Summary of Summer Intern

Kefan Xu
kefanxu@pku.edu.cn

I. Introduction

In the summer of 2019, I got a chance to take part in research projects as student of summer intern in Tan, Lin' s group. During this summer, I have mainly gotten involved into two projects:

- **Android Intent Specific generation**
- **Defect Prediction**

And here is specific introduction of these two projects below.

II. Android Intent Specific generation

a. Background

The motivation of this automatically extracting project is mainly have two purposes.
The first purpose is to get an output generated automatically and compare it with the manually generated one. In this way, the automatically one serves as a new way to check out some mistakes. (having checked out 10+ mistakes in the file which is generated manually).

The second purpose is to give a way to deal with new intent and generate a file with useful attributes. Because the introduction of Intent may update frequently and if manually checking them all the time, it may waste a lot of time to finish generating the expected output.

b. Progress

The main progress has made in this project is that having given a code which can automatically deal with raw text and give the data which is well structured.

The code mainly consists of three parts:

Deal with raw data - In this project, I mainly focus on dealing with text of introduction of Android Intent Specific. The text of this webpage is highly structured, and in this way, we can easily parse the description into several segments. We can use these segments for generating each attribute of the Intent Specific later.

Attributes Extraction – To get a well-structured output, we need to automatically extract some attributes from the related description segments. In this project, the extraction is mainly based on three methods: **lexical matching**, **pattern matching** and **mapping**.

The lexical matching means to detect specific words in the original text to extract valuable information, it highly relied on structure. (for example: the attribute of "Constant value" is usually extracted from the original text from the line with the word like "Constant Value: ")

The pattern matching means is not based on a specific word but based on some patterns. For example, the data of Extra, permission are usually with link. In this way, it has the structure of "\u202a EXTRA_XXX \u202c", and we can use this kind of pattern (with prefix and suffix) to extract some meaningful information. Pattern matching is less relied on specific word.

The mapping is to get the attribute of type and Mime type. After construct a list of type, we can map the preliminary extract information into some types.

Output Generation – recording the extracted attributes into a highly structured json file. The json file consists of 4 parts: action(basic information of action), action_extra(relationship between the action and the extra), extra_field(basic information of extra), data(the necessary data(input) of a specific action)

c. Result

Based on comparing with the automatically generated Intent specific, we successfully corrected 10+ mistakes in the manually generated file.

The accuracy of automatically generating parser is calculated in the way of taking the given manually generated output as ground truth (corrected the mistakes) and count the number of different fields. The overall accuracy is around ninety percent. However, although it works well for the given webpage, it may not work well for the one with a different structure or even don' t have a structure. To deal with these cases, maybe advanced method (such as NLP) is necessary.

All the related statistics are recorded in a Google docs. The link to the file is attached.
Link to file:
https://docs.google.com/document/d/1sQ7bZHEub1P8GofIkgrSnmaw3H_6MvnRO1ISq8hG3fM/edit?usp=sharing

III. Defect Prediction

a. Background

This project is based on the paper of “Deep Semantic Feature Learning for Software Defect Prediction”. It’s aiming to predict whether the code commits are buggy correction. This project retrieved data from GitHub and the prediction part is mainly based on neural network.

b. Progress

Get familiar with project – To achieve some result in this opening project, starting with getting familiar with previous work is necessary. First, I have read the original paper so that I understand the motivation and the structure of this model. This really help a lot to start the future work and have a better understanding of the whole project. Moreover, I also carefully read the code of data extraction, and understood how the datasets were constructed and what feature may be able to improve. Finally, I have also got familiar with the existing model, including the LSTM and Conv1d, by which I got familiar with training model on the Linux server and how to train model using the extracted datasets.

Try to change the model – This part is finished with Jiahe, we tried to combine the word embedding part of the XLNet and the training model. Because the time is kind of limited(only have 2 weeks to training), we only run our new model on a subset(contains 150000 data).

c. Result

The result of our combined model is kind of similar with original model, only improved slightly (about 1 percent, from 57% to 58% on average). And the problem of overfitting is very serious, the validation loss begins to rise after the second epoch. Using a bigger dataset may alleviate the problem of overfitting.