

オープンソースLLMによる ログステートメント 生成手法の評価実験

谷元 柊介

ログとは

システムが記録する動作履歴をログという

- ログは、デバッガが使えないような本番環境での不具合原因特定に有用[1].
- システムが動作を記録することそのものをロギングという

[1] Yuan, D., Park, S., & Zhou, Y. (2012). Characterizing logging practices in open-source software. 2012 34th International Conference on Software Engineering (ICSE), 102-112.

ログステートメント

プログラムの実行中にロギングを行うために、ソースコード中に記述する命令文をログステートメントという

- JavaのAPIであるLog4jやSlf4j, Pythonの標準モジュールであるloggingなどがログステートメントを提供している

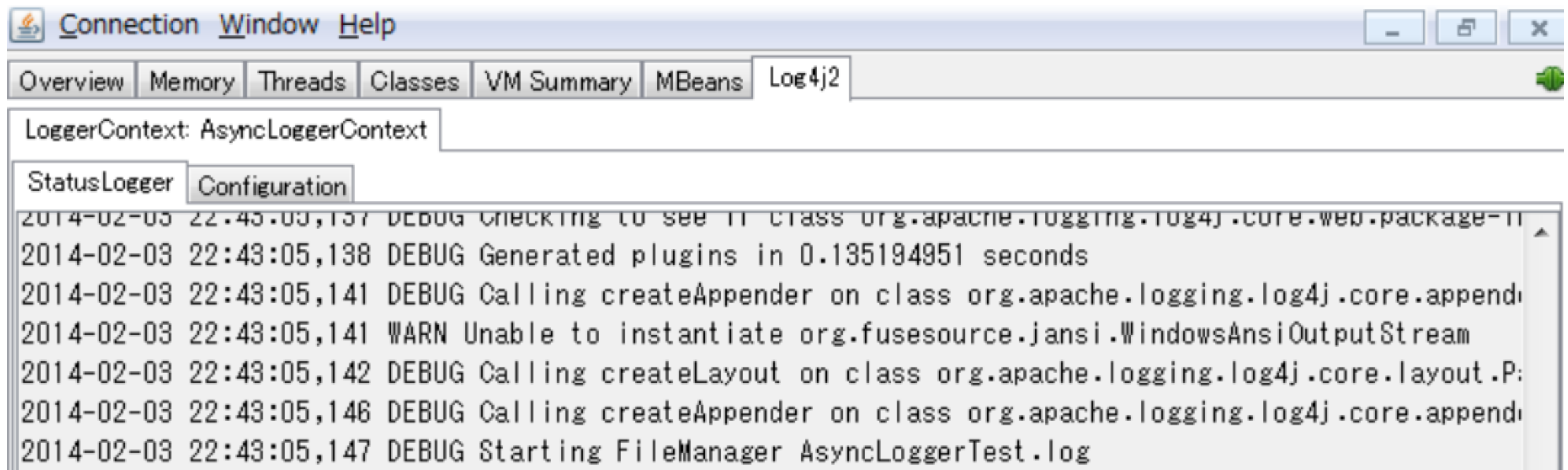


図 1 : Log4jのログステートメントにより出力されるログの例

ログステートメントとログの対応関係(Log4j)

ログステートメントの構成要素として、
ログレベル、ログ変数、ログテキスト
の3つが挙げられる

```
logger.debug("Debugging user: username=" + username + ", id=" + userId);
```

Log4jのログステートメント

ログレベル
(そのログの重要度
を示す)

ログテキスト

```
2025-05-26 10:30:00 [DEBUG] Main - Debugging user: username=vanilla, id=42
```

上記のログステートメントにより出力されるログ

※usernameとuserIdは ログ変数

自動ログステートメント生成の必要性

手動でのログステートメント記述の課題

- 開発者の経験に依存
- プロジェクト内での一貫性不足
- 適切な場所・内容の判断が困難

自動化による利点

- 開発効率の向上
- 一貫したログスタイル
- 人的エラーの削減

オープンソースLLMによるコード生成 手法の必要性

ChatGPTやClaudeのようなLLMは、ユーザーの入力データを自身の学習に使う場合がある

機密情報保護の観点等により、企業によってはオープンソースLLMにファインチューニングを施して運用している場合がある

したがって、オープンソースLLMによるソフトウェア開発タスクの効率化の研究が求められる

既存の自動ログ生成手法：LEONID

オープンソースLLMによるログステートメント生成アプローチの1つとして、LEONIDが挙げられる

- T5-smallモデルに、事前学習とファインチューニングを施したもの
- 入力：ログステートメントを挿入したいJavaメソッド
- 出力：ログステートメントが挿入されたJavaメソッド
- 事前訓練データセット：1,465のJavaプロジェクトから抽出した約690万のJavaメソッドに加工を施したもの
- ファインチューニング：
 - ログステートメントを抜いたメソッドに対して、元々入っていたログステートメントの予測を行わせる
 - 予測に際して、k個の類似メソッドのログステートメントがヒントとして与えられる(k=1, 3, 5)

LEONIDのログステートメント生成能力

LEONID(k=5)における予測成功率は27.26%

- kはファインチューニングにおいて使われる類似メソッドの数
 - 多ければ多いほど性能がよくなる
- 予測の成功は, Variable, Level, Message, Positonすべてにおいて, 正解データと完全一致していることを指す

RQ₁: Correct and partially correct predictions by LEONID and LANCE on the single-log injection task.

Variable	Level	Message	Position	LEONID (k = 5)	LANCE	p-value	OR
✓	✓	✓	✓	27.26%	26.78%	<0.01	1.12
✓	—	—	—	76.45%	77.15%	<0.01	0.88
—	✓	—	—	73.53%	74.18%	<0.01	0.91
—	—	✓	—	31.55%	30.16%	<0.01	1.36
—	—	—	✓	82.35%	82.28%	0.71	1.01

既存研究の評価法についての課題

現在行われているログステートメント生成の研究は、すべて単一メソッドに対するログステートメントの生成のみを行っている

これらの研究の評価では、**実際に出力されたログによる評価を行っていない**

- ソースコード上に存在するログステートメント同士でのみ比較している

実際に出力されるログを観察

➤ 既存研究で発見されなかった新たな課題が見つかる可能性

研究目的

研究目的：既存研究手法(LEONID)の未発見課題の発見

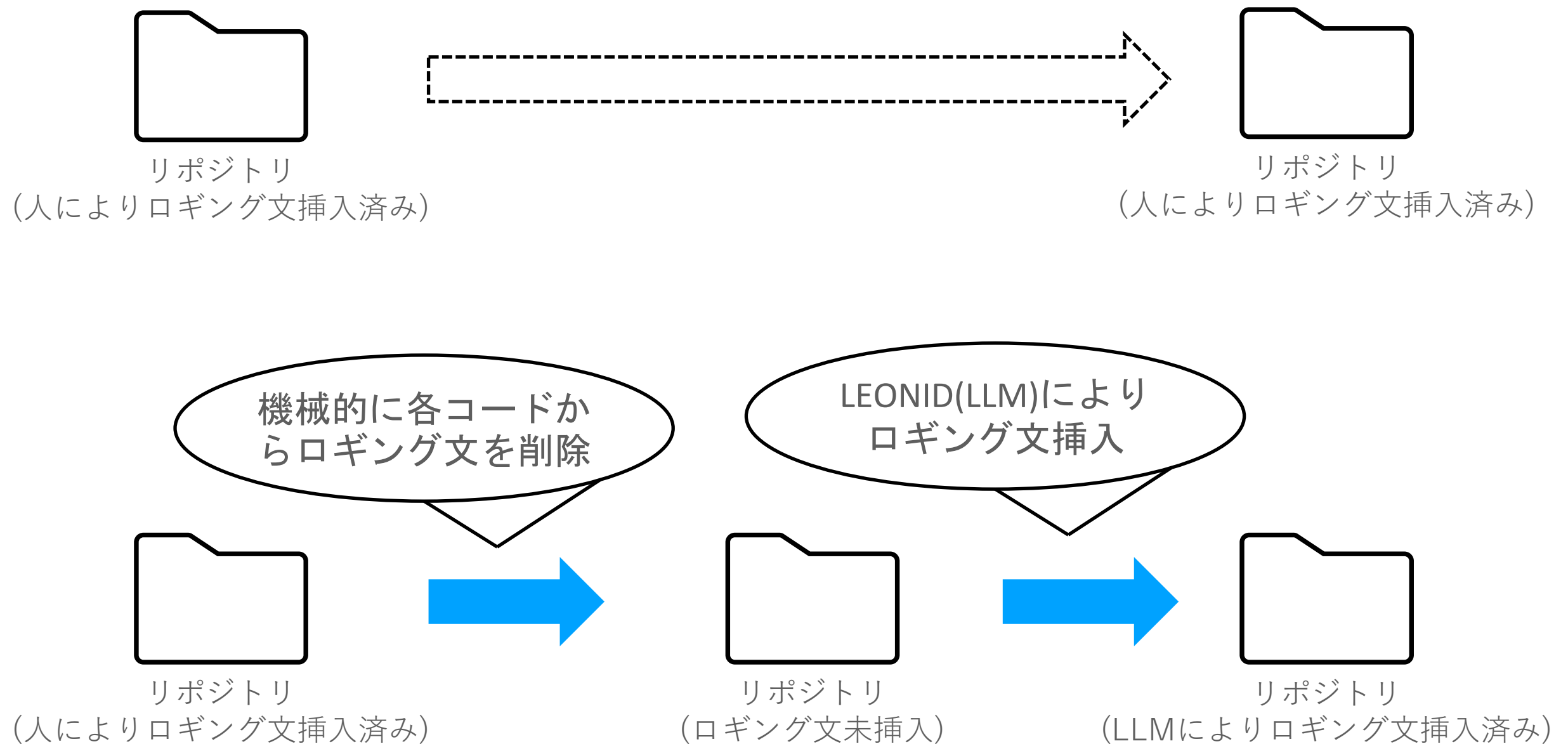
アプローチ：

- オープンソースプロジェクトに対して，LEONIDによるログステートメント生成を適用
- 実際にログを出力
- 出力されたログの観察

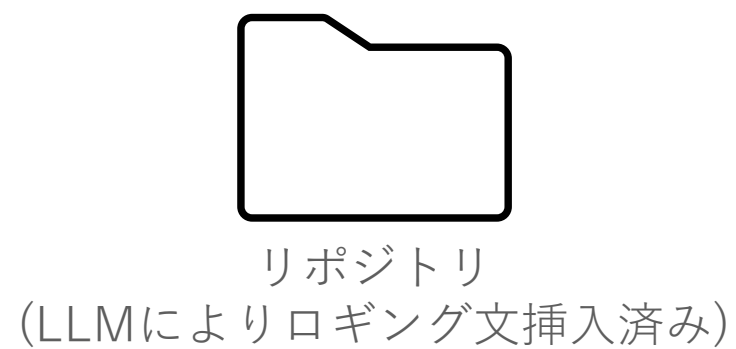
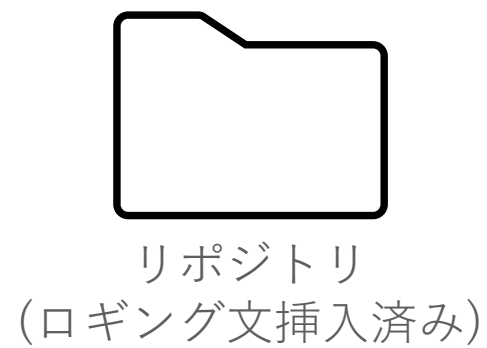
評価に用いるデータ， 評価指標について

| 検討中

実験概要(1/2)



実験概要(2/2)



実験について . . .

まだできていない

- 技術的な問題に直面している
 - オーバーライドされているメソッドに対する処理
- 仮にそこが解消されたとして、必ずいくつか構文的におかしいものが生成されるので、それへの対処をしないといけない
 - 構文エラーが出ることはLEONIDのベースとなったモデルであるLANCEの論文でも言われていた
- 「実用的な使い方を課題を見つける」以前に「**実用的な使い方をするための課題**」に先に取り組むべきなのかもしれない
 - 当たり前だが、既存研究のように単一メソッドへの生成ばかり考えていても実用的な使い方にはつながらない
 - つまり、既存研究は実用性をあまり考慮していないと言える