CLC_____ Number_____

UDC_____ Classification Level_____

SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

Undergraduate Thesis

Bachelor Thesis

Author : Junda Ai

Student ID : 11711310

Department : Computer Science and Engineering

Major : Computer Science and Technology

Supervisor : Prof. Yepang Liu

Finished Time : April, 2021

诚信承诺书

1. 本人郑重承诺所呈交的毕业设计（论文），是在导师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料均真实可靠。

2. 除文中已经注明引用的内容外，本论文不包含任何其他人或集体已经发表或撰写过的作品或成果。对本论文的研究作出重要贡献的个人和集体，均已在文中以明确的方式标明。

3. 本人承诺在毕业论文（设计）选题和研究内容过程中没有抄袭他人研究成果和伪造相关数据等行为。

4. 在毕业论文（设计）中对侵犯任何方面知识产权的行为，由本人承担相应的法律责任。

作者签名: _____

_____ 年___ 月___ 日

Preface

This undergraduate graduation project is an extension of the previous work on Python API evolution conducted by Hengcheng Zhu and Zhaoxu Zhang of SUSTech class of 2020, which was summarized in their SANER 2020 paper How Do Python Framework APIs Evolve? An Exploratory Study.

Junda Ai
March, 2021 at SUSTech

Contents

ABSTRACT

Python is a popular dynamic programming language that has thrived in the past decade with massive applications in various deciplines.

Keywords: Python, API Evolution, tree differencing, dynamic programming language

Chapter 1 Introduction

Python is a popular dynamic programming language known for its relatively free form and powerful development frameworks, thus fulfilling the need of both programming beginners and professional programmers. Massive amount of softwares in various deciplines including deep learning, data analytics, and web services heavily rely on frameworks written in Python, such as TensorFlow, Pandas, and Django, to power their client applications. This have seen it boost in popularity over recent decades.

The same as frameworks written in other programming languages, Python frameworks' APIs evolve through time in order to be satisfactory. Adding bugfixes and feature implementations into existing APIs are seen commonly, while API deprecations and updates might also occur during structural redesign or refactoring in major releases. Among all change patterns in Python API evolution, some of which are backward-compatible and would retain the integrity of dependent client applications, often referred to as non-breaking changes. While others would cause compilation or runtime errors that obstruct client programs using APIs prior to these changes, called breaking changes.

A previous work looked into the common patterns in Python API evolution and compatibility issues incurred by breaking changes, and built a tool called PyCompat to automatically detect API misuse caused by breaking changes [1]. But the automation of the detection of compound API changes such as method renaming, method relocation, and parameter renaming remains unachieved. In this undergraduate graduation project, an empirical study on the classification of compound changes was first conducted by investigating commit logs of popular Python frameworks including TensorFlow and Flask and Django. And I would try to build a tool that automatically detects compound changes, based on the edit scripts generated by a tree-differencing tool. To summarize, this thesis makes two major contributions:

- I qualitatively analyzed the major types of compound changes occurred in Python API evolution.
- I design and implemented a tool to automatically detect compound changes classified in the above empirical study.

Chapter 2 Background

2.1 The Python Programming Language

Python is a popular dynamic programming language.

2.2 Python API Evolution

There are some patterns in the evolution of Python APIs.

2.3 Compound Changes

What are the compound changes?

2.4 Tree Differencing

I use tree-differencing to help detecting API renamings.

Chapter 3 Tool Design

This is tool design.

3.1 gumtree

gumtree is an ast-differencing tool that can detect insert, delete, update, and move actions in python source files.

3.2 Edit Script Evaluation

I evaluate edit scripts generated by gumtree to decide whether a change is method renaming or not.

Chapter 4 Tool Evaluation

This is tool evaluation.

Chapter 5 Conclusions

This is conclusion.

References

[1] ZHANG Z, ZHU H, WEN M, et al. How Do Python Framework APIs Evolve? An Exploratory Study[C/OL] // 2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER). Los Alamitos, CA, USA : IEEE Computer Society, 2020 : 81 – 92. https://doi.ieeecomputersociety.org/10.1109/SANER48275.2020.9054800.

[2] FALLERI J, MORANDAT F, BLANC X, et al. Fine-grained and accurate source code differencing[C/OL] // ACM/IEEE International Conference on Automated Software Engineering, ASE '14, Vasteras, Sweden - September 15 - 19, 2014. 2014 : 313 – 324. http://doi.acm.org/10.1145/2642937.2642982.

Acknowledgements

This is ackowledgement.

Junda Ai
March, 2021