

令和 6 年 6 月 21 日現在

機関番号：34316

研究種目：基盤研究(C)（一般）

研究期間：2021～2023

課題番号：21K11839

研究課題名（和文）テストケースを考慮したソースコード推薦環境の構築

研究課題名（英文）An environment for recommending source code considering test cases

研究代表者

山本 哲男（Yamamoto, Tetsuo）

龍谷大学・先端理工学部・准教授

研究者番号：40388129

交付決定額（研究期間全体）：（直接経費） 3,200,000 円

研究成果の概要（和文）：本研究では、単純なコード補完にとどまらず、ソースコード全体に広げて補完する仕組みを提案する。まずは、単純なコード補完について実施した。具体的な、メソッド呼び出し文の順序を用いて API の推薦に限定した手法を提案した。さらに、ソフトウェアの品質を確保するために不具合の含まれる可能性の高いソースコードを予測する手法を提案した。そのための手法として、深層学習を用いてソースコード内に含まれる不具合を予測した。

研究成果の学術的意義や社会的意義

ソフトウェア開発においてプログラミング工程は開発者の知識が問われる作業であり、その言語の文法などを知っている必要がある。さらに、その文法に従い要求通りのソースコードを記述する必要がある。しかし、要求通りのソースコードを実現するためには、複数の方法があるため、正解を導き出すことは難しい。知識を補うためにソースコードの推薦や補完などの機能を備えた統合開発環境が存在するが、部分的な補完に留まっており、開発全体の作業を補うには至っていない。本研究では、単純な補完にとどまらず、開発全体を補完していく。

研究成果の概要（英文）：In this study, we propose a mechanism that extends beyond simple code completion to complete entire source codes. Initially, we focused on simple code completion. Specifically, we proposed a method limited to API recommendations using the order of method call statements. Furthermore, to ensure software quality, we proposed a method to predict source codes that are likely to contain defects. As a method to achieve this, we used deep learning to predict defects within the source code.

研究分野：ソフトウェア工学

キーワード：ソースコード コード補完 コード推薦

1. 研究開始当初の背景

ソフトウェア開発においてプログラミング工程は開発者の知識が問われる作業である。あるプログラミング言語でソフトウェアを開発する際に、その言語の文法などを知っている必要がある。さらに、その文法に従い要求通りのソースコードを記述する必要がある。

しかし、要求通りのソースコードを実現するためには、複数の方法があるため、正解を導き出すことは難しい。また、プログラミング工程では、開発者自身の間違った思い込みや知識不足により、ソースコード中に欠陥を埋め込む可能性が大きい。

知識を補うためにソースコードの推薦や補完などの機能を備えた統合開発環境が存在するが、部分的な補完に留まっており、開発全体の作業を補うには至っていない。

2. 研究の目的

ソフトウェア開発のコストで大きい割合を占めるテスト工程を短縮することは、ソフトウェア開発の現場で求められている重要な要素の一つである。テスト工程で発見した欠陥を修正するのは、コストのかかる作業であるため、欠陥の少ないソースコードにすることが求められる。しかし、ソースコードの記述の自由度の高さから、ソースコードを記述するプログラミング工程では欠陥を埋め込みやすい問題がある。

問題を解決するために、デザインパターンやイディオム（以降単にパターンと呼ぶ）と呼ばれる典型的なソースコードのパターン集を作成し、利用する場合があるが、パターンを作成することは難しい作業であり、また、パターンを利用するには開発者の能力が必要な作業である。

そこで、多くの品質の高いソースコードから知識を自動的に抽出（知識のソースコードマイニング）し、その知識を利用できれば、ソフトウェア開発に有効な手法が実現できる。この知識を蓄えることで、開発者は次のどのような処理を記述すべきかが分かるようになり、生産性向上が見込める。

3. 研究の方法

本研究では、単純なコード補完にとどまらず、ソースコード全体に広げて補完する仕組みを提案する。まずは、単純なコード補完について実施した。本研究ではメソッド呼び出し文の順序を用いて API の推薦に限定した手法を提案する。既存のソースコードからメソッド呼び出し文を抽出し、メソッド呼び出し文の並びを作成する。

これを深層学習モデルの 1 つである Masked Language Model に学習させ、ある指定した場所に適切なメソッド呼び出し文を予測し開発者に候補を提示する。

本手法は、学習プロセスと推薦プロセスの 2 つのプロセスから構成されている。

学習プロセスでは、既存のソースコードからメソッド呼び出し文を抽出し、メソッドごとにメソッド呼び出し文を順に並べる。メソッド呼び出し文を単語と見なし、また、メソッドごとのメソッド呼び出し文の並びを 1 つの文と見なすことで Masked Language Model に適用し、学習させる。

推薦プロセスでは、まず開発者がメソッド呼び出し文を挿入したい場所を決定する。決定した場所の前後のメソッド呼び出し文を抽出し、学習プロセスと同様に、メソッド呼び出し文の並びを作成する。この並びを学習済みモデルに適用すると、指定した場所に挿入できる可能性の高いメソッド呼び出し文の候補が出力され、開発者に提示を行う。開発者は提示された候補を見て、どのメソッド呼び出し文を選ぶか判断する。開発者が挿入したい場所を決めて、出力された候補から選択する具体的な流れは図 1 に示している。

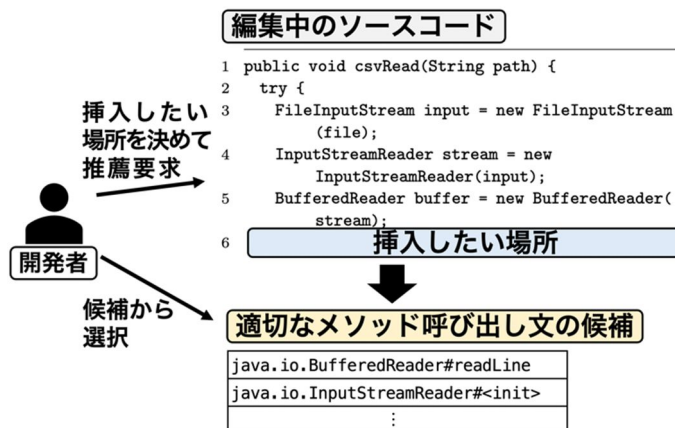


図 1 推薦の流れ

さらに、ソフトウェアの品質を確保するために不具合の含まれる可能性の高いソースコードを予測する手法を提案した。そのための手法として、深層学習を用いてソースコード内に含まれる不具合を予測した。本研究では、実験のため Git リポジトリのオープンソースソフトウェアからソースコード片を収集し、それらに対して深層学習を用いて不具合が含まれるか否かを予測した。事前学習ありのモデルはソースコード片の不具合予測において有効であること、事前学習に使用されたプログラミング言語のソースコード片においても有効であることを示した。

4. 研究成果

メソッド呼び出し文の順序を RNNLM に学習させた手法と精度の比較を行った。比較をするために、オープンソースソフトウェアのプロジェクトの一部

(Android-Universal-Image-Loader, butterknife, fresco, glide, MPAndroidChart, SlidingMenu, ViewPagerIndicator) を評価用として用いた。

これらのプロジェクトはGitHubに存在するものである。それら以外に用いられていたプロジェクトは現在取得できなかったため除外した。

評価方法は、後ろにメソッド呼び出し文がない場所で推薦させたときの方法で行った。

結果を表 1 に示す。表の列の手法の MLM は本手法であり、RNNLM は既存手法である。

推薦候補 1 位 (Top-1) の精度はほとんどのプロジェクトにおいて、本手法が少し高い結果となった。一方、推薦候補 10 位 (Top-10) は RNNLM の手法の方が高い結果が多かった。

本研究のモデルの語彙数 (メソッド呼び出し文の種類) が多いことに影響していると考えられる。ただ、推薦候補 1 位の精度は少し高くなっていることから、Masked Language Model の能力の高さが表れていると考える。

表 1 他手法との比較結果

評価プロジェクト	手法	Top-1	Top-5	Top-10
Android-Universal-Image-Loader	MLM	0.39	0.66	0.75
	RNNLM	0.33	0.80	0.87
MPAndroidChart	MLM	0.35	0.64	0.76
	RNNLM	0.33	0.68	0.86
SlidingMenu	MLM	0.51	0.74	0.80
	RNNLM	0.29	0.67	0.85
ViewPagerIndicator	MLM	0.59	0.83	0.93
	RNNLM	0.22	0.48	0.70
butterknife	MLM	0.31	0.52	0.57
	RNNLM	0.30	0.45	0.60
fresco	MLM	0.35	0.59	0.71
	RNNLM	0.20	0.59	0.80
glide	MLM	0.31	0.57	0.64
	RNNLM	0.20	0.76	0.89

5. 主な発表論文等

〔雑誌論文〕 計2件（うち査読付論文 2件 / うち国際共著 0件 / うちオープンアクセス 0件）

1. 著者名 小林 歩夢、山本 哲男	4. 巻 65
2. 論文標題 Masked Language Modelを利用したAPI推薦手法	5. 発行年 2024年
3. 雑誌名 情報処理学会論文誌	6. 最初と最後の頁 221 ~ 229
掲載論文のDOI（デジタルオブジェクト識別子） 10.20729/00231747	査読の有無 有
オープンアクセス オープンアクセスではない、又はオープンアクセスが困難	国際共著 -

1. 著者名 河方 健悟、山本 哲男	4. 巻 64
2. 論文標題 事前学習モデルを利用したソースコード片の不具合予測	5. 発行年 2023年
3. 雑誌名 情報処理学会論文誌	6. 最初と最後の頁 1659 ~ 1667
掲載論文のDOI（デジタルオブジェクト識別子） 10.20729/00231445	査読の有無 有
オープンアクセス オープンアクセスではない、又はオープンアクセスが困難	国際共著 -

〔学会発表〕 計2件（うち招待講演 0件 / うち国際学会 0件）

1. 発表者名 田中駿，山本哲男
2. 発表標題 直前のトークンの種類情報を利用したソースコード生成の試み
3. 学会等名 電子情報通信学会，信学技報，vol. 122，no. 432，SS2022-70，pp. 139-144
4. 発表年 2022年

1. 発表者名 各務仁一朗，山本哲男
2. 発表標題 メソッドの制御構造や識別子に着目したメソッド名推薦
3. 学会等名 電子情報通信学会，信学技報，vol. 123，no. 414，SS2023-68，pp. 115-120
4. 発表年 2023年

〔図書〕 計0件

〔産業財産権〕

〔その他〕

-

6. 研究組織

	氏名 (ローマ字氏名) (研究者番号)	所属研究機関・部局・職 (機関番号)	備考
--	---------------------------	-----------------------	----

7. 科研費を使用して開催した国際研究集会

〔国際研究集会〕 計0件

8. 本研究に関連して実施した国際共同研究の実施状況

共同研究相手国	相手方研究機関
---------	---------