On the Adoption of Neural Networks in Modeling Software Reliability

Kamill Gusmanov
Innopolis University, Russian Federation

ABSTRACT

This work models the reliability of software systems using recurrent neural networks with long short-term memory (LSTM) units and truncated backpropagation algorithm, and encoder-decoder LSTM architecture and proposes LSTM with software reliability functions as activation functions and LSTM with input features as the output of software reliability functions. An initial evaluation on data coming from 4 industrial projects is also provided.

CCS CONCEPTS

• **Software and its engineering → Software reliability**;

KEYWORDS

Software reliability modelling, feedforward neural networks, recurrent neural networks, long short-term memory, encoder-decoder architecture

ACM Reference Format:
Kamill Gusmanov. 2018. On the Adoption of Neural Networks in Modeling Software Reliability. In *Proceedings of the 26th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '18), November 4–9, 2018, Lake Buena Vista, FL, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3236024.3275433

1 THE RESEARCH PROBLEM AND MOTIVATION

According to Musa et al. (1987) [18], software reliability is the probability of failure-free operation of a computer program for a specified time in a specified environment. Software reliability is directly connected to software failures because if a software is not correctly functioning there is the assumption that a software failure has occurred [13]. Typically, reliability of software is measured with the number of defects that exist in the source code of the released software or with failures that happen during its execution [37]. Indeed, being able to model and predict the reliability of software systems is provides the ground for better more effective management of the overall development process [10, 17, 23, 32].

The most common approaches of the software reliability engineering can be classified as: (a) software reliability growth models, (b) multiple linear regression models, (c) Bayesian models, and (d) neural network models.

ESEC/FSE '18, November 4–9, 2018, Lake Buena Vista, FL, USA

ACM ISBN 978-1-4503-5573-5/18/11.
https://doi.org/10.1145/3236024.3275433

This study is based on the application of recurrent neural networks in software reliability modeling. It is due to their ability to predict future results based on the previous "knowledge" and to capture unforeseen relationships in the data.

2 BACKGROUND AND RELATED WORK

Neural networks have been already used to model and predict the reliability of software systems but seldom in the case of multireleases. Their performances are typically measured by Root Mean Square Error, Mean Absolute Error, and Mean Squared Error values.

Feedforward neural networks [1–4, 9, 15, 19–21, 24–27, 30, 34, 40] are the most common. The main differences between the different approaches lie in the number of hidden layers and of neurons in the input and in the hidden layer, the activation functions, and the training algorithm. The most applicable activation functions are classical functions but there are also works proposing Software Reliability Growth Model functions as activation functions, like Goel-Okumoto [9, 25, 30, 31, 34], Yamada Delayed S-Shaped [25, 30, 31, 34], Logistic growth curve [25, 30, 31], Inflection S-shaped [25], Subburaj-Gopal Generalized NHPP [9], or Schneidewind model [34] functions. The performances of the proposed models with different evaluation criteria in most cases are better than traditional reliability models.

Generalized regression neural networks [38] are an extended form of probabilistic neural network capable of approximating the mapping relationship implied in the sample data [29]. The input vector represents sets of two items - failure time and test coverage values. The output vector represents a set of the cumulative number of faults at the given time. In the case, when the test coverage is incorporated, proposed model shows 0.1708 of Mean Squared Error (MSE) value, while the backpropagation neural network shows 68.8266 of MSE value.

Convolutional neural networks [14] include an embedding layer, a convolutional layer, a max-pooling layer, a fully-connected hidden layer and a single unit output layer with a logistic regression classifier. The input vector is represented as a token vector of representative nodes of the Abstract Syntax Trees (ASTs) of the source java files. The proposed convolutional neural network shows an F1 score of 0.608, which is better and higher than simple CNN model and Deep-Belief neural network model.

Recurrent neural network [8, 11, 25, 28, 35, 36, 39]. Wang et al. (2017) [35] and Yangzhen et al. (2017) [39] used Long Short-Term Memory Neural Network models using the former time-series data of records of historical reliability as input and expected reliability as output, and the latter a series of time between failures as input and the next time between failure as output. Roy et al. (2014) [25] proposed to use recurrent neural network based dynamic weighted combination model using as activation functions various SRGM, such as Goel-Okumoto, Yamada Delayed S-shaped, Inflection S-shaped, and Logistic. Wang et.al (2018) [36] adopts deep neural network model based on the recurrent neural network encoder-decoder architecture.

3 APPROACH AND UNIQUENESS

Neural networks have shown an impressive modeling and predictive power in several domains, such as stock market index prediction [16], object detection [22], image classification [12], speech recognition [7], and many others.

However, for reliability predictions of the performances are not as good as one might expect. Our work tries to analyse why and to develop a more performant model.

Since software reliability data is a time-series, we use recurrent neural networks, the most effective neural network to handle such non-stationary data [33]. However, simple recurrent neural networks are incapable of handling "long-term dependencies," since gradients tend to either vanish or explode [6]. Long Short-Term Memory networks do not have this problem and they allow to accumulate information over a long duration and they can decide what information they should forget or remember in the next stages.

As was mentioned earlier, Wang et al. (2017) [35] and Yangzhen et al. (2017) [39] proposed applications of Long Short-Term Memory neural network models. The first paper is based on the simple LSTM network without any changes in the structure of the cell. Depending on the prediction period, the proposed model shows values lower than 0.05 to 0.15 of the Root Mean Squared Error (RMSE). The second paper used the same structure and addtionally applied truncated backpropagation and layer normalization for improving the performance of the model.

On such bases, we organize our research in four stages: **(1)** application of the LSTM with truncated backpropagation, **(2)** application of the encoder-decoder RNN architecture, **(3)** application of the software reliability model functions as activation functions in the LSTM cell, **(4)** application of output features of software reliability functions as input features for LSTM model.

The first two stages are our attempts to repeat the discussed RNN models with multirelease data. The last two steps are the main steps where two good solutions of software reliability modelling are combined – traditional functions and LSTM model. All models are evaluated on data coming from industry.

4 PRELIMINARY RESULTS

To understand the applicability of our model, we have already performed an initial validation of the LSTM model with truncated backpropagation on four industrial datasets coming from four products of a major Russian company, whose identity is anonymous for obvious confidentiality issues. They are referred to as "P-1", "P-2", "P-3", "P-4". The datasets represent defect occurring in the multireleased software systems. At the beginning, Root Mean Squared Error (RMSE) was used as main evaluation metric. In the subsequent works, the proposed methods will be additionally evaluated using Relative Error (RE) and Average Relative Error (AE).

As training data 20% and 60% of each datasets were used. By applying sliding window technique, training sets were represented in the following form: $(y_{i-r}, y_{i-r+1}, ..., y_{i-1}, y_i)$, where r is the length of the sliding window, y_i represents the cumulative number of failures at the current time, and i is the order index. In our case, the length of the sliding window is equal to 4 and 6.

Table 1: Results of application of the proposed model.

Dataset	Root Mean Squared Error				Number of failures in the dataset
	length of 4		length of 6		
	20%	60%	20%	60%	
P-1	27	27	29	7	1579
P-2	77	25	241	25	2458
P-3	17	2	14	2	402
P-4	9	13	19	12	571

We ran the proposed model 5 times on the datasets with different values of hyperparameters. The average and rounded results of RMSE values are shown in Table 1. In most of the cases, results are higher in the 20% dataset than in the 60% one. It means that this model is not able to make adequate predictions on the small amount of data. But these results are from the simple LSTM model without any significant changes in the internal structure, unlike the next stages.

5 FUTURE WORK

The first and the simple stage has not shown good results in prediction abilities.

The second stage is applying encoder-decoder RNN architecture. This model was proposed by Cho et al. (2014) [5] and applied to the neural machine translation using Gated Recurrent Unit (GRU), which is a simpler recurrent neural network unit and has fewer parameters than LSTM. But we are going to start with LSTM units.

The third stage is applying software reliability model activation functions as activation functions of the LSTM cells. The introduced feedforward and recurrent neural networks with these kinds of functions have shown good results on time-series data for modeling software reliability.

The fourth stage is applying the output feature vector of the software reliability functions as the input for the LSTM model. The combination of these functions and the proposed neural network model, theoretically, will give better results on most of the above mentioned datasets.

6 CONCLUSION

This paper proposed a software reliability modeling using neural network approach, in particular - Long-Short Term Memory model. Truncated backpropagation was added to the model for increasing its performance. Results show, that this model has high values of the RMSE on different datasets, which is not good. Next steps will improve the neural network model by applying encoder-decoder architecture and using software reliability model functions in LSTM architecture.

REFERENCES

[1] WA Adnan, M Yaakob, R Anas, and MR Tamjis. 2000. Artificial neural network for software reliability assessment. In *TENCON 2000. Proceedings*, Vol. 3. IEEE, 446–451.

[2] Sultan H Aljahdali, Alaa Sheta, and David Rine. 2001. Prediction of software reliability: A comparison between regression and neural network non-parametric models. In *Computer Systems and Applications, ACS/IEEE International Conference on. 2001.* IEEE, 470–473.

[3] Kai-Yuan Cai, Lin Cai, Wei-Dong Wang, Zhou-Yi Yu, and David Zhang. 2001. On the neural network approach in software reliability modeling. *Journal of Systems and Software* 58, 1 (2001), 47–62.

[4] Yung-Chung Chen and Xiao-Wei Wang. 2009. Neueural-Network-based approach on reliability prediction of software in the maintenance phase. In *Industrial Engineering and Engineering Management, 2009. IEEM 2009. IEEE International Conference on.* IEEE, 257–261.

[5] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning.* MIT Press. http://www.deeplearningbook.org.

[7] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on.* IEEE, 6645–6649.

[8] SL Ho, M Xie, and TN Goh. 2003. A study of the connectionist models for software reliability prediction. *Computers & Mathematics with Applications* 46, 7 (2003), 1037–1045.

[9] L Indhurani and R Subburaj. 2015. An artificial neural network approach to software reliability growth modelling. *Procedia Computer Science* 57 (2015), 695–702.

[10] Vladimir Ivanov, Manuel Mazzara, Witold Pedrycz, Alberto Sillitti, and Giancarlo Succi. 2016. Assessing the Process of an Eastern European Software SME Using Systemic Analysis, GQM, and Reliability Growth Models: A Case Study. In *Proceedings of the 38th International Conference on Software Engineering Companion (ICSE '16). Austin, TX, USA, May 14 - 22, 2016.* ACM, New York, NY, USA, 251–259. https://doi.org/10.1145/2889160.2889250

[11] Alireza Jomeiri. 2010. Software fault detection for reliability using recurrent neural network modeling. In *Software Technology and Engineering (ICSTE), 2010 2nd International Conference on*, Vol. 2. IEEE, V2–149.

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems.* 1097–1105.

[13] Anurag Kumar. 2016. Software Reliability Growth Models, Tools and Data Sets-A Review. In *Proceedings of the 9th India Software Engineering Conference.* ACM, 80–88.

[14] Jian Li, Pinjia He, Jieming Zhu, and Michael R Lyu. 2017. Software defect prediction via convolutional neural network. In *Software Quality, Reliability and Security (QRS), 2017 IEEE International Conference on.* IEEE, 318–328.

[15] I Mandal. 2010. Software reliability assessment using artificial neural network. In *Proceedings of the International Conference and Workshop on Emerging Trends in Technology.* ACM, 698–699.

[16] Amin Hedayati Moghaddam, Moein Hedayati Moghaddam, and Morteza Esfandyari. 2016. Stock market index prediction using artificial neural network. *Journal of Economics, Finance and Administrative Science* 21, 41 (2016), 89–93.

[17] Raimund Moser, Witold Pedrycz, and Giancarlo Succi. 2008. Analysis of the reliability of a subset of change metrics for defect prediction. In *Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '08).* ACM, 309–311. https://doi.org/10.1145/1414004.1414063

[18] John D Musa, Anthony Iannino, and Kazuhira Okumoto. 1987. Software Reliability: Measurement, Prediction, Application. 1987.

[19] Peng Nie, Ji Geng, and Zhiguang Qin. 2011. A Component-Oriented Reliability Model Using Back-Propagation Neural Networks. In *Computational and Information Sciences (ICCIS), 2011 International Conference on.* IEEE, 733–736.

[20] Shirin Noekhah, Ali Akbar Hozhabri, and Hamideh Salimian Rizi. 2013. Software reliability prediction model based on ICA algorithm and MLP neural network. In *e-Commerce in Developing Countries: With Focus on e-Security (ECDC), 2013 7th Intenational Conference on.* IEEE, 1–15.

[21] Jayadeep Pati and Kaushal K Shukla. 2015. A hybrid technique for software reliability prediction. In *Proceedings of the 8th India Software Engineering Conference.* ACM, 139–146.

[22] Pedro O Pinheiro, Ronan Collobert, and Piotr Dollár. 2015. Learning to segment object candidates. In *Advances in Neural Information Processing Systems.* 1990–1998.

[23] Bruno Rossi, Barbara Russo, and Giancarlo Succi. 2010. Modelling Failures Occurrences of Open Source Software with Reliability Growth. In *Open Source Software: New Horizons - Proceedings of the 6th International IFIP WG 2.13 Conference on Open Source Systems, OSS 2010.* Springer, Heidelberg, Notre Dame, IN, USA, 268–280. https://doi.org/10.1007/978-3-642-13244-5_21

[24] Pratik Roy, GS Mahapatra, and KN Dey. 2015. Neuro-genetic approach on logistic model based software reliability prediction. *Expert systems with Applications* 42, 10 (2015), 4709–4718.

[25] Pratik Roy, GS Mahapatra, Pooja Rani, SK Pandey, and KN Dey. 2014. Robust feedforward and recurrent neural network based dynamic weighted combination models for software reliability prediction. *Applied Soft Computing* 22 (2014), 629–637.

[26] Sergey A Sheptunov, Maksim V Larionov, Nataly V Suhanova, MR Salakhov, and Yuri M Solomentsev. 2016. Simulating reliability of the robotic system software on the basis of artificial intelligence. In *Quality Management, Transport and Information Security, Information Technologies (IT&MQ&IS), IEEE Conference on.* IEEE, 193–197.

[27] Yogesh Singh and Pradeep Kumar. 2010. Application of feed-forward neural networks for software reliability prediction. *ACM SIGSOFT Software Engineering Notes* 35, 5 (2010), 1–6.

[28] Adam Smiarowski, Hoda S Abdel-Aty-Zohdy, Mostafa Hashem Sherif, and Hemal Shah. 2006. Wavelet Based RDNN for Software Reliability Estimation. In *Computers and Communications, 2006. ISCC'06. Proceedings. 11th IEEE Symposium on.* IEEE, 312–317.

[29] Donald F Specht. 1991. A general regression neural network. *IEEE transactions on neural networks* 2, 6 (1991), 568–576.

[30] Yu-Shen Su and Chin-Yu Huang. 2007. Neural-network-based approaches for software reliability estimation using dynamic weighted combinational models. *Journal of Systems and Software* 80, 4 (2007), 606–615.

[31] Yu-Shen Su, Chin-Yu Huang, Yi-Shin Chen, and Jing-xun Chen. 2005. An artificial neural-network-based approach to software reliability assessment. In *TENCON 2005 2005 IEEE Region 10.* IEEE, 1–6.

[32] Giancarlo Succi, Witold Pedrycz, Milorad Stefanovic, and Barbara Russo. 2003. An investigation on the occurrence of service requests in commercial software applications. *Empirical Software Engineering* 8, 2 (2003), 197–215.

[33] V Taver, A Johannet, V Borrell-Estupina, and S Pistre. 2015. Feed-forward vs recurrent neural network models for non-stationarity modelling using data assimilation and adaptivity. *Hydrological sciences journal* 60, 7-8 (2015), 1242–1265.

[34] Gaozu Wang and Weihuai Li. 2010. Research of Software Reliability Combination Model Based on Neural Net. In *Software Engineering (WCSE), 2010 Second World Congress on*, Vol. 2. IEEE, 253–256.

[35] Hongbing Wang, Zhengping Yang, and Qi Yu. 2017. Online Reliability Prediction via Long Short Term Memory for Service-Oriented Systems. In *Web Services (ICWS), 2017 IEEE International Conference on.* IEEE, 81–88.

[36] Jinyong Wang and Ce Zhang. 2018. Software reliability prediction using a deep learning model based on the RNN encoder–decoder. *Reliability Engineering & System Safety* 170 (2018), 73–82.

[37] Alan Wood. 1996. Software reliability growth models. *Tandem technical report* 96, 130056 (1996).

[38] Yumei Wu and Risheng Yang. 2011. Study of software reliability prediction based on GR neural network. In *Reliability, Maintainability and Safety (ICRMS), 2011 9th International Conference on.* IEEE, 688–693.

[39] Fu Yangzhen, Zhang Hong, Zeng Chenchen, and Feng Chao. 2017. A Software Reliability Prediction Model: Using Improved Long Short Term Memory Network. In *Software Quality, Reliability and Security Companion (QRS-C), 2017 IEEE International Conference on.* IEEE, 614–615.

[40] Qi Yu-dong, Qu Ning, Li Ying, and Xie Xiao-fang. 2010. A BP neural network based hybrid model for software reliability prediction. In *Computer Application and System Modeling (ICCASM), 2010 International Conference on*, Vol. 15. IEEE, V15–511.