreds. Many files are very short, which makes it difficult to obtain salient features of different root causes in the time dimension.
- **Simultaneous occurrence of multiple root causes.** When multiple root causes occur simultaneously, the distributions of data are affected. For example, when Root Cause 2&3 occur simultaneously, the distributions of data are significantly different from that with only Root Cause 2 or 3. As a result, the classifiers for these root causes degrade seriously.
- **Unseen combinations of multiple root causes.** In the training data, there are only four root cause combinations, i.e., Root Cause 1, 2, 3, 2&3. Thus, we do not have training data for Root Cause 1&3, 1&2, 1&2&3. These combinations, however, are possible to appear in practice. Hence, it is a challenge to learn them.

To handle the above problems, we first analyze features highly correlated to root causes. Secondly, based on our observation that, when multiple root causes occur simultaneously, the distributions of data are affected, we propose a heuristic method to infer whether a sample includes a single root cause or multiple ones. Thirdly, we explore various machine learning models [5–7], including Random Forest and TextCNN [8] to locate root causes. Finally, to solve the lack of training data for unseen combinations of multiple root causes, we ensemble multiple single-root-cause classifiers. Our contributions are summarized as follows:

- To handle the challenge of *diverse feature types*, we design specific feature engineering method to represent the provided spatial features.
- We design an ensemble method which combines three single-root-cause binary classifiers to successfully handle the challenges of *diverse lengths of time slices* and *unseen combinations of multiple root causes*.
- We design an efficient time series analysis module to

*This work was supported in part by the National Natural Science Foundation of China under Grant No. 61572071 and No. 61872031.
†Thanks to Beijing Baolande Software Corporation for funding.

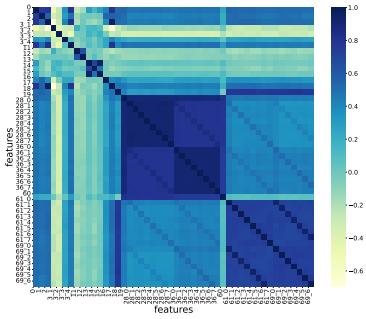

Fig. 1. Heatmap of correlation coefficient between features

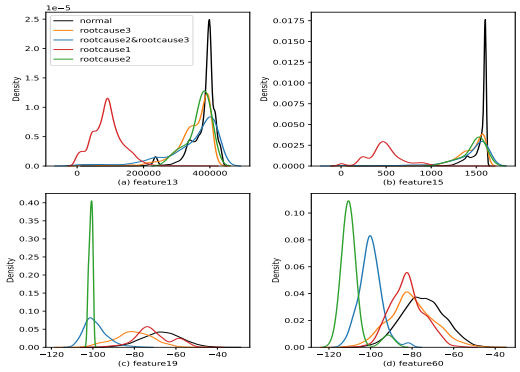

Fig. 2. Probability density distributions of four features under different root causes

infer multiple-root-cause case and successfully predict *simultaneous occurrences of multiple root causes*.

- We introduce TextCNN into root cause inference task, which is proven efficient to find the root cause from the time-series data.

We evaluated our method in competition data. The results showed that our method is efficient: it achieved score 0.93 and ranked 4th place on the leader board.

2. DATA ANALYSIS

2.1. Correlation Analysis

Fig 1 plots the correlation matrix of all features of a randomly-selected sample. As shown in Fig 1, firstly, some features are highly correlated. For example, features 13 and 15 are highly correlated. Secondly, such correlations are consistent with the causal relationship graph shown on the competition website[1]. For example, features 13 and 15 are directly connected to rootcause1. Thus, we identified them as key features and further observe them carefully.

2.2. Variance of Distributions of Features

In this subsection, we observe the distributions of features under different root causes. As an example, Fig 2 plots the distributions of four features, i.e., feature 13, 15, 19 and 60. First, as shown in Fig 2(a), the distribution of feature 13 with root cause 1 is considerably different from those of other root causes. Such a result means feature 13 is a key feature for root cause 1. Second, as shown in Fig 2(d), the distribution of feature 60 with root cause 2&3 is considerably different from those of root cause 2 and 3. Such a result means that, if feature 60 is selected as an important feature for classification of root cause 2 or 3, then, when root cause 2 and 3 occur simultaneously, the classifiers for root cause 2 or 3 may degrade.

[1]https://www.aiops.sribd.cn/home/data

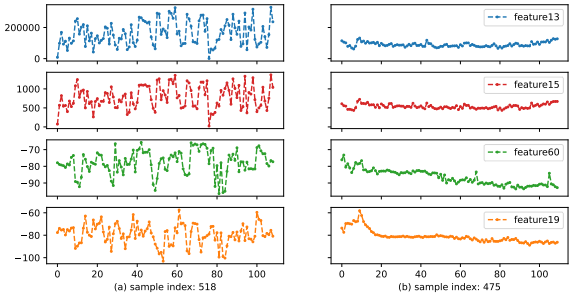

Fig. 3. Features in two time slice files (index 518 & 475)

2.3. Time Slice Analysis

The data set consists of files. Each file is a time slice. We first observe time slice length. We found that sample file lengths vary from 1 to several hundreds. Moreover, most file lengths are short and only a few files' lengths are long.

We second carefully observe the long files to obtain insights in their time-series characteristics. As an example, Fig 3 plots the time series of four features in two long files (i.e., sample index 518 & 475). As shown in Fig 3, although the two files have similar lengths, the features in Fig 3(a) fluctuate more obviously than Fig 3(b). Such a phenomenon gives us an important intuition, i.e., multiple root causes may occur simultaneously in sample 518, and we need to measure the fluctuations of features and consider them in the model.

3. METHODOLOGY

3.1. Framework

Fig 4 plots the framework of our solution. We first analyze the time series characteristics of a sample to infer whether there is a possibility of multiple root causes occurring in a sample. Based on the inference results, we then follow different classification strategies.

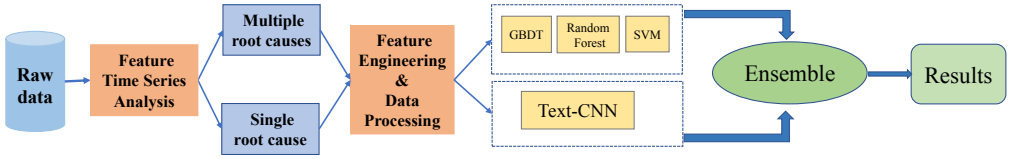

Fig. 4. Framework of our solution

Algorithm 1 Rroot cause location for time slice file

Input: Time slice file $data$
Output: Root cause of $data$

```
Multi_RC = simultaneous_multiple_root_causes(data)
if Multi_RC then
    for each row in data do
        feature_engineering(row)
        RC_classification()
    end for
    ensemble_classification_result()
else
    feature_engineering(data)
    RC_classification()
end if
```

Algorithm 1 shows pseudo-codes of our classification strategies. As shown in Algorithm 1, if we infer multiple root causes occur in a sample, we will conduct root cause classification on each row of the data and then ensemble the classification results of all rows to obtain the final root causes. For comparison, if we infer that only a single root cause occurs in a sample, we will conduct root cause classification on the total file to directly obtain the root cause.

3.2. Infer Multiple Root Causes Occurrence

Based on the observations shown in Fig 3, we propose a method to infer simultaneous occurrences of multiple root causes in a sample. Specifically, we first select features which we found are highly correlated to the root cause according to the causal relationship graph provided by the competition. Then, we evaluate their variances in a sample file. If they are large, we infer that this file has multiple root causes occurring. Due to the limitation of paper size, we skip the introduction of all parameter settings. Please refer to our code for the parameters.

3.3. Feature Engineering and Data Processing

3.3.1. Design of Spatial Features

Fig 5(a) plots the spatial distribution of the feature values. As shown in Fig 5(a), feature20_n represents the ID of eight receiving directions and the direction is represented by n. Their values are non-negative integers from 0 to 31, and arranged as a 4×8 matrix. To utilize the spatial features, we design the following two features:

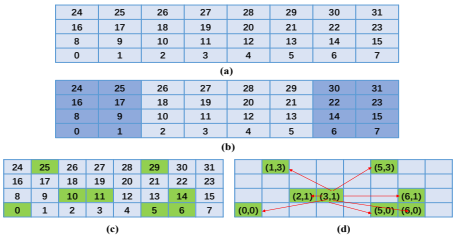

Fig. 5. Design of spatial features

feature_edge: Since rootcause2 refers to *weak signal in marginal areas*, we design the *feature_edge* to represent this characteristic. We first define the left two columns and the right two columns of the feature20 matrix as edge columns, which are marked in dark blue in Fig 5(b). We then count the number of eight direction values of feature20 hitting the edge columns at each timestamp to form the *feature_edge*.

feature_distance: Rootcause3 refers to *strong interference among nodes*. As the distance among the receiving directions reflects the degree of interference between signals, we design *feature_distance*. We first place the matrix in a rectangular coordinate system and convert the direction ID into coordinates. We then calculate the sum of Euclidean distance between each two nodes of the eight directions of feature20 at each timestamp to form the *feature_distance*. For example, receiving directions are [0,5,6,10,11,14,25,29] (Fig 5(c)), and Fig 5(d) is a schematic diagram of ID_{11} calculating the distance between other nodes.

3.3.2. Design of feature_length

The data lengths of sample files vary greatly. As the length reflects the duration of different root causes, we count the lengths of files as *feature length*.

3.3.3. Other Processing of Features

We also conduct the following data pre-processing. 1) ***Z-score Normalization:*** According to our observation, there is

a large difference in the data distribution between the training and test samples, so the z-score normalization needs to be performed. 2) ***Data Augmentation:*** We found that the given labels were extremely unbalanced, and thus used Borderline SMOTE [9] algorithm for data augmentation.

3.4. Classifiers

3.4.1. TextCNN-Based Classifier

We use a modified TextCNN [8] to capture the possible causal relationships between dynamic features. Given a time slice consisting of dozens of feature sequences, we need to infer the root cause not only from the features themselves but also from the causal relationships between the features. Thus, we use TextCNN to obtain a representation of dynamic features with causal relationships between features. Then, given a causal graph, we can take a feature path, i.e., the path from start feature node to end feature node in the graph, as a sentence and the time series of a feature as a feature vector. Fig 6 plots our model structure and its details are as follows.

Padding and Clipping: Since the length of word vector in TextCNN is fixed, we firstly pad and clip the feature values of each file. By observing file lengths, we found that 75% quartile of the lengths of test files is around 30. Thus, for files longer than 30, we clip the data. For those shorter than 30, we use the mean value to fill features. All columns are then filled by linear interpolation of adjacent two files. Finally, each file forms a matrix of $[number\ of\ feature \times 30]$ as the input.

Attention layer: To classify root causes, different features may have various levels of importance. So an attention layer is introduced to calculate feature weights dynamically. The output of the attention layer is calculated as:

$$e_n = W_n^T \cdot f_n \tag{1}$$

$$\alpha_n = \frac{exp(e_n)}{\sum_{j=0}^{nums} exp(e_j)} \tag{2}$$

$$h_n = \sum_{i=1}^{d} \alpha_n \cdot f_n^i \tag{3}$$

Where W_n^T is a trainable parameter, f_n denotes the vector of features,α_n is the attention weight of each feature, and h_n is the output of the attention layer.

Convolution layer: After the attention layer, we use CNN to capture causal relationships between features. We place neighbor features in the causal graph close so that the relationship between them can be captured during convolution. The size of convolution kernel is an important parameter for TextCNN to capture correlation between features. As shown in the causal graph, the degrees of nodes are mostly 3, 4, 5. Thus, we set the size of convolution kernel to 3, 4, 5.

After the pooling layer, we use a fully connected layer and a softmax layer for classification and output the probabilities for each category.

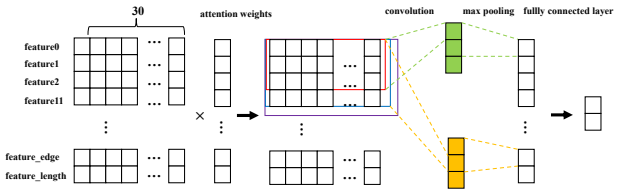

Fig. 6. Model of Textcnn with attention

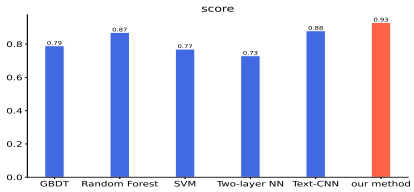

Fig. 7. Score of different methods

3.4.2. Ensemble of Machine Learning Classifiers

Besides TextCNN, we also systematically evaluate other classical machine learning models, including GBDT [5], SVM [6], Random Forest [7]. Fig 7 plots the evaluation results of different algorithms on the test dataset. As the ground truth of the result on the test dataset is unavailable, we just report the score of the competition platform. As shown in Fig 7, first, Random Forest and TextCNN outperforms other methods. Second, the correlation between the prediction results of Random Forest and TextCNN is quite low, suggesting the two models can identify different root causes within the data.

Refer to One-vs-Rest strategy which splits a multi-class classification into one binary classification problem per class, we use three different single-root-cause binary classifiers. Based on the experimental result, we finally use Random Forest for root cause 2 and 3 and TextCNN for root cause 1. Our total solution finally achieved score 0.93 and ranked 4th place on the leader board.

4. CONCLUSION

In this paper, we present our solution for ICASSPSPGC-2022 AIOps Challenge. We first conduct detailed time series analysis and identify four key challenges in the competition data, and then design specific spatial features, propose an efficient method to infer whether a sample includes multiple root causes, introduce TextCNN, and propose an ensemble method to solve the problem of unseen combinations of multiple root causes. Experimental results showed that our approach is efficient: it achieved score 0.93 and ranked 4th place on the leader board.

5. REFERENCES

[1] Jakob Runge, Peer Nowack, Marlene Kretschmer, Seth Flaxman, and Dino Sejdinovic, "Detecting and quantifying causal associations in large nonlinear time series datasets," *Science Advances*, vol. 5, no. 11, 2019.

[2] Thuc Duy Le, Tao Hoang, Jiuyong Li, Lin Liu, Huawen Liu, and Shu Hu, "A fast pc algorithm for high dimensional causal discovery with multi-core pcs," *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 16, no. 5, pp. 1483–1495, 2016.

[3] Anna Lokrantz, Emil Gustavsson, and Mats Jirstrand, "Root cause analysis of failures and quality deviations in manufacturing using machine learning," *Procedia CIRP*, vol. 72, pp. 1057–1062, 2018.

[4] Jiajin He and Hui Zhao, "Fault diagnosis and location based on graph neural network in telecom networks," in *2020 International Conference on Networking and Network Applications (NaNA)*. IEEE, 2020, pp. 304–309.

[5] Jerome H Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[6] William S Noble, "What is a support vector machine?," *Nature biotechnology*, vol. 24, no. 12, pp. 1565–1567, 2006.

[7] Andy Liaw, Matthew Wiener, et al., "Classification and regression by randomforest," *R news*, vol. 2, no. 3, pp. 18–22, 2002.

[8] Ibrahim Alshubaily, "Textcnn with attention for text classification," *arXiv preprint arXiv:2108.01921*, 2021.

[9] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao, "Borderline-smote: a new over-sampling method in imbalanced data sets learning," in *International conference on intelligent computing*. Springer, 2005, pp. 878–887.