Feature Generation for Software Defect Prediction via Pre-trained Model of Code Semantics

Midterm Presentation

M2 LI,Jidong

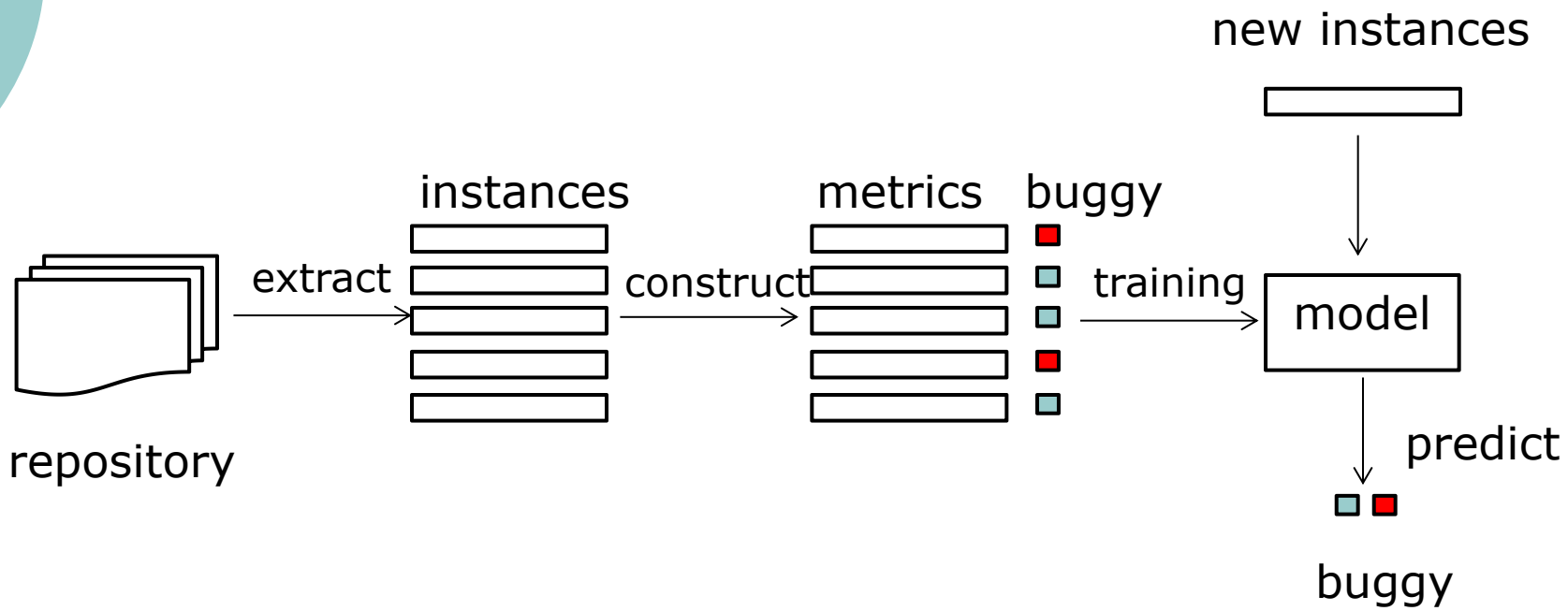
2019/01/31

Background I-III

- Make early preparation
- costy and time-consuming for bug detection.
- preciously identify bugs
  - concentrate on specific modules

Background II-III

The process of SDP



Background III-III

- machine learning[1]
  - Logistics Regression (LR)
  - Support Vector Machine
  - Ensemble learning
- metrics [2]
  - CK
  - McCabe
  - Line of Code(LOC)

[1]Paramshetti, P., & Phalke, D. A. (2014). Survey on software defect prediction using machine learning techniques. International Journal of Science and Research (IJSR), 3(12), 1394-1397.

[2]Chen X, Gu Q, Liu WS, Liu SL, Ni C. Survey of static software defect prediction. Ruan Jian Xue Bao/Journal ofSoftware, 2016,27(1):1−25 (in Chinese). http://www.jos.org.cn/1000-9825/4923.htm

Problems

- traditional metrics
  - fail to extract the semantics of code
- deep learning based
  - only word embedding.
  - not effective to extract semantics.

Research Purpose

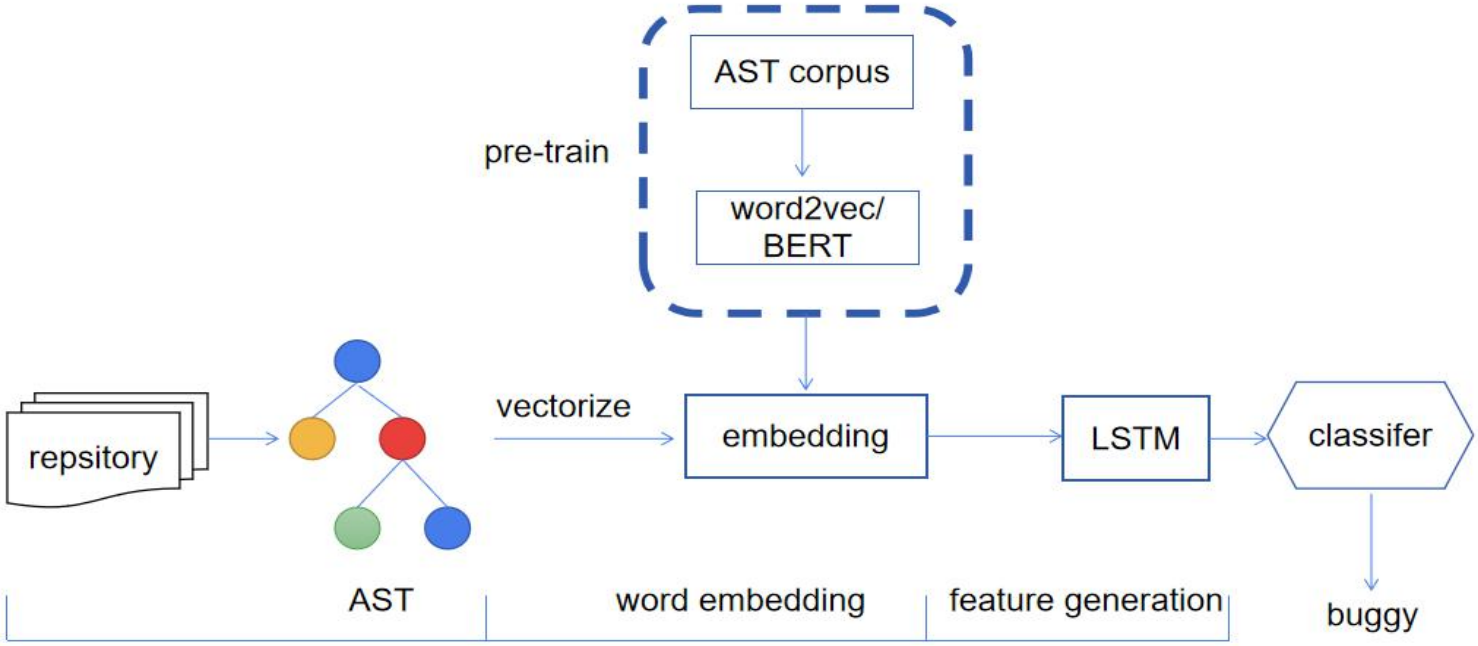
- construct effective SDP model
- improve the perfomance of semantics extraction
- identify how well deep learning based model perform

Approach I-II

- add the pre-trained into word embedding
- employed the-state-of-art pre-train model BERT.
- generate feature by deep learning models.

Approach II-II

whole process of the approach



Progress

- Data collection
  - AST corpus(Done)
  - Defect data(*Doing*)
- Experiment(*Todo*)
- Evaluation
  - Research questions(*Done*)
  - comparison group(*Done*)
  - Making benchmark(*Todo*)

Code Naturalness

- naturalness hypothesis[1]
  - similar to human language
- text feature extraction
  - count based(Bag of Word[2] & TF-IDF[3])
    - unable to extract semantics
  - prediction based(word2vec[4] & Bert[5])

[1]Miltiadis Allamanis, Earl T Barr, Premkumar Devanbu, and Charles Sut ton. A survey of machine learning for big code and naturalness. ACM Computing Surveys (CSUR), 51(4):81, 2018
[2]Sriram, B., Fuhry, D., Demir, E., Ferhatosmanoglu, H., & Demirbas, M. (2010, July). Short text classification in twitter to improve information filtering. In Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval (pp. 841-842). ACM.
[3]Ramos, J. (2003, December). Using tf-idf to determine word relevance in document queries. In Proceedings of the first instructional conference on machine learning (Vol. 242, pp. 133-142).
[4]Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Effi-cient estimation of word representations in vector space. arXiv preprintarXiv:1301.3781, 2013.
[5]Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert:Pre-training of deep bidirectional transformers for language understanding.arXiv preprint arXiv:1810.04805, 2018.

Experiment Design I-III

- effectiveness validation
  - our model+LR
  - traditional metrics(CK,LOC...)+LR
- Reasoning for LR
  - representive machine learning classifier.
  - easy to explain and understand.
  - good performance in text classification.

Experiment Design II-III

- validation of pre-train model
  - our model with pre-trained
  - without pred-trained
- reasoning
  - neural network need large data for training.

Experiment Design III-III

- validation of sematics
  - our model+LR
  - BOW+LR
  - TF-IDF+LR
- reasoning
  - How will can semantics affect SDP
  - importance of semantics of code.

Conlusion

- pre-trained model for SDP
- deep learning based SDP model
- verification of effectiveness and importances of semantics