

Is This You, LLM? Recognizing AI-written Programs with Multilingual Code Stylometry

Andrea Gurioli and Maurizio Gabbrielli and Stefano Zacchioli

<https://arxiv.org/abs/2412.14611>

概要

タイトル

- AIで書かれたプログラムを多言語コードスタイロメトリーで識別する

背景

- **情報漏洩**の観点から， ChatGPTなどのサードパーティのLLMの使用を禁止されることがある.
- 学校などの教育現場では使用がカンニング扱いされることがある.
- コード生成の際， データセットに含まれるコードをそのまま出力することがあり， **法的なリスク**を生み出す.

概要

目的

- LLMによって生成されたコードを自動的に認識し，人間によって書かれたコードと区別する.
 - コードスタイロメトリー

先行研究

1. 特定の一言語でのみ検出できる.
2. 対象のLLMが特定のものであり，一般性に欠ける.
3. 分類法が以前からある分析手法である.

複数の言語において分類を可能とする

アプローチ

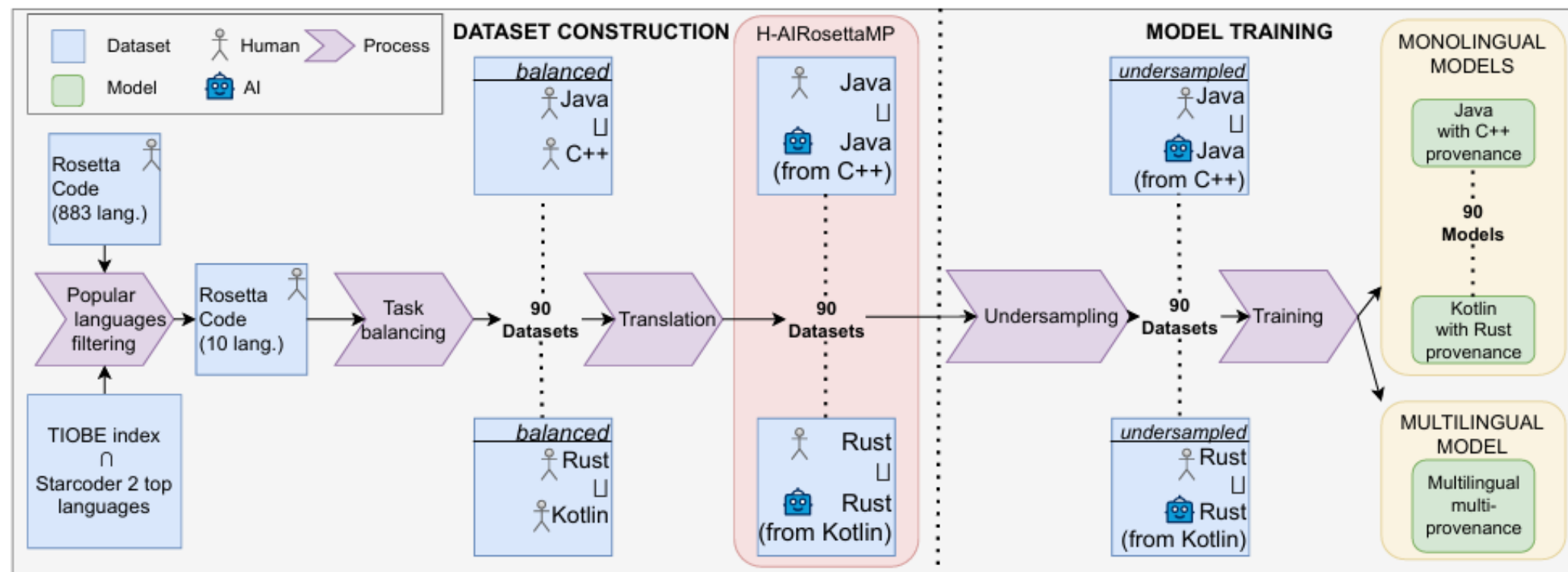


Fig. 1. Experimental methodology. The process is divided into two main steps: (1) The *Dataset construction*, which starts from the filtered Rosetta Code dataset and terminates in the *H-AIRosettaMP*, obtained via code translation, comprising 90 (sub-)datasets. Each dataset is labeled by the author (Human or AI) and is represented by dst (the language of the dataset) and src language (the provenance of the AI-generated part of the dataset); (2) The *model training*, that shows the process leading to 90 monolingual models (one per dataset) and 1 multilingual model.

- データセットの構築
- モデルのトレーニング

アプローチ

データセットの構築

- Rosetta Code（人間が書いたコード）
 - 893の言語, 79013のコード
- 人気言語のランキング（上位10種）

コードの生成

- クロス翻訳を使用して生成.
- 過学習を避けるため, 元のコード以外の9種の言語に翻訳.
- 自然言語からコードを生成.
- コードを説明させ, その説明からコードを生成.

アプローチ

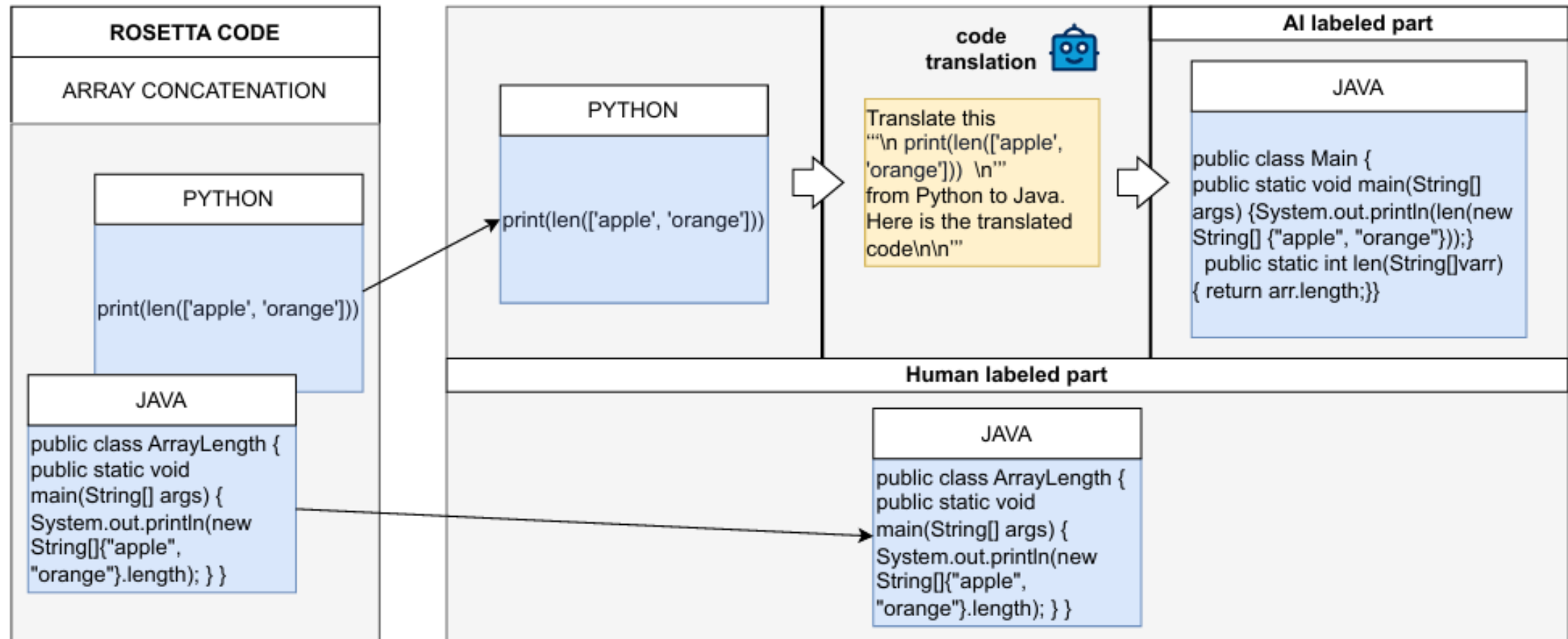


Fig. 2. Code translation step. The human-labeled part of the dataset (the `ArrayLength` Java class here) is a solution to a task from Rosetta Code (Array concatenation). The AI-labeled part is obtained via code translation (from Python to Java) using StarCoder 2. Input to the translation is a human-written solution for the same task, in a different programming language.

アプローチ

モデルのトレーニング

- コードスタイロメトリーに初めてトランスフォーマベースのエンコーダ分類器を使用.
- CodeT5plus-770Mというトレーニング済みのモデルに分類ヘッドを追加したものを使用.
- 90種のモノリンガルモデルと, 多言語モデルの2分野でモデルを作成.
 - 各モデルで人間, AIがそれぞれ470ずつとなるようにアンダーサンプリング.
- 分類に影響するノイズを削除.

結果

各モデルの結果

- 言語によって制度に有意な差がみられた。
- 特にKotlinから翻訳した言語では精度が高かった。

TABLE II
ACCURACY (%) OF ALL TRAINED CLASSIFIERS WHEN SOLVING THE AI CODE STYLOMETRY TASK.

Prov. language	Language tested										Prov. accuracy (avg. $\pm$ std)
	C++	C	C#	Go	Java	Javascript	Kotlin	Python	Ruby	Rust	
<i>Monolingual models</i>											
C++	-	88.3	87.8	91.0	81.4	86.2	91.0	87.2	86.2	84.0	87.0 $\pm$ 2.9
C	89.9	-	90.9	91.5	92.0	90.9	94.1	89.9	88.8	88.8	90.8 $\pm$ 2.0
C#	75.5	87.8	-	93.6	88.3	91.5	92.5	87.8	88.3	88.8	88.2 $\pm$ 4.9
Go	94.7	94.7	87.2	-	90.9	95.2	94.7	84.0	90.4	85.6	90.8 $\pm$ 4.1
Java	91.5	92.0	86.7	92.0	-	89.4	92.5	87.8	88.8	83.0	89.3 $\pm$ 2.9
Javascript	90.9	95.2	88.3	91.5	87.7	-	93.1	85.6	90.4	84.0	89.6 $\pm$ 3.4
Kotlin	98.4	92.0	94.7	92.6	94.1	96.8	-	94.1	94.7	89.9	94.1 $\pm$ 2.4
Python	92.5	95.2	81.9	90.9	90.9	90.9	96.3	-	77.6	85.1	89.0 $\pm$ 5.9
Ruby	90.4	96.3	86.7	93.1	87.7	89.4	94.1	78.2	-	86.2	89.1 $\pm$ 5.0
Rust	88.3	97.8	90.9	87.3	94.7	91.5	87.8	88.8	89.4	-	90.7 $\pm$ 3.3
Language accuracy (avg. $\pm$ std)	90.2 $\pm$ 5.9	93.2 $\pm$ 3.3	88.3 $\pm$ 3.4	91.5 $\pm$ 1.7	89.7 $\pm$ 3.8	91.3 $\pm$ 3.0	92.9 $\pm$ 2.3	87.0 $\pm$ 4.1	88.3 $\pm$ 4.3	86.1 $\pm$ 2.3	-
<i>Multilingual model</i>											
Multilingual model accuracy	88.8	88.3	84.0	89.4	82.4	79.3	87.8	79.3	80.8	81.4	84.1 $\pm$ 3.8
Multilingual model F1	88.8	88.8	84.0	89.3	82.4	78.9	87.8	78.8	80.5	81.1	84.0 $\pm$ 4.0
Multilingual model AUC	94.1	94.9	91.8	93.9	89.9	90.4	94.6	90.8	88.6	93.9	92.3 $\pm$ 2.1

結果

先行研究との比較

- 3言語について調査した結果, モノリンガルモデルであればどれも高い精度を発揮した.
- クロス言語テストでも上回ることがあった.
- 既存手法が報告よりも精度が下がった.
 - より難度の高いデータセット

TABLE IV
COMPARISON BETWEEN THE ACCURACY OF BASELINES CLASSIFIERS [23], [33] AND CLASSIFIERS INTRODUCED IN THIS WORK. BEST RESULTS FOR EACH COLUMN ARE SHOWN IN **BOLD**.

Model	Tested language									
	Java	C++	Python	Javascript	C#	C	Go	Ruby	Rust	Kotlin
<i>Baseline models</i>										
RF Java [23]	79.3	-	-	-	-	-	-	-	-	-
J48 Java [23]	86.9	-	-	-	-	-	-	-	-	-
RF C++ [23]	-	85.5	-	-	-	-	-	-	-	-
J48 C++ [23]	-	89.5	-	-	-	-	-	-	-	-
XGB-TF-IDF Python [33]	-	-	92.6	-	-	-	-	-	-	-
<i>Proposed models</i>										
Java (monolingual)	94.1	83.9	87.7	90.6	88.9	80.4	86.7	86.6	85.0	31.3
C++ (monolingual)	89.3	98.4	88.0	90.7	86.7	89.5	86.1	87.0	86.2	24.1
Python (monolingual)	63.1	82.4	94.1	90.2	58.5	83.6	58.7	90.7	84.0	44.1
Multilingual	82.4	88.8	79.3	79.3	84.0	88.3	89.4	80.8	81.4	87.8

DISCUSSION

調査結果（言語による特徴）

- Kotlinが翻訳元であるモデルが最も精度が高い.
- 学習元ではない言語には精度が下がる.

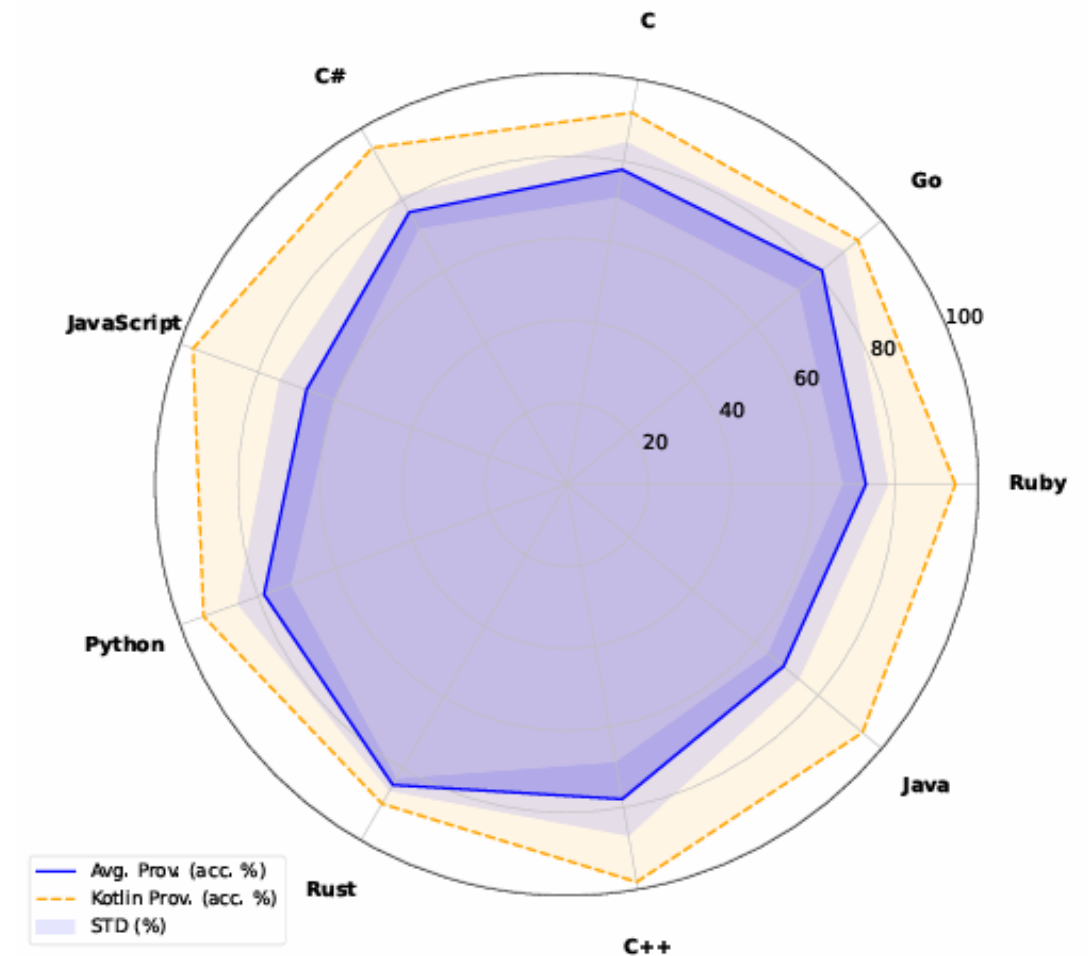


Fig. 6. Accuracy of our best-performing monolingual models. The dashed orange line corresponds to the Kotlin row in Table II (in-distribution). The solid blue line shows the accuracy of the same models on test sets with a different language provenance (potentially out-distribution, as is general translating solutions for the same task from different languages to the same language will result in different snippets). Model performances degrade significantly in the latter case.

DISCUSSION

調査結果（言語による特徴）

- Kotlinが翻訳元であるモデルが最も精度が高い.
- 学習元ではない言語には精度が下がる.

実用性は低くなる

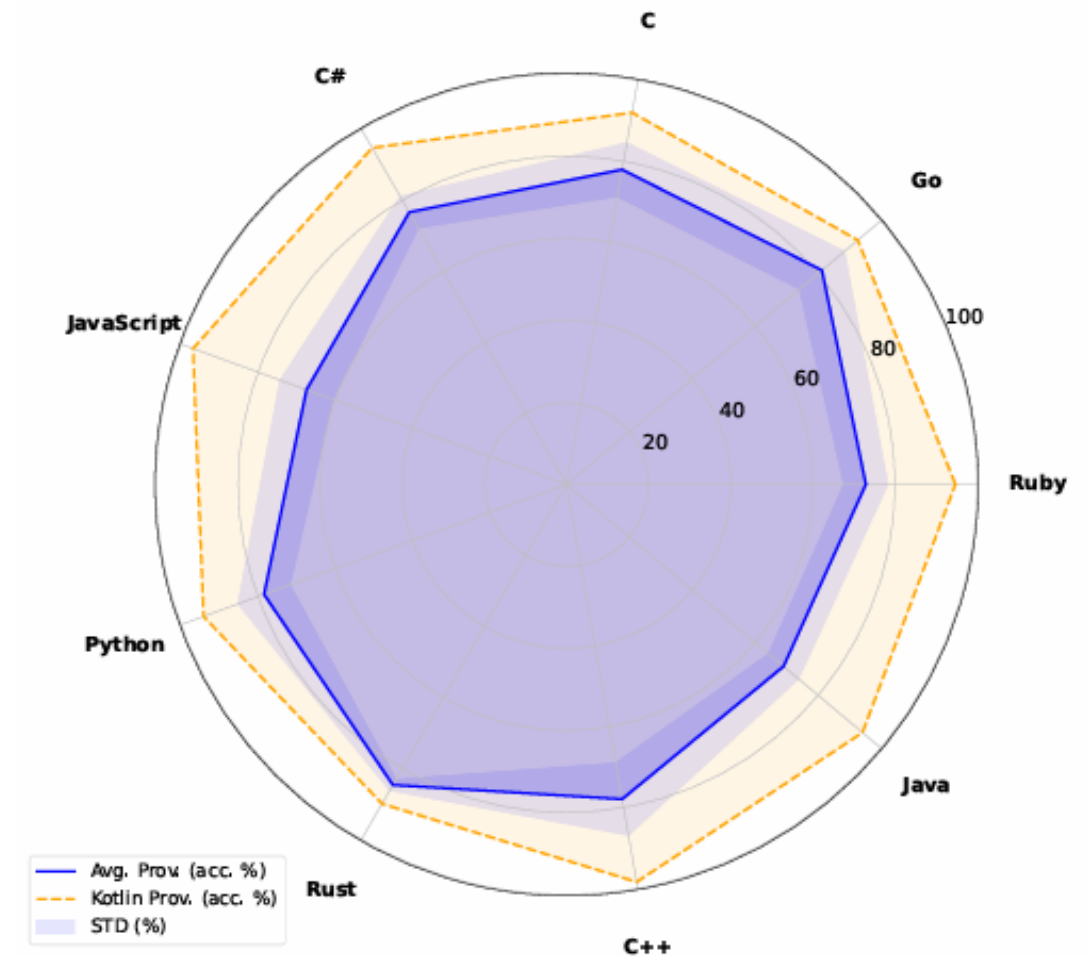


Fig. 6. Accuracy of our best-performing monolingual models. The dashed orange line corresponds to the Kotlin row in Table II (in-distribution). The solid blue line shows the accuracy of the same models on test sets with a different language provenance (potentially out-distribution, as is general translating solutions for the same task from different languages to the same language will result in different snippets). Model performances degrade significantly in the latter case.

DISCUSSION

ChatGPTとの比較

- ChatGPTを使用した既存モデルのデータセットを用いた結果，制度は72.8%（6.3%の低下）
- 既存モデルと比較すると25.2%制度が低下しているものの，どちらのクラスでも50%を大きく上回っており効果を発揮しているといえる.

著者の主張

- コード生成とコードスタイロメトリーの争いにおいて，閉鎖的なモデルとデータセットに頼った手法に依存すべきではない.
 - 使用できなくなるなどで再現できなくなる可能性
 - 過程の透明性のなさ

結論

AIコードスタイロメトリー

- コードが人間によって書かれたものか, AIが生成した物かを分類するタスク.

研究成果

- 複数の言語をカバーしたモデルで84.1%の精度を達成した.

有意性

- 高い精度を発揮するうえに, 最新の技術を使用し, 完全に再現可能である.