Deep Learning Model for Automation of CI/CD Regression Failure Analysis based on Text Classification using TensorFlow

Renukesh N K, Rajesh N
Student of Computer Network Engineering, Assistant Professor
Information Science and Engineering,
The National Institute of Engineering, Mysuru, India

Abstract: This work presents a method for automation of the CI/CD Regression failure analysis using Tensor flow text classification deep learning model. Regression failure stands for the recent code change has affected the previously existing features or failed when it is committed to a new environment. The aim of this proposed method is to automate the manual stuff done and to find the actual cause for the failure. We built the dataset required to train the model which is in JSON format where the key(category) represents the root cause of failure and the values represent the failures. We carried out preprocessing of data using several NLP techniques. The operations involve stemming, tokenizing, removing punctuations and transformed the unstructured raw data into a standard format and applied the bag of words model to convert the sentences into the numeric binary array because tensor flow being a math library that can understand data in the numeric form. Next, we built a DNN model and used that for training. After the model being trained, we tested it by feeding failed log files. The Accuracy of the prediction is increased by using the dropout and SoftMax functions. The proposed method reduced the time taken by Engineers to manually analyze the failures and to find the root cause of those failures. By this approach the overall analysis time of the Engineers has been reduced by 90 percent.

Index Terms - Deep learning, neural network, log mining, text classification, bag of words, tensor flow

I. INTRODUCTION

Nowadays, every Product based company will be carried out with CI/CD regression failure analysis. Regression analysis stands for the recent code change has not affected the previously existing features. The software may fail when it is executed under several environments. When the changes made to the code repository, the results may vary and may get failed for several test cases or conditions. Each company will follow different methodologies for software delivery such as Agile, DevOps etc. In the agile methodologies, companies would release software in monthly, quarterly, bi-annual. in DevOps methodologies, software releases are scheduled daily, weekly, or even multiple times a day.

Continuous integration aims at merging the work of individual developers together into a centralized repository. The changes made by them are validated by creating builds and executing automated tests against the builds. The primary goal is the early detection of integration bugs. Continuous integration helps to verify that the developed application will not malfunction when new changes are made or upgraded to the next version. Continuous delivery is the next level of continuous integration in which the software release process is done automatically to make easy and flexible deployments into production any time. Continuous delivery makes sure that it automates from committing the code into the centralized repo to releasing fully tested functional builds without any hassles that are ready for production.

Continuous deployment is the extension of continuous delivery so that every change made to the codebase passes all the tests. In such cases, there is no need for a person to decide when and what goes into final production. The final stage of CI/CD system will automatically deploy the builds successfully exit the delivery pipeline. Text classification is the method which classifies text into a particular category. Automated text classification is done by various algorithms such as neural networks, random forest, naive Bayes, Decision trees etc. Supervised text classification is the approach where we have a predefined set of examples according to our requirement. The model will be trained using the predefined dataset and will give predicted output based on the categorized data set. The model will be evaluated by feeding unobserved data to know the accuracy of the prediction and it will be continuously tuned to increase the accuracy close to the desired output. Unsupervised text classification, the training data is unlabeled. The system tries to learn without a teacher. The design of the text classification algorithm needs to be done properly because the performance of the text classifier depends on how we input the model with training data and the testing set. We need to apply various NLP operations on the dataset in order to make the classifier predict correctly such as stemming, tokenizing, noise free corpus, lemmatization. Bag of Words model is used to create a unique list of words in our vocabulary and turn words into vectors. The proposed approach to classification was implemented using the TF-Learn with TensorFlow library for deep neural networks with the models trained on a CPU. TF-Learn is a deep learning library built on the top of Tensor flow. The main objective of the work is to automate the regression failure analysis using the tensor flow text classification deep learning model to predict the root cause of the script failure. The paper is organized as the following: section II describes the literature review, section III brief about the existing system, section IV explains the proposed approach and the results of the given solution, last section provides the general discussion and the future scope.

II. LITERATURE REVIEW

2.1 Deep Learning

Deep learning is a subbranch of machine learning that enacts the working of the human brain in processing data and creating patterns for use in decision making. It sets up neural networks to simulate the analysis and learning functions of human brains, it simulates the mechanism of human brains to resolve data like images, sound and text. With the successful application in image identification and speech recognition, deep learning is more and more popular in natural language processing [1]. Deep learning is a family of representation learning algorithms employing complex neural network architectures with a high number of hidden layers, each composed of simple but non-linear transformations to the input data. Given enough such transformation modules, very complex functions may be modeled to solve classification, regression, transcription and numerous other learning tasks.

Artificial neural networks are the roots of deep learning. As neurons are the fundamental element of the human brain, artificial neurons form the basic structure of neural network. ANNs are very powerful, scalable and are used to tackle complex computations. The perceptron is one of the simplest ANN architectures. It is based on a slightly different artificial neuron called linear threshold unit, here the inputs and outputs are the numbers and each input connection is associated with a weight. The LTU computes a weighted sum of its inputs then applies a step function to that sum and outputs the result.

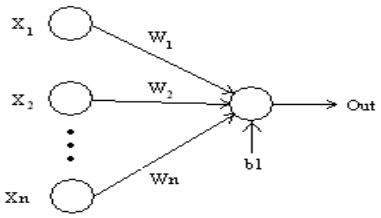

Figure 1: Single Layer Neuron

2.2 TensorFlow

In November 2015, Google released TensorFlow, an open source deep learning software library for defining, training and deploying machine learning models. In TensorFlow, machine learning algorithms are represented as computational graphs. A computational or dataflow graph is a form of directed graph where vertices or nodes describe operations, while edges represent data flowing between these operations [2]. In TensorFlow, edges represent data flowing from one operation to another and are referred to as tensors. A tensor is a multi-dimensional collection of homogeneous values with a fixed, static type. The number of dimensions of a tensor is termed its rank. A tensor's shape is the tuple describing its size, i.e. the number of components, in each dimension. In the mathematical sense, a tensor is the generalization of two-dimensional matrices, one-dimensional vectors and scalars, which are simply tensors of rank zero. A tensor is a central unit of data in Tensor flow. The graph defines the specific operation. It does not carry out any operation. The session helps in the execution of graphs or part of graphs. Here a, b are the input tensors, c is the output tensor, add is the node represents mathematical operation.

TensorFlow has several use cases, it us used in Video/sound recognition, Text based application, Image recognition, Time series algorithms, Motion detection, Sentiment analysis. Tensor flow comes with an inbuilt tool called Tensor board helps in visualizing the computational graphs, allowing the user to understand exactly how data flows through it.

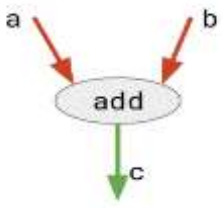

Figure 2: A simple Numerical Operation representing Tensors

2.3 Neural Network architecture

Neural networks are the base for deep learning. Perceptron contain a single layer of neurons connected to all inputs and they omit whatever inputs are fed. Multi-layer perceptron are composed of one input layer, one or more hidden layers, and one output layer. The input layer is the first layer which receives the input, the layers which perform the specific operations on the incoming data are hidden layers, the final layer which generates the output is the output layer. There are several subsections in the architecture: deep neural networks, convolutional neural networks, recurrent neural networks. In this proposed approach we have used deep neural networks. When the ANN has two or more hidden layers, it is called a Deep neural network.

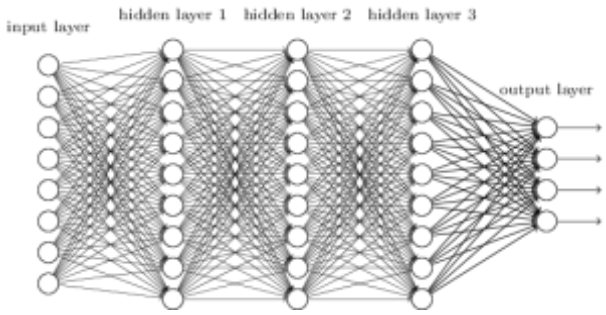

Figure 3: Deep Neural Network Architecture

2.4 Bag of Words(BoW) Model

The problem with the text data is, they are in a format which the machine learning algorithms cannot work with them, they need to be converted to a numeric form, such as vectors. The approach of extracting the feature from the text data is called the bag of words. They represent the occurrence of words. The problem with the text data is, they are in a format which the machine learning algorithms cannot work with them, they need to be converted to a numeric form, such as vectors. The approach of extracting the feature from the text data is called the bag of words. They represent the occurrence of words. According to this the order or structure of the words is not considered. the model concentrates whether the known words occurred. The first step in the bag of words model is to create a vocabulary of unique words. The method to know the presence of a word is a Boolean value. 0 for present and 1 for absent. The problem arises when our document contains only a few words matching the vocabulary. To reduce the sparse representation (vectors with lots of zero scores), we need to carry some text cleaning techniques such as ignoring case, ignoring punctuation, ignoring stop words, stemming and using some word frequency detection algorithm such as TF-IDF. There are some limitations in Bag of words model: Vocabulary, sparsity and meaning. We need to be very careful while designing the vocabulary for the text corpus (large collection of text data). It is very important to handle the sparse representation. Bag of words model ignores the semantics and the order of the words in the document.

III. EXISTING SYSTEM

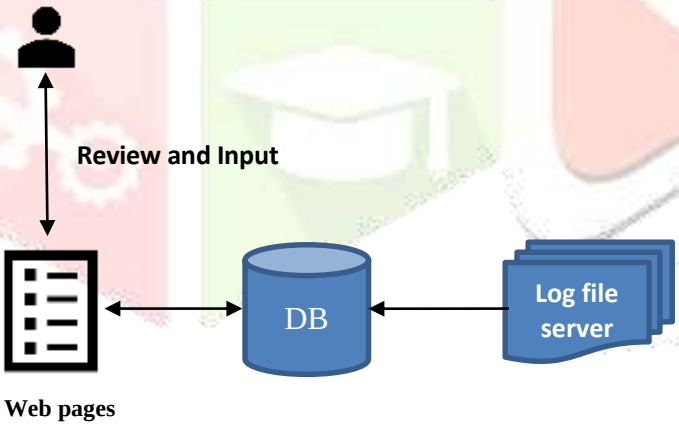

Figure 4: Existing System

The workflow of the existing system is as follows: When the scripts fails during the execution, they are logged in the database, the concerned person need to manually get those failed logs and to find the root cause for the script failure and need to update it manually. This consumes a lot of time and the manpower.

IV. PROPOSED APPROACH

In this work, we proposed an approach which overcomes the manual tasks mentioned in the existing system. we used a deep neural network classifier to identify the root cause of the failures in logs.

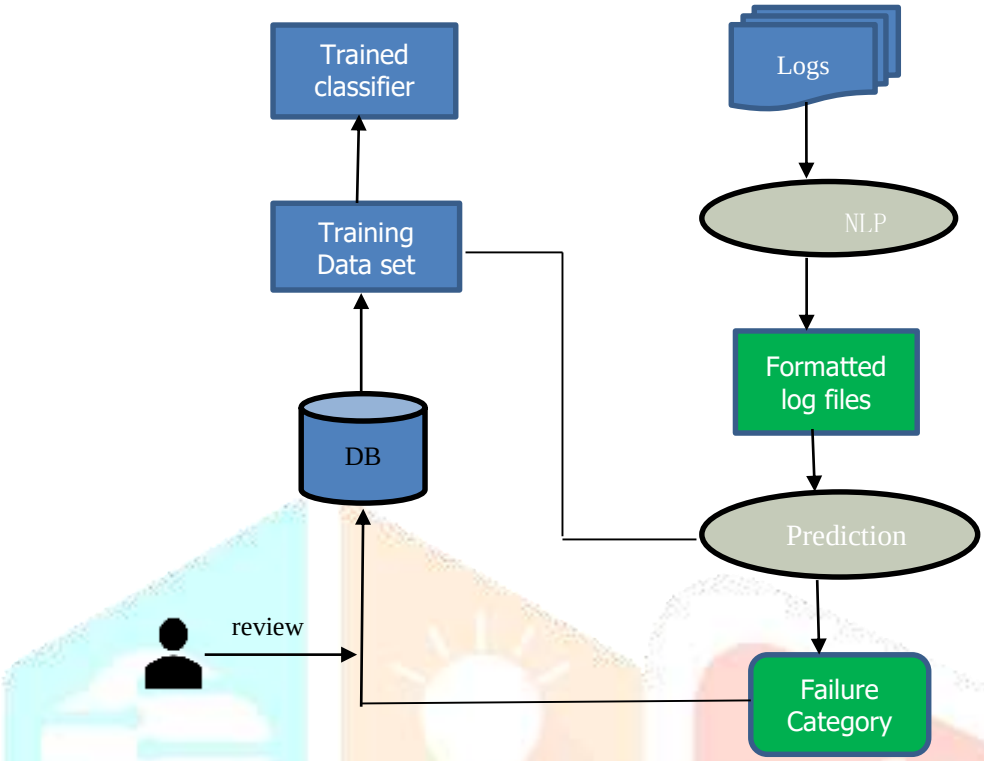

Figure 5: Proposed System

4.1 Data Collection

The training set was generated from the database that contains failed logs. The dataset is in JSON format(key-value) where the key represents the failure reason and the value contains the list of failures. As we are creating the dataset from the log files which are unstructured, we need to convert them to a structured format as the training set is the base for any model. The training set consists of 15 categories and each category consists of 40 failure reasons, totally the dataset consists of 600 samples.

4.2 Data Preprocessing

Data preprocessing is carried out in order to remove the noise from the data, for example, special symbols, punctuations, cases, URLs, duplicates. We used Natural language processing toolkit(NLTK) to carry out the preprocessing of the dataset. Since we are creating the dataset from logfiles we need to design the preprocessing properly. We did tokenization, stemming, removing punctuation, ignoring stop words. After the data preprocessing we created the bag of words for each document obtained.

4.3 Word to Vector conversion

Deep neural network architectures are giving ground breaking results in every field of artificial intelligence and, Natural Language Processing is one of it [3]. The idea is to model the words into an appropriate vector space such that, we can use those vectors to perform any task. Since we are using Tensor flow which is a mathematical library, that does not accept data in text form, so we used Bag of words to convert data into numeric binary array.

4.4 Proposed Deep Neural Network architecture

After the conversion of words to vectors, now we initiate Tensor flow text classification, we built a simple deep neural network using TF-Learn which is deep learning library built on the top of TensorFlow. The defined network consists of 1 input layer, 2 hidden layers, 1 output layer.

These layers are fully connected layers, in which every neuron in one layer is connected every neuron in another layer. We used SoftMax activation function applied at the output layer of the network since ours is a classification problem. Activation functions will transform input

signals into output signals. SoftMax activation function will be usually used in the multiclass classification that will give values to each class and can be interpreted as probabilities.
We used this defined network to train our model and we trained the model for 1000 epochs and we kept the batch size as 16. We used Adam optimizer to iteratively update the network weights. We randomly shuffled the training data and split into training and validation set.
After training the model we tested it by feeding failed log files as testing set. Before feeding the logs, we formatted those log files and converted to a structured data. As the log files contain much irrelevant data, so we did log mining and grepped only the error lines and applied some text processing techniques such as stemming, tokenizing, removing Stop words, ignored punctuations and applied TF-IDF to identify the frequency of words. By this, we can reduce the sparse representation in the bag of words.

$$\text{TF-IDF} = tf(w) * idf(w)$$

TF -> Term frequency
IDF-> Inverse document frequency

tf(w) is the number of times the word appears in the document by the total number of words in that document.

idf(w) is the number of documents by the number of documents that contain word w.

The prediction accuracy was 68% for the first run. And after tuning the parameters and increasing the number of iterations the accuracy of the prediction was 72%.

V. Conclusion And Future Scope

In this work we proposed the method that automates the CI/CD regression failure analysis using deep learning TensorFlow text classification algorithm. The goal was to find the root cause of the script failure. We are completely dependent on the log files, we did not use any standard ready dataset, we created our own dataset using the log files. This work was very challenging because the training dataset is the base for the model to train. we made use of deep neural networks and defined the network using TF-Learn, which is a higher level API built on the top of TensorFlow. We got pretty good accuracy. This work can be continued in order to increase the accuracy, to find the better way to create vectors and to use advanced more powerful libraries to support scalable needs in future. Using other types of neural network architectures are worth discovering.

References

[1] Guolong Liu, Xiaofei Xu, Bailong Deng, Siding Chen, Li Li. "A Hybrid Method for Bilingual Text Sentiment Classification Based on Deep Learning, IEEE SNPD 2016, May 30-June 1, 2016, Shanghai, China.

[2] Peter Goldsborough, Fakultät für Informatik, "A Tour of TensorFlow".

[3] Anmol Chachra, Pulkit Mehndiratta, Mohit Gupta "Sentiment Analysis of Text using Deep Convolution Neural Networks" Proceedings of 2017 Tenth International Conference on Contemporary Computing (IC3), 10-12 August 2017, Noida, India.

[4] Feng shen, Xiong luo, Yi chen, "Text classification Dimension reduction algorithm for Chinese web page based on deep learning".

[5] Piotr Semberecki, Henryk Maciejewski," Deep Learning methods for Subject Text Classification of Articles", Proceedings of the Federated Conference on Computer Science and Information Systems pp. 357–360 ISSN 2300-5963 ACSIS, Vol. 11.

[6] Kamran Kowsari, Donald E. Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi§, Matthew S. Gerber§, and Laura E. Barnes, "HDLTex: Hierarchical Deep Learning for Text Classification", 2017 16th International Conference on Machine Learning and Applications, IEEEDOI 10.1109/ICMLA.2017.0-134.

[7] Abdalraouf Hassan, Ausif Mahmood, "Efficient Deep Learning Model for Text Classification Based on Recurrent and Convolutional Layers", 2017 16th IEE International Conference on Machine Learning and Applications.

[8] Abubakr H. Ombabi, Onsa Lazzez, Wael Ouarda, Adel M. Alimi, "Deep Learning Framework based on Word2Vec and CNN for Users Interests Classification", 2017 Sudan Conference on Computer Science and Information Technology (SCCSIT).

[9] Fatih Ertam, Galip Aydin, "Data Classification with Deep Learning using Tensorflow", (UBMK 17) 2nd international conference on computer science and engineering.