

Retrieval-Based Prompt Selection for Code-Related Few-Shot Learning

Noor Nashid

University of British Columbia
Vancouver, Canada
nashid@ece.ubc.ca

Mifta Sintaha

University of British Columbia
Vancouver, Canada
msintaha@ece.ubc.ca

Ali Mesbah

University of British Columbia
Vancouver, Canada
amesbah@ece.ubc.ca

2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE), pp. 2450-2462

発表者：松尾 和輝

背景 | プロンプト学習

- 自然言語による指示, タスクの例, などのテキスト (プロンプト) を入力として学習する手法
 - 特に, 複数の例を与えるケースをFew-Shot学習という^[15]
 - プロンプトを用いることで, 大量のデータ収集やfine-tuningを行うことなく, LLMの性能を高めることができる
 - プロンプトの内容によって出力が大きく変わる
 - プロンプトの内容は千差万別

課題

効果的なプロンプトを作成することは手間

[15] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pretrain, prompt, and predict: A systematic survey of prompting methods in natural language processing," arXiv:2107.13586, 2021.

目的・アプローチ

- 目的

- 効果的なプロンプトを構築するための技術CEDARの提案

※本研究ではコード関連タスクに特化したプロンプトを構築

アサーション生成とプログラム修正

- アプローチ

- 埋め込みや頻度分析に基づく検索ベースの技術を利用
 - 大量のコード例から，類似する数個の例を選択

CEDAR | 手続きの流れ

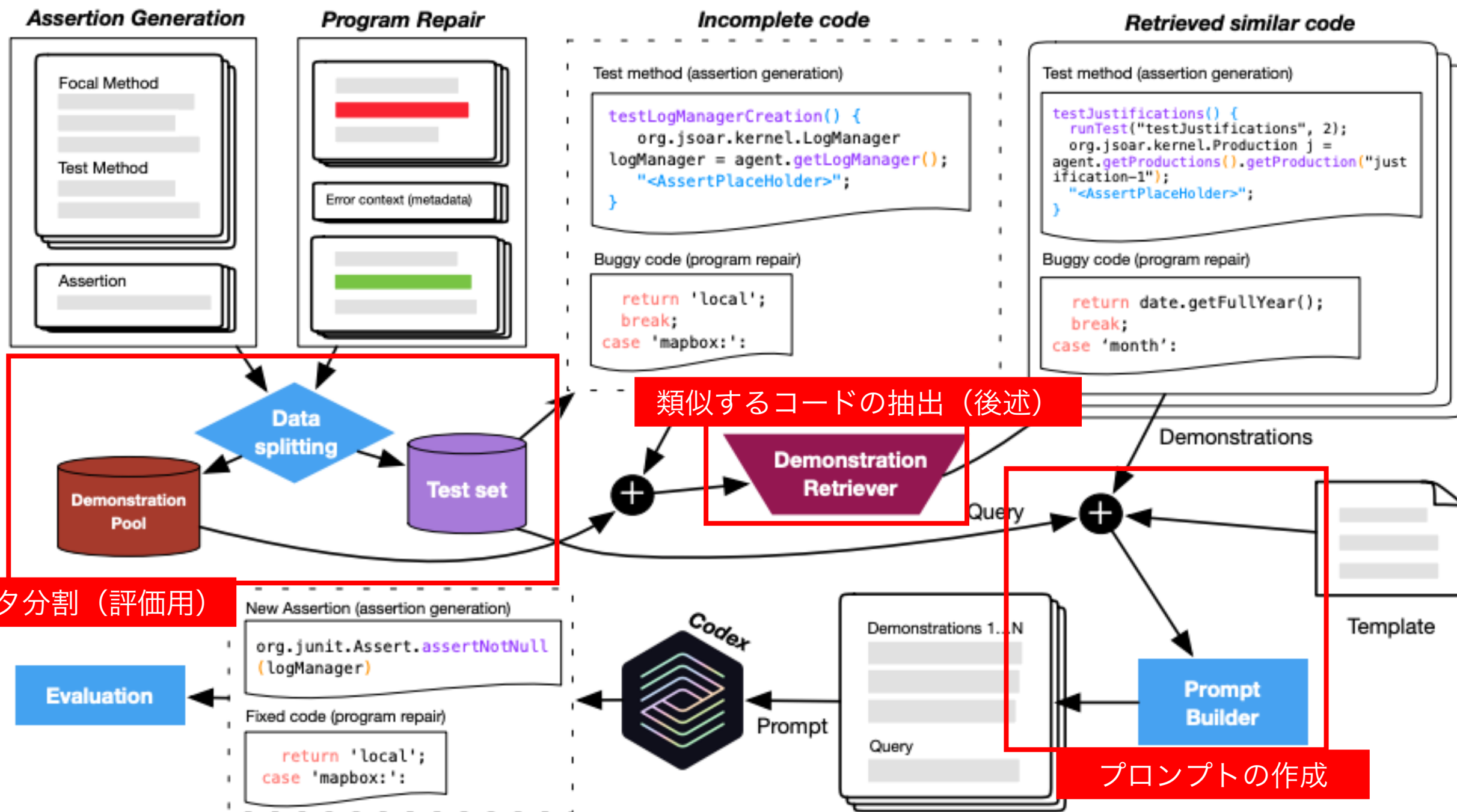


Fig. 1: Overview of our approach CEDAR.

CEDAR | 作成されるプロンプト

- アサーション生成の例

Input

```
### METHOD_UNDER_TEST:
getProduction(java.lang.String) {
    return productionsByName.get(name);
}

### UNIT_TEST
testJustifications() {
    runTest("testJustifications", 2);
    org.jsoar.kernel.Production j = agent.getProductions()
        .getProduction("justification-1");
    "<AssertPlaceholder>";
}

[METHOD_UNDER_TEST]: getProduction
[UNIT_TEST]: testJustifications

### generate assertion
org.junit.Assert.assertNull(j)
END_OF_DEMO
```

Code Demonstrations

Demonstration 1

Demonstration 2

...

Demonstration N

Query

```
### METHOD_UNDER_TEST:
getLogManager() {
    return logManager;
}

### UNIT_TEST
testLogManagerCreation() {
    org.jsoar.kernel.LogManager logManager =
        agent.getLogManager();
    "<AssertPlaceholder>";
}

[METHOD_UNDER_TEST]: getLogManager
[UNIT_TEST]: testLogManagerCreation

### generate assertion
```

Focal Method
Test Method
Context
Instruction

Output

```
org.junit.Assert.assertNotNull(logManager)
```

CEDAR | 作成されるプロンプト

- プログラム修正の例

Input

Code Demonstrations

```
### Fix ESLint error in the following JavaScript code:
### Buggy JavaScript
  return date.getFullYear();
  break;
case 'month':

"rule_id" : no-unreachable
"evidence": Unreachable code.
"warning_line": break

### Fixed JavaScript
  return date.getFullYear();
case 'month':
```

END_OF_DEMO

Demonstration 1

Demonstration 2

...

Demonstration N

Query

Instruction

```
### Fix ESLint error in the following JavaScript code:
### Buggy JavaScript
  return 'local';
  break;
case 'mapbox:':

"rule_id" : no-unreachable
"evidence": Unreachable code.
"warning_line": break;

### Fixed JavaScript
```

Error Snippet

Error Context

Context

Output

```
  return 'local';
case 'mapbox:':
```

CEDAR | 類似するコードの抽出

- ①埋め込み (SRoBERTa^[32])
 - 文変換モデルSRoBERTaを用いてコードスニペットをベクトル化
 - コサイン類似度を用いて、最も近い上位N個のコードスニペットを、demonstration poolから検索
- ②頻度 (BM-25^[31])
 - TF-IDF^[34]を拡張したスパース検索手法BM-25を使用
 - 各テストサンプルについて、最も類似した関連スコアを持つ例を検索

※スパース検索手法：重要な情報だけを用いた検索手法（多分）

[31] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: Bm25 and beyond," Found. Trends Inf. Retr., vol. 3, no. 4, p. 333–389, 2009.

[32] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). ACL, 2019, pp. 3982–3992.

[34] J. Ramos et al., "Using tf-idf to determine word relevance in document queries," in Proceedings of the first instructional conference on machine learning. Citeseer, 2003, pp. 29–48.

評価 | リサーチクエスチョン

- RQ1: 検索ベースのプロンプト作成はどの程度有効か?
 - ランダム手法との比較
- RQ2: CEDARの精度は最先端のモデルと比較してどうか?
 - fine-tuningモデルとの比較

評価 | データセット

- ATLASデータセット^[8]（アサーション生成）
 - アサーションを生成するためのコンテキスト（フォーカルメソッド・テストメソッド）が含まれる
 - assertEquals, assertTrue, assertNull, assertNotNull, assertThat, assertFalse, assertEquals, assertEqualsの8つのカテゴリのアサーションが含まれる
 - 訓練：150,523, 評価：18,815
- TFIXデータセット^[39]（プログラム修復）
 - バグのあるコードスニペット（通常は2～4行）と、エラータイプ、メッセージ、警告行に関する情報を提供する52種類のESLintからのエラーコンテキストが含まれる
 - 訓練：84,892, 検証：9,432, 評価：10,480

[8] C. Watson, M. Tufano, K. Moran, G. Bavota, and D. Poshyvanyk, "On learning meaningful assert statements for unit test cases," in ACM/IEEE 42nd International Conference on Software Engineering. ACM, 2020, p. 1398–1409.
[39] B. Berabi, J. He, V. Raychev, and M. T. Vechev, "Tfix: Learning to fix coding errors with a text-to-text transformer," in ICML, 2021, pp. 780–791.

評価 | 評価指標

- 完全一致精度 (EM Acc.) [%]
- 妥当な一致精度 (PM Acc.) [%]
- 最長共通部分列 (LCS) [%]
- 編集距離 (ED.)
- アサーションメソッドの一致 (AMM.) [%]
- 予測時間[sec.]

結果 | RQ1: プロンプトの有効性

- どちらのタスクにおいても, ランダム手法よりも CEDARの方が精度高
- どちらのタスクにおいても, 埋め込みよりも頻度の方が精度高

Task	Strategy	Types	Examples	EM Acc.(%)	PM Acc.(%)	LCS(%)	ED.	AMM.(%)	Inf. time(s)
Assertion Generation	Random	Zero-shot	0	0.00	14.42	15.34	36.62	29.16	1.33
		One-shot	1	44.41	47.64	64.97	57.83	70.24	3.64
		8-shot/category	8	50.33	53.69	62.65	17.94	74.48	2.54
		8-shot	8	50.57	53.34	68.77	19.10	74.12	2.41
		20-shot	20	49.99	52.73	74.64	20.53	73.80	3.16
	CEDAR	Embedding	7	75.79	77.40	88.50	11.35	88.49	1.78
		Frequency	6	76.55	78.32	88.99	13.18	89.31	2.20
	Random	Zero-shot	0	27.33	32.57	68.02	98.30	-	3.35
		One-shot	1	28.54	37.57	67.93	87.13	-	3.95
		52-shot/category	52	33.47	44.10	67.05	23.34	-	3.83
		52-shot	52	34.65	45.02	67.64	23.71	-	3.86
		60-shot	60	39.88	49.80	68.22	23.09	-	3.85
Program Repair	CEDAR	Embedding	48	46.12	55.63	73.99	21.47	-	3.78
		Frequency	41	51.72	60.55	75.57	32.97	-	4.22

結果 | RQ1: プロンプトの有効性

- 自然言語による指示の有無の影響
 - どちらのタスクにおいても、あまり違いは見られない

Task	Template	Examples	EM Acc.(%)	PM Acc.(%)	LCS(%)	ED.	AMM.(%)	Inf. time(s)
Assertion Generation	w/o NL instructions	6	76.53	78.27	89.00	13.68	89.37	2.95
	w/NL instructions	6	76.55	78.32	88.99	13.18	89.31	2.20
Program Repair	w/o NL instructions	53	49.31	58.33	74.41	36.25	-	3.27
	w/NL instructions	41	51.72	60.55	75.57	32.97	-	4.22

RQ1の回答

ランダム手法よりもCEDARの方が精度が高く、検索ベースの手法は有効と言える

自然言語の存在はあまり精度に影響を与えない

結果 | RQ2 : 最新技術との比較

Task	Approach	Model type	Exact Match Acc. (%)
Assertion Generation	ATLAS	Task-specific	17.66
	Mastropaolo et al. [9]	Fine-tuned	57.60
	Mastropaolo et al. [10]	Fine-tuned	68.93
	CEDAR	Few-shot learner	76.55
Program Repair	HOPPITY	Task-specific	7.90
	CoCoNuT	Task-specific	11.70
	SEQUENCER	Task-specific	17.90
	TFIX	Fine-tuned	49.30
	CEDAR	Few-shot learner	51.72

RQ2の回答

両タスクにおいて, CEDARはSOTAモデルを凌駕している

[9] A. Mastropaolo, S. Scalabrino, N. Cooper, D. N. Palacio, D. Poshyvanyk, R. Oliveto, and G. Bavota, "Studying the usage of text-to-text transfer transformer to support code-related tasks," in 43rd International Conference on Software Engineering (ICSE). IEEE, 2021, pp. 336–347.

[10] A. Mastropaolo, N. Cooper, D. N. Palacio, S. Scalabrino, D. Poshyvanyk, R. Oliveto, and G. Bavota, "Using transfer learning for coderelated tasks," IEEE Transactions on Software Engineering, pp. 1–20, 2022.

まとめ

- 目的
 - 効果的なプロンプトを構築するための技術CEDARの提案
- 結果
 - 入力データと類似するデータをプロンプトの例として与えることでモデルのパフォーマンスを向上させることが可能
 - CEDARで出力されるプロンプトを用いることで、既存手法よりも高い精度が得られる

所感

- CEDARの手続きがいまいちよくわからない
 - Data splittingでは何をしているの？
 - Incomplete Codeはどうやって作っているの？自動？
 - Templateとは？
- 評価指標の計算方法もよくわからない
 - 妥当な一致精度とは？