

How Much Do Code Language Models Remember? An Investigation on Data Extraction Attacks before and after Fine-tuning

Fabio Salerno

Delft University of Technology
Delft, The Netherlands
f.salerno@tudelft.nl

Ali Al-Kaswan

Delft University of Technology
Delft, The Netherlands
a.al-kaswan@tudelft.nl

Maliheh Izadi

Delft University of Technology
Delft, The Netherlands
m.izadi@tudelft.nl

背景

コードLLMの訓練データ（事前学習）

- インターネット上から収集された**未精製**のソースコードを利用

精製とは？

- ・機密情報（APIキー、パスワード、メールアドレス、名前など）の除去
- ・ライセンスのフィルタリング
- ・品質フィルタリング（極端に短い・長いファイルの除去など）

- 訓練データの内容をデータ抽出攻撃によって**再現できる**ことが判明^[4]

機密情報を抽出できる可能性がある

先行研究

記憶：訓練データの内容を再現すること

事前学習されたLLMの“記憶”に関する研究が行われている

- ・どのような内容が記憶されやすいか^[27]
- ・どのようなプロンプトであれば抽出されやすいか^[29] などを調査

[4] N. Carlini, F. Tramer, E. Wallace, et al., “Extracting training data from large language models,” in 30th USENIX Security Symposium (USENIX Security 21), 2021, pp. 2633–2650.

[27] N. Kandpal, E. Wallace, and C. Raffel, “Deduplicating training data mitigates privacy risks in language models,” in International Conference on Machine Learning, PMLR, 2022, pp. 10 697–10 707.

[29] N. Carlini, D. Ippolito, M. Jagielski, K. Lee, F. Tramer, and C. Zhang, “Quantifying memorization across neural language models,” preprint arXiv:2202.07646, 2022.

ファインチューニングでの影響

事前学習時と同様な問題

- 企業・個人が自社の機密コードや独自データでモデルをFT
 - └─ これらのデータを抽出できる可能性がある
- モデルを公開したり，APIとして提供した時に問題となる
- 先行研究
 - FT手法の違いによる影響を調査^[11]
(Full FT vs LoRAの比較など)
 - FTタスクの違いによる影響を調査^[12]
(要約 vs 翻訳の比較など)

コードLLMでは調査されていない

[11] F. Miresghallah, A. Uniyal, T. Wang, D. Evans, and T. Berg-Kirkpatrick, “Memorization in nlp fine-tuning methods,” in First Workshop on Pre-training: Perspectives, Pitfalls, and Paths Forward at ICML 2022, 2022.

[12] S. Zeng, Y. Li, J. Ren, et al., “Exploring memorization in fine-tuned language models,” arXiv preprint arXiv:2310.06714, 2023

本研究の目的・RQs

目的

コードLLMが事前学習データとFTデータをどの程度記憶し、抽出攻撃に対してどれだけ脆弱であるかを調査

RQs

- 1, 事前学習データはどの程度抽出可能か
- 2, FT後のモデルから事前学習データはどの程度抽出可能か
- 3, FTデータはどの程度抽出可能か
- 4, 特定のデータカテゴリがFT後により抽出されやすいか

調査方法 | RQ 1 : 事前学習データはどの程度抽出可能か



サンプル例

```
package com.example.project;

import java.util.List;
import java.util.ArrayList;

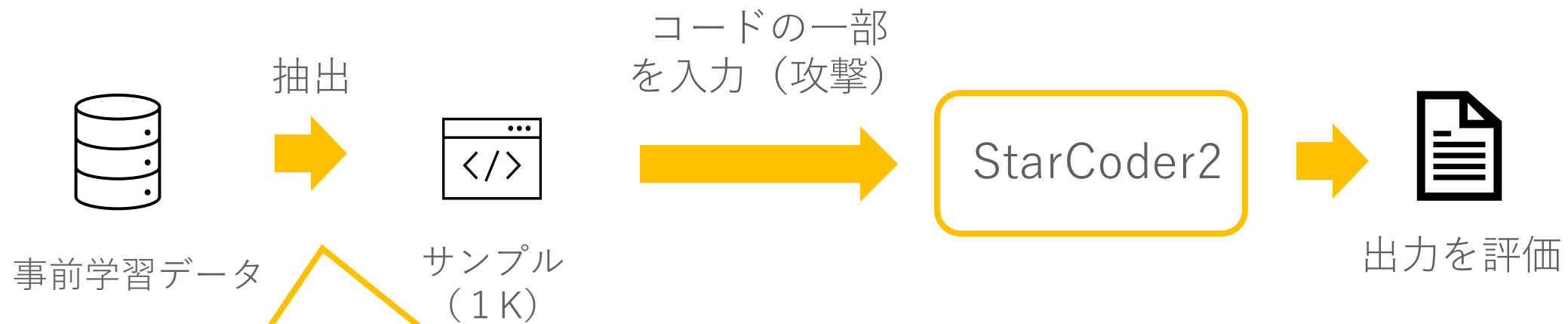
public class DataProcessor {
    private List<String> data;
    public DataProcessor() {
        this.data = new ArrayList<>();
    }
    public void addData(String item) {
        if (item != null && !item.isEmpty()) {
            data.add(item);
        }
    }
}
```

250トークン (可変)
入力として利用
(プレフィックス)

50トークン (可変)
抽出対象 (正解)
(サフィックス)

300トークン
(固定)

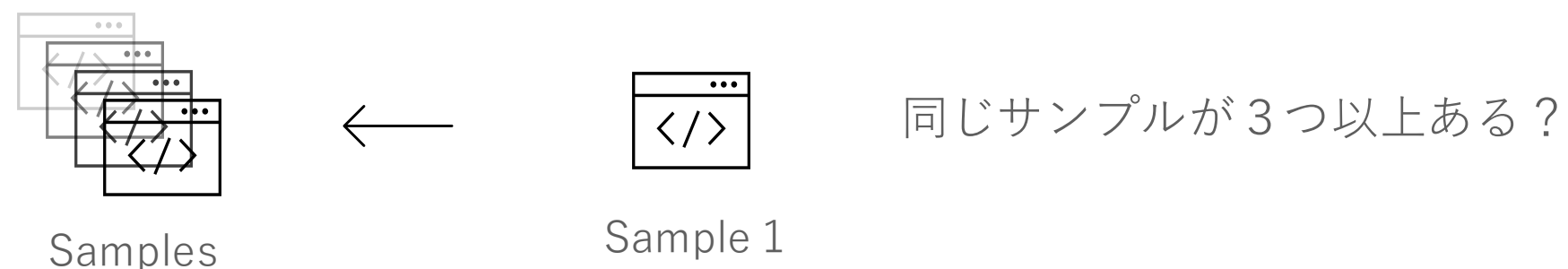
調査方法 | RQ 1 : 事前学習データはどの程度抽出可能か



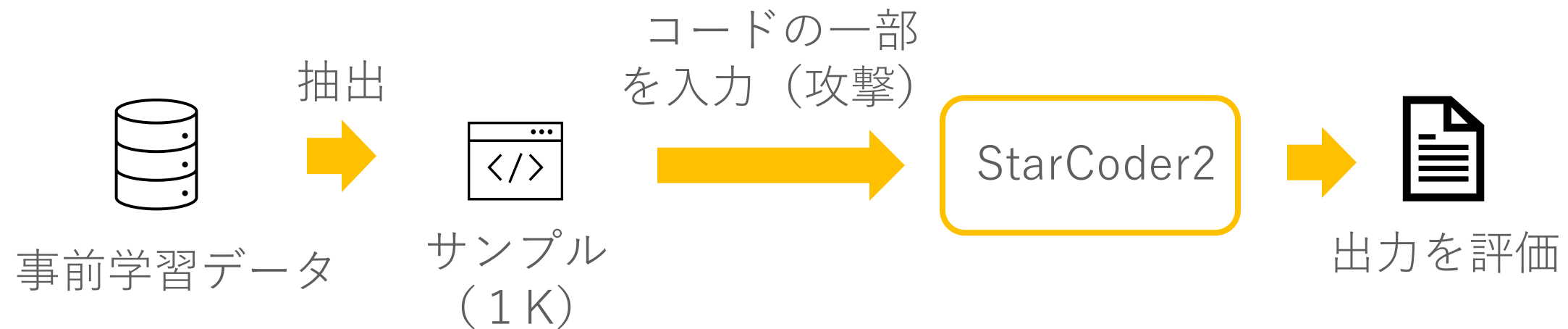
(step1) 1 ファイルごとに300トークンをランダムに抽出



(step 2) 重複数が3以下のサンプルをフィルタリング



調査方法 | RQ 1 : 事前学習データはどの程度抽出可能か



実験設定

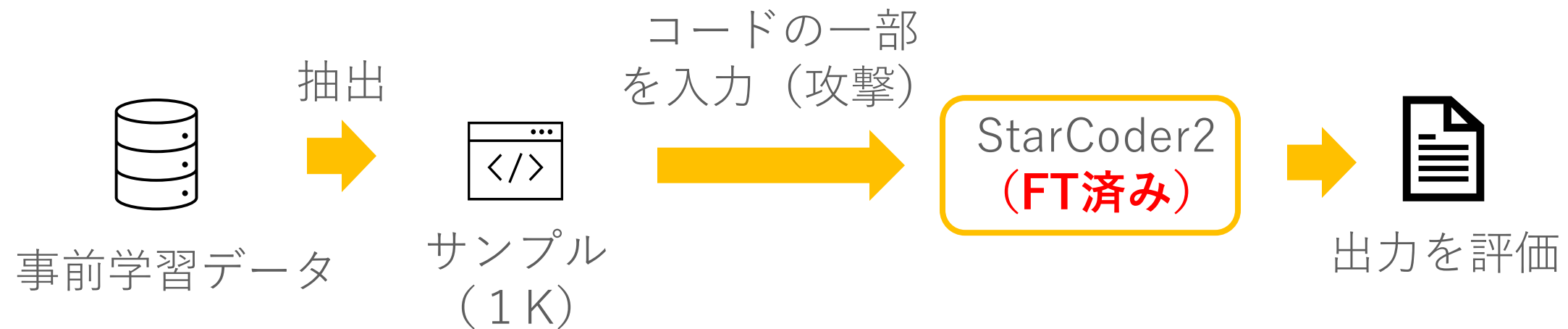
- ・ モデルサイズ (3B, 7B, 15B)
- ・ 入力プレフィックス (100, 150, 200, 250 トークン)

評価指標

Exact Match, BLEU, METEOR, ROUGE-L

調査方法

RQ2：FT後のモデルから事前学習データはどの程度抽出可能か



FTデータセット：The Heap

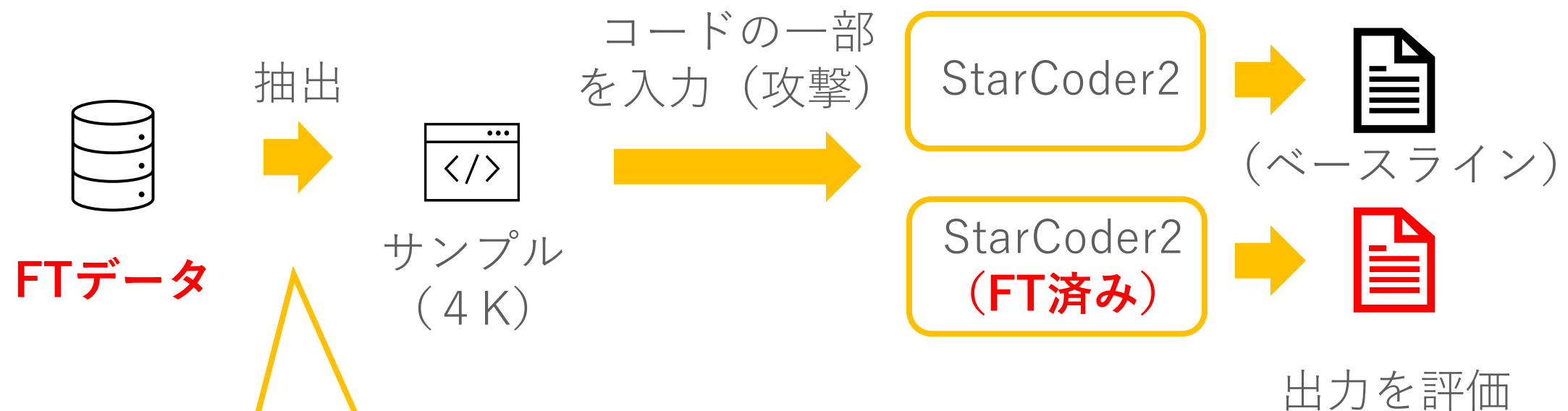
言語：Java

サイズ：20kファイル

事前学習には含まれていないコード（強いコピーレフトのため）

その他実験設定（サンプル抽出，モデルサイズ等）はRQ1と同様

調査方法 | RQ3: FTデータはどの程度抽出可能か



(step 1) スライディングウィンドウで300トークンずつに分割
[1 ~ 300], [50 ~ 350], [100 ~ 400]

(step 2) 重複率ごとに1Kずつランダムに抽出

重複率 $d = 1, 2, 3, >3$

結果 | RQ 1 : 事前学習データはどの程度抽出可能か

プレフィックス長に比例

モデルサイズに比例

Model	Prefix length	Memorization rate			
		EM	BLEU	METEOR	ROUGE-L
Starcoder2-3B	100 tokens	0.389	0.652	0.706	0.682
	150 tokens	0.432	0.688	0.735	0.712
	200 tokens	0.462	0.715	0.756	0.737
	250 tokens	0.483	0.728	0.770	0.751
Starcoder2-7B	100 tokens	0.395	0.663	0.720	0.694
	150 tokens	0.452	0.701	0.756	0.733
	200 tokens	0.485	0.730	0.772	0.753
	250 tokens	0.498	0.742	0.781	0.763
Starcoder2-15B	100 tokens	0.455	0.714	0.760	0.738
	150 tokens	0.506	0.757	0.770	0.774
	200 tokens	0.536	0.777	0.806	0.794
	250 tokens	0.549	0.790	0.819	0.805

まとめ

- ・モデルは事前学習データを記憶している
- ・モデルサイズが大きいほど記憶しやすい
- ・プレフィックス長が長いほど抽出しやすい

結果

RQ2：FT後のモデルから事前学習データはどの程度抽出可能か

プレフィックス長を100に固定

Model	Epochs	Memorization rate			
		EM	BLEU	METEOR	ROUGE-L
Starcoder2-3B	0	0.389	0.652	0.706	0.682
	1	0.087	0.448	0.551	0.517
	2	0.108	0.452	0.552	0.518
	3	0.105	0.447	0.551	0.514
Starcoder2-7B	0	0.395	0.663	0.712	0.694
	1	0.015	0.381	0.524	0.483
	2	0.110	0.460	0.566	0.527
	3	0.116	0.470	0.572	0.533
Starcoder2-15B	0	0.455	0.714	0.760	0.738
	1	0.145	0.525	0.613	0.583
	2	0.143	0.530	0.619	0.585
	3	0.155	0.544	0.629	0.595

FTによって抽出可能なデータが減少

まとめ

- 抽出可能だったデータの約30%が抽出不可に
- モデルサイズに関係なく同じ傾向
- プレフィックス長の効果は限定的

結果 | RQ3：FTデータはどの程度抽出可能か

エポック数・重複数の影響

モデル	エポック数	EM	モデル	重複数	EM
Starcoder2-3B	0	0.263	Starcoder2-3B	=1	0.049
	1	0.459		=2	0.237
	2	0.595		=3	0.451
	3	0.631		>3	0.631

- ・ エポック数を増やすとより多くのサンプルが記憶される
- ・ 重複率が多いとより抽出しやすい

結果 | RQ3: FTデータはどの程度抽出可能か

プレフィックス長の影響

プレフィックス長に比例

Model	Prefix length	Memorization rate			
		EM	BLEU	EM↑%	BLEU↑%
Starcode2-3B	100 tokens	0.631	0.764	0.367	0.226
	150 tokens	0.689	0.785	0.393	0.231
	200 tokens	0.747	0.816	0.443	0.260
	250 tokens	0.815	0.865	0.502	0.297
Starcode2-7B	100 tokens	0.578	0.750	0.330	0.202
	150 tokens	0.646	0.773	0.386	0.220
	200 tokens	0.712	0.802	0.436	0.243
	250 tokens	0.800	0.850	0.492	0.245
Starcode2-15B	100 tokens	0.610	0.775	0.310	0.198
	150 tokens	0.730	0.805	0.372	0.216
	200 tokens	0.817	0.830	0.441	0.232
	250 tokens	0.888	0.868	0.474	0.244

まとめ

- 重複率/エポック数/プレフィックス長に比例して抽出されやすくなる

結果 | RQ4：特定のデータカテゴリがより抽出されやすいか

1Kサンプルに含まれるカテゴリ
カテゴリ分けは先行研究より

Category	Purpose	Pre-training	Fine-tuning
Code	Code Logic	388	223
Dicts	Dictionaries or other data carriers	164	117
Docs	Documentation	173	291
License	License information	124	284
Testing	Test Code	50	2
Total		899	917

割合としては
Dictsと**License**が記憶される

DictsはFT後も忘却されにくい

→データ漏洩リスクは減らない

Code Attack type		Pre-training (RQ1-2)		Fine-tuning (RQ3)
Model	Category	Base	Fine-tuned	Fine-tuned
Starcoder2-3B	Code	115	53	56
	Dicts	<u>124</u>	<u>31</u>	54
	Docs	32	4	104
	License	<u>85</u>	<u>8</u>	185
	Test	3	2	1
Starcoder2-7B	Code	117	64	53
	Dicts	<u>112</u>	<u>26</u>	52
	Docs	42	5	123
	License	<u>88</u>	<u>11</u>	181
	Test	3	3	1
Starcoder2-15B	Code	136	62	88
	Dicts	<u>133</u>	<u>47</u>	64
	Docs	52	16	137
	License	<u>94</u>	<u>15</u>	195
	Test	4	4	1

実務への影響

リスク

- ・ セキュリティ脆弱性の伝搬
脆弱なコードを再生成する可能性
- ・ データプライバシー漏洩
- ・ 著作権侵害, ライセンス違反のリスク

対策

- ・ データ重複除去
- ・ 許容的ライセンスの使用
- ・ アンラーニング（忘れさせる技術）の活用

まとめ

コードLLMが学習データをどれほど再現するかを調査

大規模モデル（FT前）は学習データの約55%を再現した

ライセンス情報，データ構造は再現されやすい

学習データに重複したコードがあれば再現されやすい

企業・個人がFTするときはセキュリティリスクがあることに注意しなければならない