GPTによるプログラミング初学者の誤答に適した正解とそれを利用した効果的なヒントの生成

Generation of suitable correct answers and effective hints using GPT for wrong answers of novice programmers

山品 壱平[*1], 杉谷 賢一[*1], 中野 裕司[*1], 久保田 真一郎[*1], 渡邊 健太[*1], 市原 大裕[*1], 田村祐貴[*1]
Ippei YAMASHINA[*1], Kenichi SUGITANI[*1], Yuji NAKANO[*1], Shin-ichiro KUBOTA[*1],
Kenta WATANABE[*1], Daisuke ICHIHARA[*1], Yuki TAMURA[*1]
[*1]熊本大学
[*1]Kumamoto University
Email: 208t3265@st.kumamoto-u.ac.jp

あらまし：プログラミング初学者にとってエラーが発生した場合の対処は非常に困難なものである．しかし，学生にエラーを修正したコードを提供するだけでは，学生の理解を促すことはできない．本研究では，学生がプログラムエラーを解決する際に役立つヒントを生成することを目的とする．これまで，学生の作成したバグを含むプログラムデータをもとに実験を行い，プログラミング初学者がエラーを修正するためのヒント生成手法の検討を行った．
キーワード：GPT，Retrieval Augmented Generation，プログラミング教育

1. 背景・目的

プログラミング初学者にとってエラーが発生した場合の対処は非常に困難なものである．しかし，学生に答えをそのまま提示するだけでは，学生のプログラミングに対する理解を促すことはできない．そのため，プログラムのエラーで躓いている学生に対して，答えではなく有益なヒントを提示することを目的とする．

また，本研究で提案するモデルは本大学の情報基礎Bという科目で使用することを想定する．情報基礎Bの提出物にはWebページ作成課題が存在する．Webページを作成する際にはhtml，cssなど複数のファイルを使用することになるため，エラー発生時にどのファイルにバグがあるかを見つけ出すのは初学者にとって難しいものとなる．

2. 先行研究

PhungらはOpenAIが提供する大規模言語モデルであるGPT-4，及びGPT-3.5を用いたプログラミング問題に対するヒント生成モデルであるGPT4Hints-GPT3.5Val を提案した[1]．最初のステップとして，私たちの技術はヒントを生成するための「教師」モデルとして GPT-4 を利用する．失敗したテスト ケースの出力情報とプロンプト内の修正を使用することで，生成の品質を高める． 次のステップとして，GPT-4より弱いモデルである GPT-3.5 を「学生」モデルとして活用し，ヒントの品質をさらに検証する．

3. 提案手法

今回使用するヒント生成プロセスはPhungらの提案モデルを参照にしてStage-1， Stage-2， Stage-3の3つの段階の処理を行い，ヒントを生成する．先行研究はPythonの問題をもとに提案されたモデルだが，今回は本大学の情報基礎Bという科目で使用することを想定している． そのため，先行研究とは違い，GPT-4，GPT-3.5を使用する際にRetreiverを実装する．

Retrieverとは言語モデルの知識を補完するために関連する文書を外部の文書集合から検索し，言語モデルに入力として与える技術である[2]．Retrieverを実装し，GPTの出力の際に情報基礎Bのテキストを参照できるようにすることで，適切なヒントを提示できるか検証する．

3.1 Stage-1

学生の質問Qとバグを含むプログラムP_bを入力に加え，GPT-4に対してバグを修正したプログラムを出力させる．また，ここではP_bのみを入力に加えた場合の修正プログラムを出力させる．それぞれ30個ずつプログラムを生成する．次に60個の生成プログラムとP_bとの編集距離を求め，編集距離が0を除いて最小のものを修正されたプログラムP_f として取得する．

3.2 Stage-2

P_bとP_fを入力に加え，バグに関する詳細な説明Xとヒント Hを出力させる．ここで，Hだけでなく Xを出力させるのは Chain-of-Thoughtというプロンプトエンジニアリングを参照している[3]．

3.3 Stage-3

2つのgpt-3.5-turboを用いた学生モデルを比較する．学生モデル1ではQ，P_bを入力に加え，バグを修正したプログラムを出力させる． 一方，学生モデル2ではQ，P_bとバグの詳細な説明Xを入力に加える． それぞれ30回出力させ，修正精度を比較する．

4. 実験

4.1 実験手法

本研究の評価実験はStage-1，Stage-2，Stage-3の3つの処理に分けて実験を行う．使用するデータは学

生の質問Q，2つのプログラムファイルP_bをセットとして7つの学生データを用意する．使用するプログラムデータの一部を図1に示す．

Stage-1では取得する修正されたプログラムP_fがバグを修正したプログラムであるかを確認する．

Stage-2では生成されるバグの詳細な説明Xが，プログラムのバグを特定し，適切な修正案が提示されているかを確認する．入力として加える修正されたプログラムP_fは，P_bを手作業で最低限の修正を行ったものを扱う．また，Stage-2では10回出力させる．

最後にStage-3では，バグに関する説明Xを入力に加えない学生モデル1とXを入力に加える学生モデル2の修正精度を比較する．入力に加えるXはStage-2で生成されたヒントの中で最も適切であるものを扱う．

```
/*ヘッダー部分*/
/*ヘッダー部分に背景と文字色を設定して横幅いっぱいに表示*/
.header{
  background-color: #336699;
  color: white;
  width: 100%;/*フッタの幅*/
}
/*文章部分*/
/*文章部分に背景と文字色を設定して横幅いっぱいに表示*/
.article{
  background-color: #D7E6EF;
  color: black;
  width: 100%;
}
/*フッター部分*/
/*フッター部分に背景と文字色を設定して横幅いっぱいに表示*/
.footer{
  background-color: #fd0;
  color: black;
  width:100%
}
```

図1. 学生Aのプログラムデータ(css)

4.2 実験結果

4.2.1 Stage-1

実行結果を確認した7つの学生データの中で、学生Bを除いた6つの学生データがバグを修正したプログラムを出力していた．

また，編集距離を求めることで取得する修正されたプログラムP_fは少なくとも1つのバグを修正したプログラムを取得していた．

4.2.2 Stage-2

生成されたバグの詳細な説明Xについて，ほとんどのプログラムデータでプログラム内のバグを特定し，修正案を説明していた．しかし，学生Bのプログラムデータはバグを特定することができず，誤った修正案を提示していた．

4.2.3 Stage-3

ここでは学生Bのプログラムデータは，Stage-2で適切なXとHが出力されなかったため，実行結果は確認しない．学生Cと学生Fのプログラムデータを除いたプログラムデータはバグの詳細な説明Xを入力に加える学生モデル2の方が学生モデル1よりも修正精度が高かった．しかし，学生Cと学生Fのプログラムデータは学生モデル2の修正精度が学生モデル1の修正精度を下回った．

5. 考察

Stage-1では学生B以外のプログラムのバグを修正することができていた．学生Bのプログラムのバグはhtmlに全角スペースがあることで発生している．また，このバグはプログラム実行時にエラーコードを出力するわけではないため，バグを特定することが難しいと考えられる．また，Stage-2で全角スペースを修正したプログラムを入力として加えたが，正しいバグの原因と修正案を述べていなかった．そのため，GPT-4にとって全角スペースによって発生するバグを解決することが難しいと考えられる．

Stage-2では学生Dのデータに関して，バグの詳細な説明Xは修正する必要のない部分をバグとして扱い，修正案を出力していた．この修正案はjsファイルに簡略化できる部分があったため，出力されたと考えられる．この問題は学生に必要以上のヒントを提示することにつながり，学生を混乱させてしまう．

Stage-3では学生Cと学生Fのデータで，バグの詳細な説明Xを入力に加える学生モデル2が学生モデル1の修正精度を下回っていた．2つの学生データで共通することは，バグがhtmlとJavaScriptで共有する値が一致していないことである．2つの学生データに関するXの修正案は正しいものであったのにも関わらず，学生モデル2の修正精度は低いものとなっていた．このことから，プログラムのバグが複数のプログラムで共有するidによるものである場合，修正案を入力に加える学生モデル2の修正精度が下がっていしまうことが分かった．

6. 結論・展望

Stage-1のほとんどのプログラムデータで，出力されるプログラムP_fがバグを修正していたことが確認できた。Stage-2では，いくつかのプログラムデータで実際には修正する必要のない箇所をバグとして扱い修正案を提示していた．Stage-3ではいくつかのプログラムデータで，バグの詳細な説明Xを入力に加えた学生モデル2の修正精度が下がる場合があった．

今後の方針として，結論に述べたStage-2，Stage-3の問題を解決することを目指す．また，Stage-2で生成されたヒントの人的評価を行う予定である。

参考文献

(1) Tung Phung, Victor-Alexandru Pădurean, Anjali Singh, Christopher Brooks, José Cambronero, Sumit Gulwani, Adish Singla, Gustavo Soares. Automating Human Tutor-Style Programming Feedback: Leveraging GPT-4 Tutor Model for Hint Generation and GPT-3.5 Student Model for Hint Validation. https://arxiv.org/abs/2310.03780

(2) https://python.langchain.com/docs/modules/data_connection/retrievers/

(3) Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In NeurIPS, 2022.