令和 6 年度卒業研究発表会（日本大学工学部情報工学科）
A-3-18

学生が提出した C 言語プログラムの深層学習を用いた解析手法の調査

Investigation of analysis methods using deep learning for programs submitted by students

20216115　達川　航充　[関澤研究室]

1　はじめに

大学の情報教育では,様々なコンピュータに関する知識を講義や演習を通して学ぶことになり,その中にプログラミングも含まれている．プログラミングは，実際に自分でプログラムを組む実践的な取り組みにより,プログラミングに関する理解を深める.その過程で教員から課題が与えられその解答プログラムを提出することになる.教員はこれらを評価していくことで,学生のプログラミング理解度を把握することできる.

しかし，教員が受け持つ学生数は非常に多いため，学生のプログラムを一つ一つ手作業で評価する方法は膨大な時間を要することになる．また，もう一つの手法として学生の提出プログラムの出力が正しいかを機械的に評価をする自動採点手法[1]があり，この手法は採点者の負担を減らせる．しかし，出力結果のみを評価しプログラムの内容を評価することができない．そのため，この手法では学生の正確な理解度を把握することができない.それに加え,手作業による採点ではないので採点者から各学生へのフィードバックを得ることができない.

本研究では,自動採点手法によって学生のプログラムを評価することで採点者の負担を軽減すると同時に各学生のプログラムの理解度を正確に把握することができる手法を確立することを最終目標とする．そして，その目標達成のための一歩目として,プログラムの解析手法についての調査を行う．この時，深層学習を利用する手法について着目する．解析手法の調査から始める理由は，数多くの学生が作成するプログラムは異なったものであり,それぞれの内容について機械的かつ正確に把握する手段を見つける必要があると考えたためである．また，高度な自動化を狙うだけでなく深層学習によって蓄積された膨大な学習データを学生へのフィードバックに役立つことも狙っている.

2　調査

2.1　既存の自動採点手法

自動採点手法には出力結果のみを比較してプログラムを評価するだけではなく,ソースコード同士を比べて評価をする手法もある.

その一つとして,課題ごとに設定されている評価軸ごとにチェック項目を作り,項目ごとに解答となる構文木を作成する．その後，学生のプログラムも同様に構文木を作成しそれら二つを比較することで同一構造があった場合に正解とする手法[2]である．この手法は編集距離という方法を用いることでプログラムごとの違いの度合いを求めることができ,同一構造でない場合も採点することができる.さらに,この手法を発展させたものがある.採点の際，特定の関数がどこで呼び出されているかが重要になることもある．例えば,「入力された数字の合計を表示せよ．0 の入力で終了とする」という問題では，入力が反復処理の中で行われているかどうかを見る必要がある.その時に構文木のノード中にある Call に呼び出し関数を付け加えることで対応をする．また反復処理を必要とする問題で for 文や while 文の指定がない場合にどちらの処理を使った場合でも構造に問題がない場合には正解とするといった柔軟な対応を必要とする.こういった問題を解決する手法[3]があり,この手法によってより制御構造の理解度を正確に測ることができる.

もう一つの手法として,類似度関数を用いて二つの抽象構文木(abstract syntax tree:AST)の一致判定を行い，完全一致なら１,完全不一致なら０を出力することにより０～１の類似度を算出するといった手法でプログラムの評価を行う[4]．この手法ではテキスト上の同じ意味を持つ別の言葉などゆらぎの影響を受けない類似度関数を用いている.

2.2　深層学習

深層学習(Deep Learning)は人間の脳をモデルにしたネットワークを活用し,コンピュータが膨大なデータを処理する方法である．人間のサポートを必要とせず，データの法則性や特徴を見つけ出し,段階的に自動で学習することで精度の高い結果を導き出す.

深層学習のモデルはニューラルネットワークアーキテクチャに基づいている．ニューラルネットワーク(Neural Network)は層構造内で相互接続されたノード,すなわちニューロンで構成されており,これらのニューロンが入力を望ましい出力に関連付ける.入力層と出力層の間のニューロンは，隠れ層と呼ばれており「Deep」はこの層の数を指し,深層学習のモデルでは数百･数千にもなる場合がある.層をより深くすることにより複雑な問題に対処する.大規模な一連のラベルデータを使用して学習が行われることで，データから特徴量を直接学習できる場合が多くある.そのため，深層学習には高い計算能力が求められる．クラスターまたはクラウドコンピューティングを組み合わせることにより学習時間を数週間から数時間以内に削減できるようになる.

深層学習の精度はデータ量に比例するので,パフォーマンスは学習データのサイズが大きくなるにつれて改善されていく．したがって，深層学習ではモデル学習のために極めて膨大なデータが必要となる.

2.3　code2vec

code2vec は ALON らによって提案されたプログラムコードの分散表現を得るための手法[5]である．得られた分散表現をもとに任意のメソッド本体からメソッド名の提案及び推定やプログラム中の必要な依存関係の予測といったタスクをこなすことが可能になる．プログラムから AST を作成し,その AST 中の一つの終端記号とそれを繋ぐ非終端記号を連結して一つのパスコンテキストとし特徴量として抽出する．この特徴量はメソッド名の予測に使われ,特徴量の値と近いものが候補に挙げられる.以下の三つの手順によって分散表現を得ることができる.

1. プログラムコードからパスコンテキストを抽出
2. パスコンテキストの埋め込みを行い分散表現のパスを作成
3. 注意重みを使って分散表現の加重平均を行い集約

これらからクロスエントロピー損失を使用して学習を行う.この学習によって得られたコードベクトルはコサイン距離で最も近いものは意味的に似た名前を含む.またベク

トル同士の加減算によって，新たなベクトルを作ることができる．

ここで code2vec のデモサイト[6]を参考に使用例を示す．図 1 は配列の並びを反転するコードであり，これに code2vec を使用すると表 1 で示す結果を得る．図 1 のコードの名前として適切な「reverseArray」を高い確率で予測している．「reverse」という文字はコード中に使用されていないが，予測の候補として挙げられている．

```
String[] f(final String[] array) {
        final String[] newArray = new String[array.length];
        for (int index = 0; index < array.length; index++)  {
                newArray[array.length - index - 1] = array[index];
        }
        return newArray;
}
```

図 1．プログラムコード例

表 1．コードに適した関数名の予測

メソッド名	reverseArray	reverse	subArray	copyArray	doResize
予測（%）	77.34	18.18	1.45	0.74	0.70

しかし，これらを実現するために code2vec では比較的大規模なデータが必要となる．code2vec のモデルでは終端ノードの値が似た意味でも違う名称であれば，別物として扱われる．同様にして，AST や予測したいラベル名に異なる部分や揺らぎがあると別のものとして扱われる．これは学習において効果を弱める原因となっている．また，変数名に依存していることや新生のラベルに対応できないといった学習した者からしか予測できないこと，学習するデータが正確なものであることが前提となっていることが code2vec のデメリットとして挙げられる．それに加え，code2vec は Java や C#といったオブジェクト指向言語を対象としている．C 言語を適用する場合には Clang と LLVM によって AST 中の終端記号すべてを抽出し，それぞれを繋ぐ非終端記号や終端記号をパスとする．これを特徴量とする．また，TF-IDF によって，オブジェクト指向言語のようにモジュール固有名と一般的な操作名に関数名を分解する．子の関数名と特徴量を用いて code2vec にて学習する手法[7]である．

3　考察

既存の自動採点手法では，構文木を利用することにより正確に学生のプログラミング理解度を測る手法を確立していた．しかし，解答プログラムの作成や課題の評価ポイントとなる部分の設定，提案した手法を用いても採点しきれない部分について目視確認の必要性など，人間の手から完全に離れることができなかった．しかし，code2vec であれば，ソースコード同士の比較をより低次元のベクトル同士の比較に落とし込むことができる．これにより，自動採点によって負担を減らしながらプログラムの内容を評価することができ，正確な採点を行うことができる．また，学生プログラムの学習によってより高精度な採点を行うことができ，将来的には人間の手を借りずとも採点者の手作業による採点とほぼ同等による採点を行うことが予想される．このことから，深層学習を利用して学生が提出したプログラムの解析する手法を確立していく中で，ツールとして code2vec を利用することは効果的であると考えられる．

code2vec をツールとして利用しない方法として既存の自動採点手法の評価軸や使用されている関数を特徴量や特徴量抽出に利用する方法も考えられる．code2vec や深層学習を通して既存の手法を昇華させより高度に自動化された自動採点手法になることも予想される．

しかし，深層学習を利用する場合には膨大で大規模なデータが必要となり，それに伴いそれらを処理するための高度な計算が行える高性能な CPU が必須となる．それらがない場合には予測精度の低い学習しか行えず，人間の介入なしに自立的な評価を行えるまでに膨大な時間を要することとなり，かえって採点者への負担を増やすことも考えられる．そのため，既存の自動採点手法と code2vec を使ったプログラムの評価を実践し，これら二つの評価の精度について調べる必要があると考える．

4　今後の課題・まとめ

本研究では学生の提出プログラムの深層学習を利用した解析手法の確立を目的として，既存研究と深層学習を利用する場合に考えられる自動採点ツールとして有用な code2vec について調査を行った．今後の課題として，既存研究の手法と code2vec を利用した場合の手法の比較と評価を行うことだけでなく，prog2vec[8]などの code2vec 以外の深層学習ツールとして利用することができる手法のプログラムの評価の精度を調査する必要がある．それに加え，プログラムの採点を行った後に採点によって得られた学習データをどのように役立たせていくかを考えていく必要があり，学生に与える最終的なフィードバックに大いに役立てるよう学習させる手法を確立していく事も視野に入れなければならない．

参考文献

[1] 倉田英和，富永浩之，林敏浩，垂水浩幸．実行テストによるプログラム判定を用いた初級 C プログラミング演習支援と授業実践．情報処理学会研究報告コンピュータと教育(2007-CE-091), pp.11-18 (2007)

[2] 小山昂紘，原田史子，島川博光．階層構造化による C 言語ソースコードの自動採点，情報科学技術フォーラム講演論文集, 2012, 11, 3, p.595-596

[3] 漆原宏丞，本多佑希，岸本有生，兼宗進．抽象構文木を利用したプログラミング理解度採点の試み．情報処理学会研究報告, Vol.2021-CE-162, No.16, pp.1-6, 2021

[4] 小川弘迪，小林亜樹．プログラミング課題の自動採点に向 けた構文木上のカーネル法による類似度関数の提案．第 80 回全国大会講演論文集, 2018, 2018, 1, pp.661-662

[5] Uri Alon, Meital Zilberstein, Omer Levy, Eran Yahav. code2vec: learning distributed representations of code. Proceedings of the ACM on Programming Languages, Vol.3, No.40, p.1-29, 2019

[6] Guy Waldman. “code2vec：learning distributed representations of code”. code2vec. https://code2vec.org. (参照 2025-02-02)

[7] 檜枝琴里，久住憲嗣，矢川博文，福田晃. code2vec for C：C 言語を対象としたコードの分散表現の獲得手法の提案．研究報告組込みシステム（2019-EMB-51）, 4, p.1-2, 2019

[8] 竹末裕，竹内和広．プログラミング問題における多様な回答の可視化．情報教育シンポジウム論文集, 2019, p.326-330, 2019