様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

令和　５ 年　 ５ 月 １８ 日現在

機関番号：１２６０１
研究種目：挑戦的研究（萌芽）
研究期間：2019～2022
課題番号：１９Ｋ２２８４０

研究課題名（和文）機械学習によるプログラミング言語文法の推定

研究課題名（英文）Extracting syntactical structures in programs by using machine learning

研究代表者
千葉　滋（Chiba, Shigeru）

東京大学・大学院情報理工学系研究科・教授

研究者番号：８０２８２７１３

交付決定額（研究期間全体）：（直接経費）　　5,000,000円

研究成果の概要（和文）：近年、大きく進歩した機械学習に基づく言語モデルによるプログラミング支援の研究をおこなった。言語モデルを用いて、プログラミング言語がもつ広い意味での「文法構造」を抽出し、それを実用的なプログラミング支援に応用する研究をおこなった。具体的には、ライブラリを利用したプログラミングに現れる、メソッドの呼び出し順というある種の文法や、異なるプログラミング言語に共通する文法構造、プログラムのモジュール分割に見られるある種の文法構造、を抽出し応用に活かす手法の研究をおこなった。

研究成果の学術的意義や社会的意義
現在、言語モデルは大規模化することで、本研究が用いていたような小規模なモデルでは達成できなかったような精度を実現できている。本研究は、モデル自体の研究ではなく、モデルの機能を実用的なプログラミング支援に活かす方法を探る研究であった。本研究の成果は、大規模化した言語モデルと組み合わせることで、さらなる性能向上が期待でき、現実世界のプログラミングの支援技術の向上に貢献できると考えられる。これは社会基盤として安心安全なＩＴシステムの実現の一助になる技術である。

研究成果の概要（英文）：We conducted research on programming supports by using language models based on machine learning, which is significantly advanced in these years. By utilizing language models, we extract the broad sense of "syntactical structures" in programming languages, and we apply them to practical programming supports. Specifically, we focused on extracting and applying certain syntactical structures of the order of method calls, which is found in programming using a library, syntactical structures common to different programming languages, and certain syntactical structures observed in modularization for programs.

研究分野：プログラミング言語、ソフトウェア工学

キーワード：プログラミング言語　ソフトウェア工学　機械学習

様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９（共通）

１．研究開始当初の背景

当時、深層機械学習の技術が世界中で活発に研究された結果、当初注目されていた画像認識の分野だけでなく、自然言語処理の分野にも大きな進歩をもたらしていた。その結果、英語や日本語など様々な言語の間の機械翻訳が実用化されてきており、その後の生成ＡＩの登場につながっていく時期であった。一方で、プログラミングの分野でも深層機械学習の技術が利用され始めており、自然言語の分野での文章入力の際の予測変換や入力補完の技術を応用して、プログラミングの最中に次にタイプすべきプログラムの候補を提示する技術などが実用化されつつあった。またプログラム中の関数やメソッドなどの意味を要約した文章を自動生成する機能などもさかんに研究されていた。

２．研究の目的

プログラミングの分野でも深層機械学習の技術の応用研究が始まっていたものの、それらの研究はプログラミングの分野における英語や日本語など自然言語の扱いに深層機械学習の技術を応用しようとするものが中心で、プログラミング言語独自の性質に注目した研究はあまりないように思われた。プログラミング言語は言語と呼ばれるが、自然言語と異なって深い入れ子構造をもつ人工言語である。このため浅い入れ子構造しかもたない自然言語の技術をそのままプログラミング言語に適用してもうまくいかず、プログラミング言語に特徴的な文法構造に着目した独自の研究がおこなえるのではないかと思われた。

この着想にそって、GitHub 等の公開レポジトリから集めた大量のコーパスを学習させることで、プログラミング言語における何らかの文法構造を推定し、それを利用した実用的な応用例を発見することを目的に研究を進めた。ここでいうプログラミング言語の文法とは、BNF のような、いわゆる通常の形式文法で定められる文法ではなく、より広い意味での新しい「文法」である。プログラミング言語のコーディング規約や、ライブラリ関数の呼び出し順、あるいは異なるプログラミング言語にまたがった共通的な文法などである。大規模なプログラムの場合、モジュール分割が必要不可欠だが、このモジュール分割の規則のようなものも、ここでいう「文法」に含めて研究する。

３．研究の方法

要素技術を研究開発して、それを応用につなげていく、という研究手法はとらず、まず具体的な応用例、実用的な課題を発見し、その課題を解決する手法を研究しながら、汎用的な要素技術の研究開発を試みる、という手法で研究をおこなった。また特定のプログラミング言語を選んで集中的に研究をおこなうということもせず、応用例に合わせてプログラミング言語を選び、研究をおこなっていった。具体的には次のような応用例、実用課題を軸に研究をすすめた。

（１）動的プログラミング言語においてメソッド・チェインを入力している際のコード補完

オブジェクト指向プログラミング言語のライブラリは、近年、メソッド・チェインと呼ばれる形式でライブラリ・メソッドを提供するものが増えている。メソッド・チェインとは、同一のオブジェクトに対して、いくつかのメソッドを連続して次々と呼び出す形式である。Java 言語のような静的型付け言語では、型情報を利用してエディタが次に呼び出すメソッドの候補を提示するコード補完の機能を提供するのが普通である。しかし、JavaScript 言語のような動的型付け言語の場合、エディタが型情報を利用できないので、十分に高い精度で候補を提示できない。しかしながら、ライブラリが提供するメソッド・チェインの場合、ある種の領域特化言語のプログラムと見なせるので、一定の文法構造なり規則性があるはずである。機械学習によってこれを抽出し、候補の精度を改善することを試みる。

（２）異言語にまたがるコード・クローンの検出

近年の web サービスの開発では、複数のプログラミング言語を使い分けながら一つのソフトウェアを開発することが一般的になりつつある。大きなプログラムともなると、内容が類似した重複コード（コード・クローン）が発生しがちなので、それを発見し、リファクタリングで重複を解消するか、重複コードとして維持しつつ管理下におき、修正があった場合は重複コード全てが正しく修正されるように管理する必要がある。同一言語のプログラム内の重複コードの発見手法はさかんに研究されてきたが、異なる言語で書かれたプログラム内の重複コードを発見する効率的な手法は明らかになっていない。機械学習を用いて効率よく重複コードを発見する手法の研究をおこなう。

（３）不適切なモジュールに属するメソッドの検出

大規模なプログラムでは、プログラムをいくつかのモジュールに分割して全体の保守性や拡張性を高める手法が必須である。しかしながら、ソフトウェア開発者が、どのようなモジュール分割が適切であるか判断できるようになるためには、知識や経験が必要である。このため開発チームの中の上級者が、チームメンバのプログラムを検査して、モジュール分割が適切であるか判断するのが一般的であるが、チームの中の上級者の人数は少なく、無視できない負担となっていた。そこでモジュール分割の適切さを自動判定する機械学習によるモデルを研究開発する。

４．研究成果

それぞれの課題について研究をおこなった。

（１）動的プログラミング言語においてメソッド・チェインを入力している際のコード補完

JavaScript 言語のプログラムを対象に GitHub より 546 レポジトリのソースプログラムを取得し、その中で使われているメソッド・チェインを抽出してデータセットを開発した。メソッド・チェインは可読性のためにしばしば複数の式に分割されるが、静的解析により分割前の長いチェインを抽出した。このデータセットを用いて言語モデル LSTM を教師ありで訓練し、途中までチェインを入力した段階で次のメソッド呼び出しを補完候補として予測できるかの実験をおこなった。モデルへの入力トークンを作り出す際に、メソッド名をどのようにエンコードするかは課題の一つである。自然言語と異なり、プログラムに現れるメソッド名は任意の文字列が許されるので、単語数は非常に多くなり、自然言語で用いられている手法をそのまま適用することはできない。我々はメソッド名を分割し、自然言語のコーパスを学習して得たモデルを使って、その分散表現を得る手法を試みた。残念ながら、この研究は途中で同様の研究成果が海外の他グループにより論文発表されてしまったため、途中で打ち切り、得られた研究成果を他の課題に適用することとなった。

（２）異言語にまたがるコード・クローンの検出

当時注目され始めた教師なし学習の手法を異言語コード・クローン検出に適用した。まずプログラム学習サイト LeetCode の問題の解答として書かれた Java と Python のプログラムを GitHub から集めてデータセットを開発した。開発したデータセットは 103 個の Java と Python のプログラムのペアからなる。これらを組み合わせた 103×103=10,609 個のペアの中に 103 個のペアがあるというデータセットである。このデータセットの中のプログラムの各単語（トークン）に Byte Pair Encoding を適用することで語彙数を 157,347 個から 30,829 個へ圧縮した。LSTM や Transformer など異なる encoder を試し、最終的には F1 スコアで 0.84 の検出性能を得た。

この研究では当初、モデルへの入力は通常の自然言語処理と同様、プログラムを単語列に分解したものであった。しかしながら、これではプログラミング言語に特徴的な文法構造をうまく認識できるとはいいがたい。そこでプログラムを構文木に変換し、Code2Vec で採用された構文木の中のパスを用いる手法で入力列を作りモデルに与える方法を試みた。この手法では機械学習を用いた従来手法よりもかなり優れた異言語コード・クローンの検出性能を得ることができた。現在知られている中で検出性能が最も優れた手法は機械学習を用いない手法であるが、この研究で開発した手法は最も優れた手法に近い性能を示した。最も優れた手法は計算時間が非常に長いのが欠点であるが、この研究で開発した手法は非常に短い時間で計算できるという利点を持つ。この研究の成果は現在、論文発表を目指して投稿中である。

（３）不適切なモジュールに属するメソッドの検出

Java 言語のプログラムを対象として、few-shot 分類の機械学習モデルを用いて、与えられたメソッドを適切なパッケージ（Java 言語におけるモジュールの単位）に分類し、そのメソッドが誤ったパッケージに属する場合に警告を発する手法を研究開発した。メソッドをどのようなパッケージに分類すべきかの基準はプログラムごとに異なる。適切な分類の仕方は、そのプログラムが採用しているアーキテクチャや、開発しているチームの考え方に依存するからである。このため、うまくパッケージ分類されていると思われるプログラムを大量に集めて既存の機械学習モデルに学習させるだけでは、適切な分類は難しい。この研究では、few-shot 分類用として知られる prototypical network をモデルとして用いることで、与えられたプログラムにおけるパッケージ分類の基準に沿った基準でメソッドが属すべきパッケージを示すモデルを開発した。この研究により、few-shot 分類用のモデルがプログラミング言語の分野でも有用であることを示せた。本研究では学習データとして、オープンソースとして公開されており、かつ人気が高いプログラムを用いた。そのようなプログラムはパッケージ分類も適切におこなわれており、学習データとして適切と考えたからである。本研究の成果は査読付きの国際会議論文として発表した。

５．主な発表論文等

〔雑誌論文〕　計1件（うち査読付論文　1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 Yoda Kazuki、Nakamaru Tomoki、Akiyama Soramichi、Chiba Shigeru	４．巻 N/A
２．論文標題 An Anomaly-Based Approach for Detecting Modularity Violations on Method Placement	５．発行年 2022年
３．雑誌名 Proc. of the 22nd IEEE International Conference on Software Quality, Reliability, and Security (QRS 2022)	６．最初と最後の頁 287-298
掲載論文のDOI（デジタルオブジェクト識別子） 10.1109/QRS57517.2022.00038	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5件（うち招待講演　0件／うち国際学会　0件）

１．発表者名 依田 和樹，中丸 智貴，穐山 空道，山崎 徹郎，千葉 滋
２．発表標題 Java システムにおけるパッケージ誤りのニューラルネットワークを用いた検出手法
３．学会等名 日本ソフトウェア科学会第38回大会
４．発表年 2021年

１．発表者名 Feng Dai, Shigeru Chiba
２．発表標題 Attempts on using syntax trees to improve programming language translation quality by machine learning
３．学会等名 38th JSSST Conference
４．発表年 2021年

１．発表者名 白石 誠，千葉 滋
２．発表標題 コード内にコメントを入れる時に適切なコメントを例示するシステムの開発
３．学会等名 日本ソフトウェア科学会第38回大会
４．発表年 2021年

１．発表者名 劉 宇澤，千葉 滋
２．発表標題 教師ラベルなし単言語学習データのみでのcross-languageコードクローン検出の試み
３．学会等名 日本ソフトウェア科学会第37回大会
４．発表年 2020年

１．発表者名 松永 智將，千葉 滋
２．発表標題 機械学習手法を用いた動的型付け言語のコード補完に向けて
３．学会等名 日本ソフトウェア科学会第36回大会
４．発表年 2019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