Construction of Effective Software Defect Prediction Model via Machine Learning

Li Jidong 17M38124

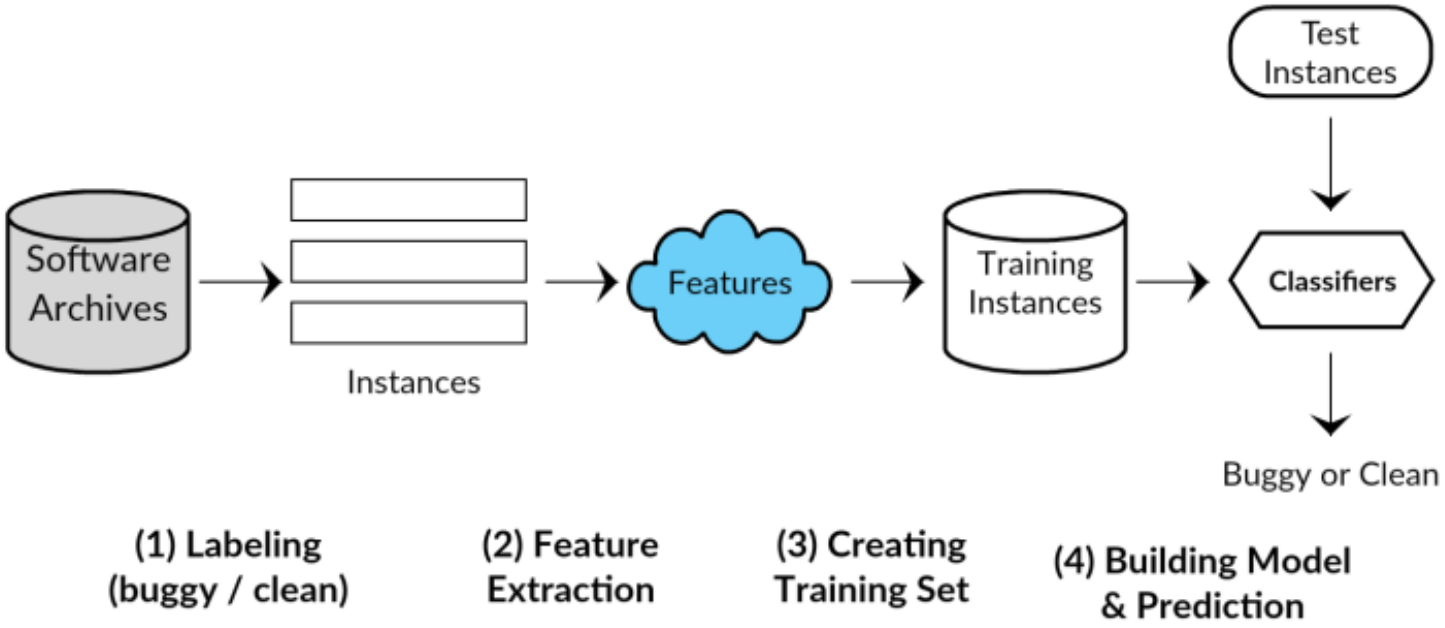
2018/09/13

Background(1)

- Software testing is a serious section
  - Lead to severe problems
- How to test the software efficiently?
  - Software defects prediction(SDP)

Background(2)

Workflow of SDP[1]



[7] J. Li, P. He, and MR. Lyu, ”Software defect prediction via convolutional neural network,” in QRS’17: Proc. Of the International Conference on Software Quality, Reliability and Security,2017

Related Research

- Metrics
  - Code based metrics
    - McCabe metrics[2]
    - Halstead metrics[3]
    - CK metrics[4]
  - Software processing metrics
    - Change metrics[5]
    - Developers based metrics[6]

[2] McCabe TJ. A complexity measure. IEEE Trans. on Software Engineering, 1976,2(4):308-320. [doi: 10.1109/TSE.1976.233837]
[3] Halstead MH. Elements of Software Science (Operating and Programming Systems Series). New York: Elsevier Science Inc., 1977.
[4] Chidamber SR, Kemerer CF. A metrics suite for object oriented design. IEEE Trans. on Software Engineering, 1994,20(6): 476-493. [doi: 10.1109/32.295895]
[5] Nagappan N, Ball T. Use of relative code churn measures to predict system defect density. In: Proc. of the Int'l Conf. on Software Engineering. 2005. 284-292. [doi: 10.1145/1062455.1062514]
[6] Graves TL, Karr AF, Marron JS, Siy H. Predicting fault incidence using software change history. IEEE Trans. on Software Engineering, 2000,26(7):653-661. [doi: 10.1109/32.859533]

Existed problems

- Traditional metrics fail to capture semantic information of programs[1,7]
  - Semantic: describe the process of execution of programs
- Latest machine learning classifiers were hardly considered
- Have not deeply classified the defects

[7] S. Wang, T. Liu, and L. Tan, “Automatically learning semantic features for defect prediction,” in ICSE’16: Proc. of the International Conference on Software Engineering, 2016.

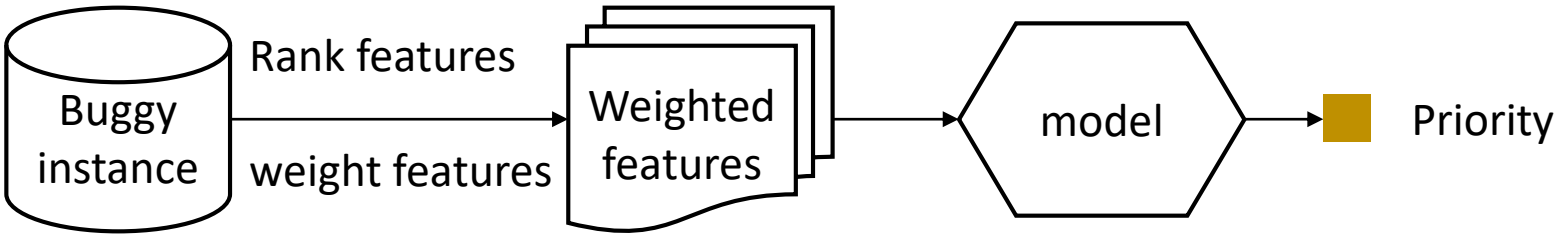
Research Purpose

- Construct effective software defect prediction model via machine learning

- Sub-goals:
  - Develop useful metrics
  - Develop a methods to predict the priority of buggy modules
  - Verify whether the latest machine learning classifier are superior than traditional one

Approach

- Extract the semantic feature from programs
  - By NLP or deep learning
- Use the machine learning classifier to predict the instances
  - Xgboost, lightGBM & Catboost VS traditional classifiers
- Deeply classify the priority of buggy instances



Conclusion

- Background
  - Efficiency of software testing
- Goal
  - Effective software defect prediction model
- Sub-goal
  - Useful metrics
  - Priority of buggy instances
  - Comparison of classifiers

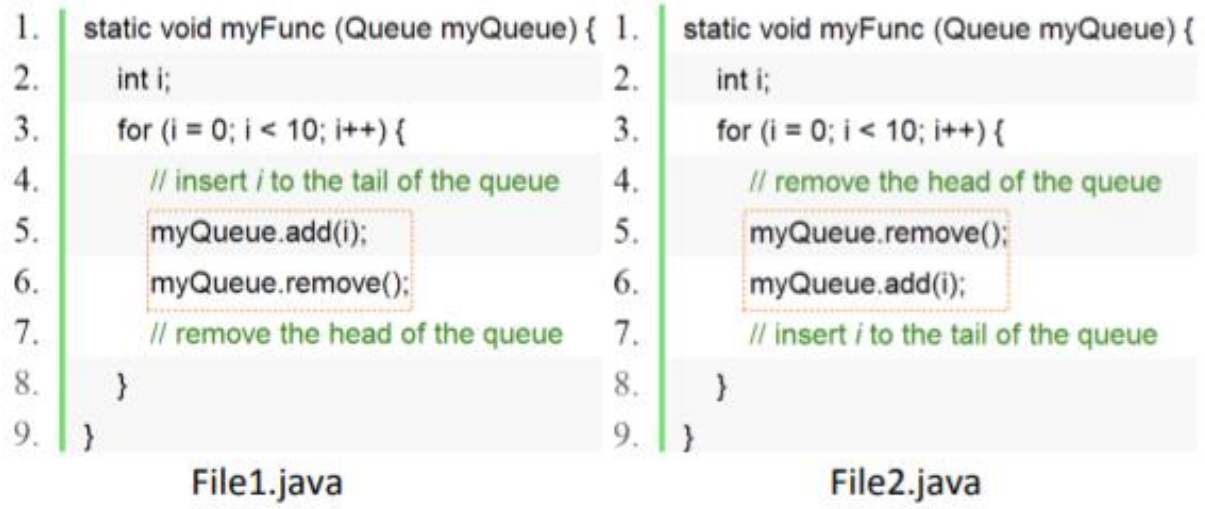
Preparatory Slides

Program Semantics

- Semantics describes the processes a computer follows when executing a program in that specific language

Extra Related Research(1)

- Relationship between semantic information and buggy module[1]



```
1. static void myFunc (Queue myQueue) {
2.     int i;
3.     for (i = 0; i < 10; i++) {
4.         // insert i to the tail of the queue
5.         myQueue.add(i);
6.         myQueue.remove();
7.         // remove the head of the queue
8.     }
9. }
```

File1.java

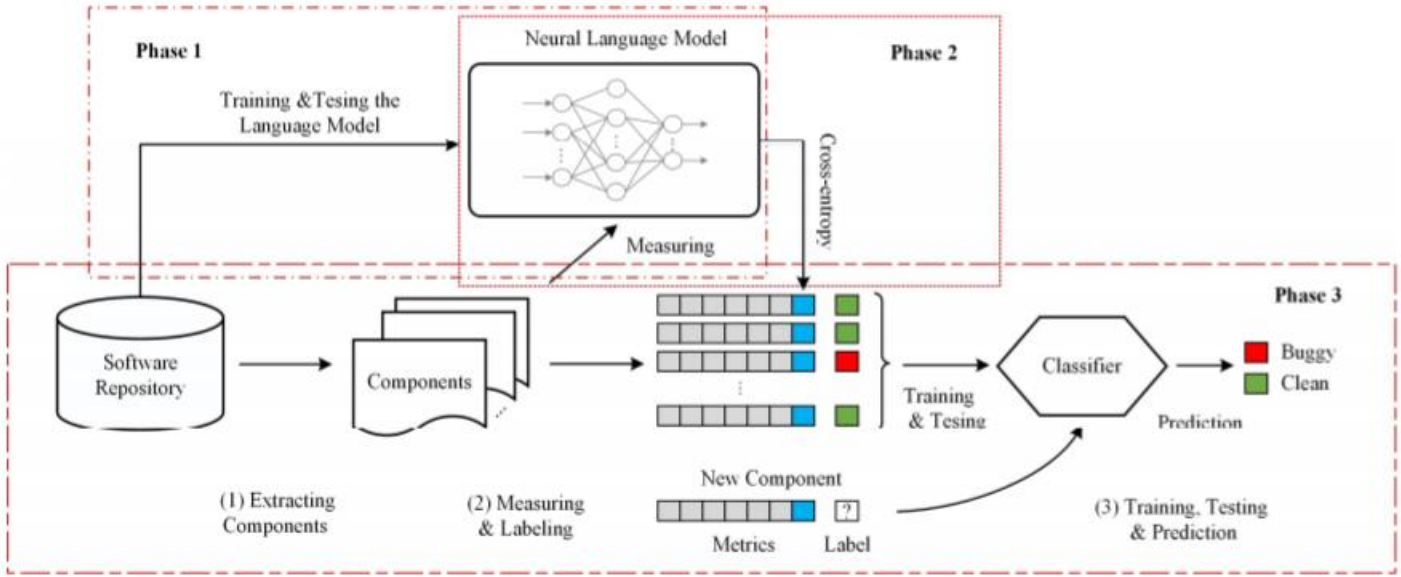
```
1. static void myFunc (Queue myQueue) {
2.     int i;
3.     for (i = 0; i < 10; i++) {
4.         // remove the head of the queue
5.         myQueue.remove();
6.         myQueue.add(i);
7.         // insert i to the tail of the queue
8.     }
9. }
```

File2.java

Fig. 1. A motivating example. *File2.java* will encounter an exception when calls *remove()* at the beginning if the queue is empty.

Extra Related Research(2)

- One example of Workflow[8]



[8] X. Zhang, K. Ben & J. Zeng, "*Cross-Entropy: A New Metric for Software Defect Prediction*", in QRS's 18: Proc. of: International Conference on Software Quality, Reliability and Security,2108

Example of deep classification

- The dataset can be seen in my dataset
  - https://github.com/tklab-group/mthesis-li.git
- Our plan:
  - Unlike the datasets, We just roughly classify the priority of each bug module, instead of each bug.

Latest machine learning classifiers

- Xgboost:
  - https://xgboost.readthedocs.io/en/latest/
- Lightgbm:
  - https://lightgbm.readthedocs.io/en/latest/
- Catboost
  - https://catboost.ai/