Journal of Advances in Mathematics and Computer Science

24(1): 1-8, 2017; Article no.JAMCS.35796

Previously known as British Journal of Mathematics & Computer Science

ISSN: 2231-0851

Log Prediction of Wireless Telecommunication Systems Based on a Sequence-To-Sequence Model

Weiliang Ji[1], Renai Chen[1], Feng Li[1] and Qiang Ling[1*]

[1]*Department of Automation, University of Science and Technology of China, Hefei 230027, China.*

Authors' contributions

This work was carried out in collaboration by all authors. Author WJ designed the study, implemented the algorithm, and wrote the first draft of the manuscript. Author RC proposed the concerned problem, and finished the final manuscript. Authors FL and QL optimized the method and guided the experiments. All authors read and approved the final manuscript.

Article Information

DOI: 10.9734/JAMCS/2017/35796

Editor(s):

(1) Junjie Chen, Department of Electrical Engineering, University of Texas at Arlington, USA.

Reviewers:

(1) Anand Nayyar, KCL Institute of Management and Technology, India.

(2) S. V. Saboji, Basaveshwar Engineering College, India.

Complete Peer review History: http://www.sciencedomain.org/review-history/20635

Received: 29th July 2017

Accepted: 15th August 2017

Published: 23rd August 2017

Original Research Article

Abstract

Nowadays people are becoming increasingly dependent on wireless networks. Taking precautions and acting in advance to avoid problems of wireless networks have already shown great importance. In analyzing problems of wireless telecommunication systems, current methods mainly rely on structured data, like alarm data. Compared with structured data, log data are more abundant and recently implemented to detect problems of wireless telecommunication systems. In order to predict future problems, it becomes essential to predict future log data based on current log data. In the paper, we propose a novel method to predict log data of the wireless telecommunication system based on a sequence-to-sequence model. We use current logs as input, and generate future log with our model. In addition we discuss the effects of different parameters of our model through some experimental results.

*Corresponding author: E-mail: qling@ustc.edu.cn;

Keywords: *Log prediction; sequence-to-sequence; machine learning; wireless telecommunication systems; machine log; sliding window.*

2010 Mathematics Subject Classification: 53C25, 83C05, 57N16.

1 Introduction

With the rapid development of the social economy, wireless networks provide us with more convenient, faster and better-quality services. Meanwhile people are becoming increasingly dependent on these networks. Thus how to manage the wireless networks to make them function properly has been a tremendous challenge. More importantly, we prefer to take precautions and act in advance instead of handling problems afterwards since failures can hardly be avoided completely.

Wireless networks generate a large amount of data when running and maintaining the systems, including alarm data(ALM), call history record(CHR), key performance indicators(KPI) and logs [1], [2], [3], [4]. Among these generated data, ALM, CHR and KPI are generally utilized. Compared to others, logs are seldom analyzed thoroughly. In fact, since all network elements are outputting log data all the time, the quantity of log data is significantly larger than several other data. However, because logs are unstructured, they are hard to be analyzed automatically and heavily rely on manual processing. As a result, there is little analysis for logs, which haven't been fully taken advantage of.

Sequence-to-sequence(seq2seq) model, which is based on RNN, takes one sentence as its input and outputs another sentence [5], [6]. Nowadays seq2seq is widely used in natural language processing (NLP) field, such as machine translation [6], automatic question answering [7], [8] and text summarization [9]. Recently it has been used to do some amazing work. For example, in [10], Chinese poetry was generated base on a seq2seq model.

Therefor inspired by NLP techniques, we propose a seq2seq based method to predict future logs. The prediction model takes current logs as inputs and predict the future logs. The remainder of this paper is arranged as follows: Section 2 introduces our data setup, Section 3 describes how seq2seq works to predict logs and Section 4 shows our experiments and results, and at last Section 5 is a summary of our contributions.

2 Data

Our log data is from a simulated wireless telecommunication system, and different lines of log doesn't have same structure. Three examples are shown as below:

```
[2.02102257],mme_diameter=>hss_diameter, Send MME_DIM_DIM_AUTH_DATA_REQ
Debug:[2.03101127] [top.MME4.mme_nas] [init.attach] T3460 stop for ue<900000>!
T3460 start for ue<900000>!
```

We can see there exists great differences between three lines of log entries, which are obviously unstructured. Thus in order to use seq2seq model to predict, we must preprocess our data first. Preprocessing includes two steps: data cleaning in Subsection 2.1 and dictionary building in Subsection 2.2.

2.1 Data cleaning

As is shown, logs are unstructured, and have a lot of non-alphanumeric signs and numbers. It is hard to predict directly using raw log data. And what's more, most non-alphanumeric signs and numbers don't imply much important meaning. So we need to take off non-alphanumeric signs and numbers to clean the data.

However at the same time, we find _ in some words like *MME_DIM_DIM_AUTH_DATA_REQ* works as a connection. So *MME_DIM_DIM_AUTH_DATA_REQ* is supposed to be a single word, and _ is of vital importance which should be reserved. So we finally replace numbers and other non-alphanumeric sign with space to keep letters and _. After washing the raw data, every line of logs becomes words and is separated by space.

After data cleaning, three sentences mentioned above are turned to:

mme_diameter hss_diameter Send MME_DIM_DIM_AUTH_DATA_REQ
Debug top MME mme_nas init attach T stop for ue
T start for ue

2.2 Dictionary building

When every line becomes a sequence of words, we can then get the frequency of each word. According to word frequency, we can set a threshold manually. When the frequency of one word is above the threshold, we keep the word. Otherwise we replace the word with *_UNK*. So the final dictionary includes words whose frequency is above threshold and *_UNK*.

3 Model

Our model is based on seq2seq model with an attention mechanism [11]. We take several log entries as input(we call input sequence), and then can get future log as output(we call output sequence). Here, we discuss how to obtain the input and output sequences, and how to use model to generate output sequence which is just the predicted log.

3.1 Input and output sequence

As is shown in Fig. 1. We use two sliding windows to get input sequences and output sequences. Let l_{win} denote the length of sliding windows, *gap* denote the gap between two windows, and *step* denote the step size every time sliding windows move forward. Then every sliding window contains l_{win} lines of log entries. Lines in one sliding window are joint together by space. Then we can get a long sequence from each window. These sequences are regarded as input sequences and output sequences.

It should be noted that, even l_{win} is fixed, the length of sequence in different sliding window may be different because the number of words in different lines is different. Besides, *gap* also represents the elapsed time between current logs and logs to be generated in the future, so *gap* indicates our model's ability to predict the future.

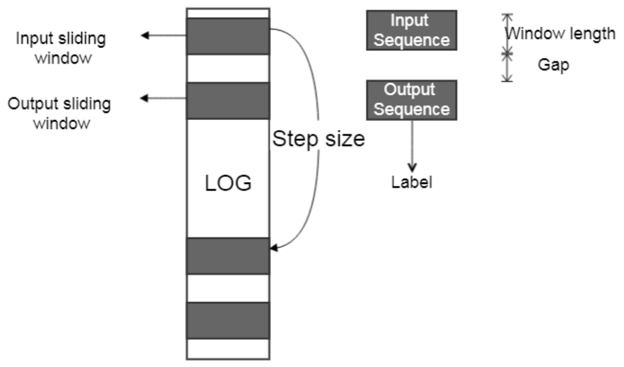

Fig. 1. Obtain input and output sequence with sliding windows

3.2 Model building

A typical structure of a seq2seq model with an attention mechanism is shown in Fig. 2. The model takes a sentence ABC as input one word at a time, and gets an output sentence WXYZ, also one word a time. <EOS> is a placeholder which means the end of a sentence.

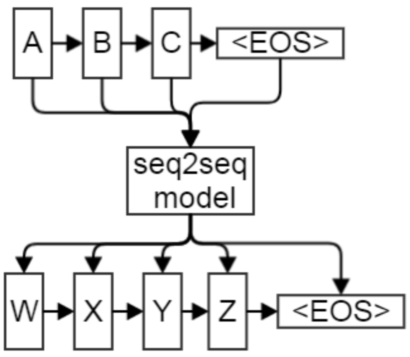

Fig. 2. Using of seq2seq model

In our case, *'ABC'*, corresponding to current logs means an input sequence from the input sliding window described in Subsection 3.1, and *'WXYZ'* corresponding to logs in the future, is an output sequence. During the training procedure, input and output sequences are both fed to our model. The model is train to maximize the cross entropy of the actual output sequence. When the model is well trained, we can use current logs to predict future logs.

3.3 Evaluation

In order to observe the effectiveness of our model, WER which is a common metric in speech recognition and machine translation, is chosen to be the evaluating criterion. WER is derived from the Levenshtein distance and works at the word level. Given a generated log and the true log, this problem is solved by first aligning the generated log with the true log using dynamic string alignment, and WER can then be computed as:

$$WER = \frac{S+D+I}{N} = \frac{S+D+I}{N} \tag{3.1}$$

S is the number of substitutions, D is the number of deletions, I is the number of insertions, C is the number of the corrects, N is the number of words in the true log($N = S + D + C$).

It should be noted that we don't choose other metric which are usually used in machine translation like bilingual evaluation understudy(BLEU) [12], because BLEU involves a lot human behavior information. However, our log data is generated by machines instead of human beings, so BLEU isn't suitable in our case.

4 Experiments and Results

We have 1GB log data in our experiments. We take 70% as training set to train our model, with the rest 30% as test set to judge the performance of our model.

In order to leverage the sequential information in the input sequences, we choose to set a relatively large l_{win}. However, it takes so much time for seq2seq to process such a long input sequence. Thus in order to reduce the computational cost, we introduce a sample strategy to condense these input and output sequences before they are fed into the model. To maintain the integrity of every log entry, sample the original logs by lines. The sampling rate is set to 0.5. Specifically, we select one line for every two lines of logs. Besides we will show that sampling doesn't decrease the performance of our model.

4.1 An example

We set l_{win} to 30, *gap* to 2000 and *step* to 10. Then we get our input and output sequences. After splitting the data into training set and test set, we can train our model. When the training procedure is finished, we choose an input sequence from current logs. An example is shown as below:

enb_s ap mme_s ap Send ENB_S AP_S AP_UL_NAS_TRANS enb_s ap mme_s ap Send ENB_S AP_S AP_UL_NAS_TRANS enb_s ap mme_s ap Send ENB_S AP_S AP_UL_NAS_TRANS enb_s ap mme_s ap Send ENB_S AP_S AP_UL_NAS_TRANS enb_s ap mme_s ap
Send ENB_S AP_S AP_UL_NAS_TRANS enb_s ap mme_s ap Send ENB_S AP_S AP_UL_NAS_TRANS enb_s ap mme_s ap Send ENB_S AP_S AP_UL_NAS_TRANS enb_s ap mme_s ap Send ENB_S AP_S AP_UL_NAS_TRANS enb_s ap mme_s ap Recv ENB_S AP_S AP_UL_NAS_TRANS enb_s ap mme_s ap Recv ENB_S AP_S AP_UL_NAS_TRANS ue_nas mme_nas Recv UE_NAS_NAS_AUTH_RSP ue_nas mme_nas Recv UE_NAS_NAS_AUTH_RSP mme_nas ue_nas Send MME_NAS_NAS_SECU_MODE _CMD mme_nas ue_nas Send MME_NAS_NAS_SECU_MODE_CMD Debug top MME mme_nas init attach T stop for ue

Then we can get an output sequence as predicted future logs:

enb_s ap mme_s ap Recv ENB_S AP_S AP_UL_NAS_TRANS enb_s ap mme_s ap Recv ENB_S AP_S AP_UL_NAS_TRANS enb_s ap mme_s ap Recv ENB_S AP_S AP_UL_NAS_TRANS ue_nas mme_nas Send ue_nas mme_nas Send UE_NAS_NAS_ATTACH_CMP ue_nas mme_nas
Send UE_NAS_NAS_ATTACH_CMP ue_nas mme_nas Send UE_NAS_NAS_ATTACH_CMP ue_nas ue_rrc Send UE_NAS_RRC_DETACH_NOTIFY ue_nas mme_nas Send UE_NAS_NAS_ATTACH_CMP ue_nas ue_rrc Send UE_NAS_RRC_DETACH_NOTIFY ue_nas ue_rrc Send UE_NAS_RRC_DETACH _NOTIFY ue_nas ue_rrc Send UE_NAS_RRC_DETACH_NOTIFY mme_nas ue_nas

Recv MME_NAS_NAS_DETACH_ACPT mme_nas ue_nas Recv MME_NAS_NAS_DETACH_ACPT Debug top UE_ ue_nas detach Stop T for UE Debug top UE_ ue_nas detach UE is UE_EMM_STATE_DEREGISTERED mme_nas ue_nas Recv MME_NAS_NAS_DETACH_ACPT

4.2 Parameter analysis

In order to known evaluate our method, we take more experiments using our log data with different parameters.

At first, we test different l_{win}. As is shown in Fig. 3, we set l_{win} to 30 and 50. What's more, we also test when l_{win} is 30 without sampling. From the figure, we can know that the bigger the l_{win} is, the lower the WER is, which implies that model is better. It's easy to understand that, when l_{win} is big, the input sequence contains more information. However, when l_{win} is too big that model can hardly manage, the performance may become worse. And when l_{win} is 30, we can see there is no significant change after sampling. Hence it's feasible to sampling logs.

Then, set different *gap*. As shown in Fig. 4, for most epochs WER is the lowest when *gap* is set to 1000, and highest when *gap* is set to 5000. For all three curves, WER decreases when epoch increase. It's obviously that when *gap*, which implies the time between number of lines between generated log in the future and current log, is big, it's more likely for systems to change the current state, so it's hard for our model to predict. As a result, the potential for error with big *gap* can be large.

Finally, we test different *step*. When *step* is small, we can get more samples. As is shown in Fig. 5, when epoch is small, three curves are close. However when epoch increases, the performance when *step* is 100 becomes worse than the performance when *step* is 10 and 50. When l_{win} is 50, we can take use of all information in logs when *step* is 10 or 50. The difference is that when *step* is 10, there is much duplicated information in input and output sequences, which doesn't matter in the case of a large dataset. However, when *step* is 100, bigger than l_{win} 50, we will lose lots of information when we move sliding windows. Thus the result can be worse.

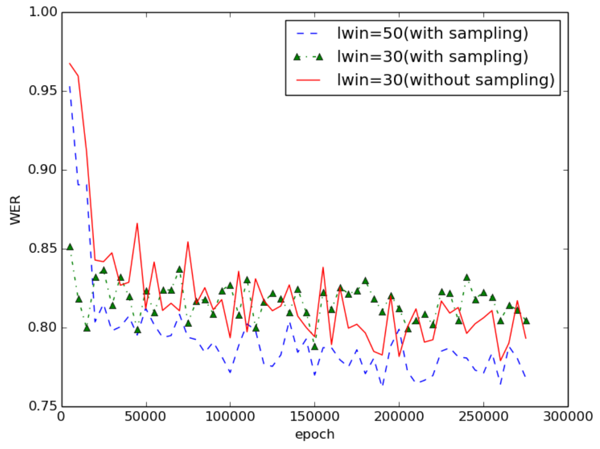

Fig. 3. WER with different window length

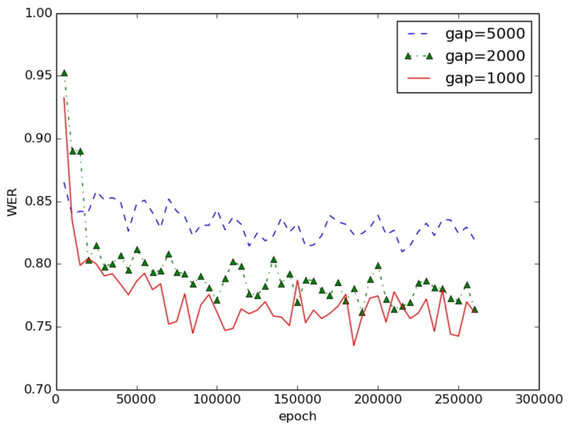

Fig. 4. WER with different gap

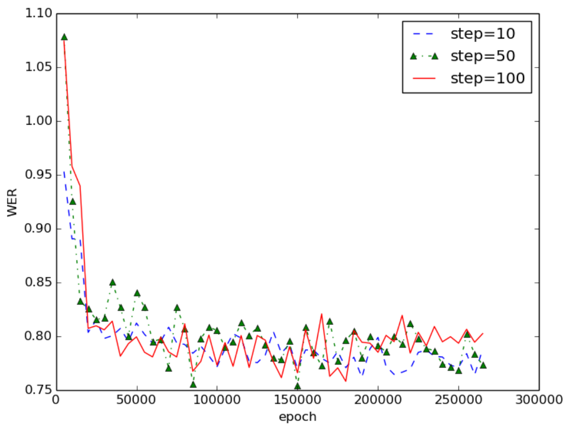

Fig. 5. WER with different step

5 Conclusion

In this paper, we propose a novel method to generate log in the wireless telecommunication system. Experiments shows that by using our model, we have the ability to use current logs to predict logs in the future. And what's more, we analyze three key parameters of our model, and show how they work. However, our results aren't ideal enough, there is still work to do, for example to modify the model so we can predict after a longer period of time. The input and output sequences are still too long for seq2seq model which remains to be solved later. What's more, the ability to predict future logs allows us to do more analyses on wireless telecommunication system. We are able to analyze the state of the system in the future based on future logs and predict the failure in advance, which is also expected in the further studying.

Acknowledgement

This work was supported by the National Key Research and Development Program of China (No. 2016YFC0201003).

Competing Interests

Authors have declared that no competing interests exist.

References

[1] Weiss G. Predicting telecommunication equipment failures from sequences of network alarms[J]. Handbook of Knowledge Discovery and Data Mining. 2002;891-896.

[2] Sundsy P, Bjelland J, Reme B, et al. Deep learning applied to mobile phone data for Individual income classification[C]. 2016 International Conference on Artificial Intelligence: Technologies and Applications. Atlantis Press; 2016.

[3] Barco R, Wille V, Dez L. System for automated diagnosis in cellular networks based on performance indicators[J]. Transactions on Emerging Telecommunications Technologies. 2005;16(5):399-409.

[4] Zheng Z, Lan Z, Park B H, et al. System log pre-processing to improve failure prediction[C]. Dependable Systems & Networks, 2009. DSN'09. IEEE/IFIP International Conference on. IEEE. 2009;572-577.

[5] Cho K, Van Merrinboer B, Gulcehre C, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation[J]. arXiv preprint arXiv. 2014;1406.1078.

[6] Sutskever I, Vinyals O, Le Q V. Sequence to sequence learning with neural networks[C]. Advances in neural information processing systems. 2014;3104-3112.

[7] Yin J, Jiang X, Lu Z, et al. Neural generative question answering[J]. arXiv preprint arXiv. 2014;1512.01337.

[8] Vinyals O, Le Q. A neural conversational model[J]. arXiv preprint arXiv. 2015;1506.05869.

[9] Nallapati R, Zhou B, Gulcehre C, et al. Abstractive text summarization using sequence-to-sequence rnns and beyond[J]. arXiv preprint arXiv. 2016;1602.06023.

[10] Wang Z, He W, Wu H, et al. Chinese poetry generation with planning based neural network[J]. arXiv preprint arXiv. 2016;1610.09889.

[11] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate[J]. arXiv preprint arXiv. 2014;1409.0473.

[12] Papineni K, Roukos S, Ward T, et al. BLEU: A method for automatic evaluation of machine translation[C]. Proceedings of the 40th annual meeting on association for computational linguistics. Association for Computational Linguistics. 2002;311-318.