

API Recommendation for Machine Learning Libraries:How Far Are We?

Moshi Wei
York University
Toronto, Canada
moshiwei@yorku.ca

Jiho Shin
York University
Toronto, Canada
jihoshin@yorku.ca

Yuchao Huang
Institute of Software Chinese
Academy of Sciences
yuchao2019@iscas.ac.cn

Nima Shiri Harzevili
York University
Toronto, Canada
nshiri@yorku.ca

Junjie Wang
Institute of Software Chinese
Academy of Sciences
junjie@iscas.ac.cn

Song Wang
York University
Toronto, Canada
wangsong@yorku.ca

ESEC/FSE 2022 Research Papers “API Recommendation for Machine Learning Libraries: How Far Are We?”

発表者：田中 智達

背景

- 近年機械学習アプリケーションの開発への関心が高まっている.
- オープンソースの機械学習ライブラリやAPIは数多く公開されているが、特定のタスクに適したAPIを探すことは特に機械学習の分野では未熟者が多いため容易ではない[50].
- これまで、APIを高精度に検索するためのAPI推薦手法が数多く提案され、活発に研究されてきた[16, 17, 19, 35].

既存のAPI推薦手法

- 多くは、主にJavaなどの静的型付けプログラミング言語を用いた一般的なプログラミングタスクで評価.
- Pythonのような動的型付けプログラミング言語を用いたMLプログラミングタスクに対する実用性や有用性についてはほとんど知られていない.
- 本論文ではPythonベースのMLプログラミングタスクに対する既存のAPI推薦アプローチの有効性を調査.
 - 機械学習プラクティスの人気の高まりと、一般からの機械学習に関する質問の数が圧倒的に多いことを考えると、非常に価値がある.

RQ

- RQ1

- PythonベースのMLプログラミングタスクにおいて、既存のAPI推薦アプローチの性能はどの程度か.

- RQ2

- PythonベースのMLタスクにおいて、既存のAPI推薦アプローチの性能を向上させることは可能か.

評価指標

- 情報検索や推薦システムの評価でよく用いられる評価指標であるMRRとMAPを用い、API推薦アプローチを評価する [19, 22, 35, 36, 44, 46-48].
 - MRRは推薦リストにおいて最初に正しく推薦されたAPIの順位を記述。これが高いということは、最初に正しくマッチしたAPIのランクが高いことを示す。
 - MAPは最初の正解だけに注目するのではなくすべての正解のランクをチェックする。これが高いということは、推薦の中に複数の正確な一致があり、推薦が完全であることを示す。
- いずれも0～1までの範囲である。

研究対象

- 既存のAPI推薦アプローチの性能を調査するため、Python Package Index (PyPI) [2] におけるダウンロード数に応じ人気のある6つのPython MLライブラリを選択.
- API の数は、パッケージの最新バージョンでドキュメント化された関数ノードの数をカウントすることで算出.
- PythonベースのML質問データセットを作成するため、StackExchange [3]が提供するデータエクスプローラを使用して、Stack Overflowからプログラミング質問とその回答を抽出.
 - 質問に複数のAPIが含まれている場合は全てのAPIを連結してAPIリストとして回答.
 - 学習データセットのノイズを減らすため、少なくとも1つのAPIを含み、全てのAPIが同じライブラリに含まれる回答のみを選択.

研究対象

ライブラリ名	概要	#APIs In documents	#Unique APIs In dataset	#SO questions
Numpy	多次元配列と行列の演算を行うためのライブラリ	1,373	1,151	29,771
Pandas	データ操作・解析のためのライブラリ	1,367	913	16,197
Scikit-learn	機械学習アルゴリズムのためのライブラリ	1,154	1,082	6,376
Keras	人工ニューラルネットワークのためのインターフェイス	1,407	1,233	8,728
TensorFlow	Googleが開発したディープラーニング用のライブラリ	8,028	1,591	10,263
PySpark	Apache SparkのPythonによるインターフェイス	860	785	8,861

図：本研究で検討した機械学習ライブラリの詳細

実証実験(RQ1)

- Stack Overflowから収集したPythonベースのML質問に対して、BIKERとDeepAPIの学習と評価を行った.
- Python MLデータセットに適用した場合、両アプローチの性能はJava JDKデータセットでの性能と比較して有意に低下する(p 値 < 0.01)ことが判明.
- 既存のAPI推薦アプローチはAPI推薦の完全性をほとんど捉えることができないという課題がある.

Dataset	Approach	Evaluation Metrics	
		MRR	MAP
Python ML	BIKER	0.271	0.115
	DeepAPI	0.176	0.068
Java JDK	BIKER	0.554	0.505
	DeepAPI	0.188	0.153

図：PythonベースのML問題に対するBIKERとDeepAPIの性能

実証実験(RQ1)

- この優位な性能差の原因について、2つのデータセットとBIKERとDeepAPIの両方の推奨APIについて調査.
 - PythonベースのML質問はより多くのAPIを必要とする.
 - Java JDKの問題を解くのに必要なAPIは1つであるが、Python MLでは平均して5つのAPIが必要.
 - Java JDKの質問に対する回答の多くは単一のAPIしか含んでいないので、最初の正解確率を高めることに重点を置いている.
 - 既存のアプローチは、一般的なAPIの使用法を考慮していない.
 - 既存のAPI推薦手法では、与えられた質問に対するAPIを推薦する際、関係するAPI間の関係を見逃しがちであった.
 - さらに既存のAPI推薦アプローチのパフォーマンスを低下させる可能性のあるいくつかの小さな原因も特定.

実証実験(RQ2)

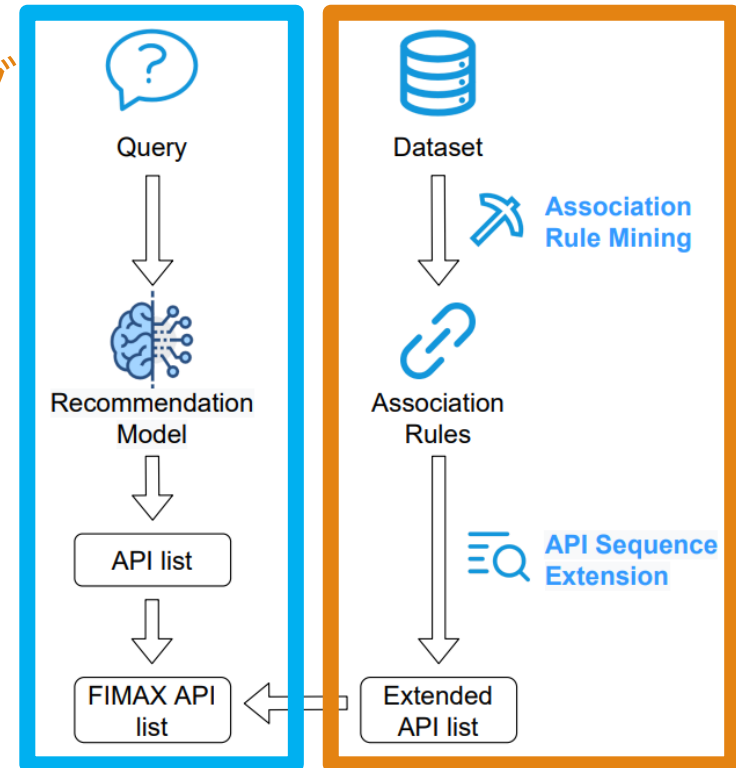
- RQ1で得られた知見に基づき、FIMAXと名付けた頻出アイテムマイニングに基づくアプローチを提案。
 - 既存のAPI推薦結果をAPIの利用情報で拡張することにより、PythonベースのML質問に対する既存のAPI推薦手法の性能を証明。
- 既存のAPI推薦手法とFIMAXの性能比較
 - BIKERとDeepAPIはFIMAX適用後に有意に良好な性能を達成。

Library	BIKER		BIKER+FIMAX		Improvement		DeepAPI		DeepAPI+FIMAX		Improvement	
	MRR	MAP	MRR	MAP	MRR	MAP	MRR	MAP	MRR	MAP	MRR	MAP
Numpy	0.4369	0.1639	0.5689	0.2114	30.75%	27.39%	0.2160	0.0699	0.4918	0.1697	127.6%	142.7%
Pandas	0.1131	0.0514	0.1955	0.0635	67.50%	21.48%	0.2286	0.0791	0.2749	0.0975	20.22%	23.24%
Scikit-learn	0.0187	0.0037	0.0417	0.0417	100.5%	96.95%	0.1751	0.0656	0.1888	0.0604	7.786%	-7.99%
Keras	0.0447	0.0098	0.0612	0.0237	7.021%	88.95%	0.2124	0.0765	0.2662	0.0986	25.34%	28.85%
Tensorflow	0.2543	0.0666	0.3578	0.1137	37.40%	61.39%	0.2954	0.1020	0.4060	0.1497	37.41%	46.73%
Pyspark	0.2294	0.0422	0.3998	0.0969	71.31%	126.7%	0.2513	0.0838	0.4578	0.1413	82.19%	68.49%
Overall	0.1829	0.0563	0.2714	0.0864	48.36%	53.46%	0.2298	0.0795	0.3476	0.1196	54.28%	57.36%

図：BIKERとDeepAPIにおけるFIMAXの性能

FIMAXの概要

1. ML質問の回答からアソシエーションルールマイニングの手法でAPIの利用パターンを抽出.
2. API推薦モデルから抽出されたAPI利用パターンを用い, さらに推薦を拡張.



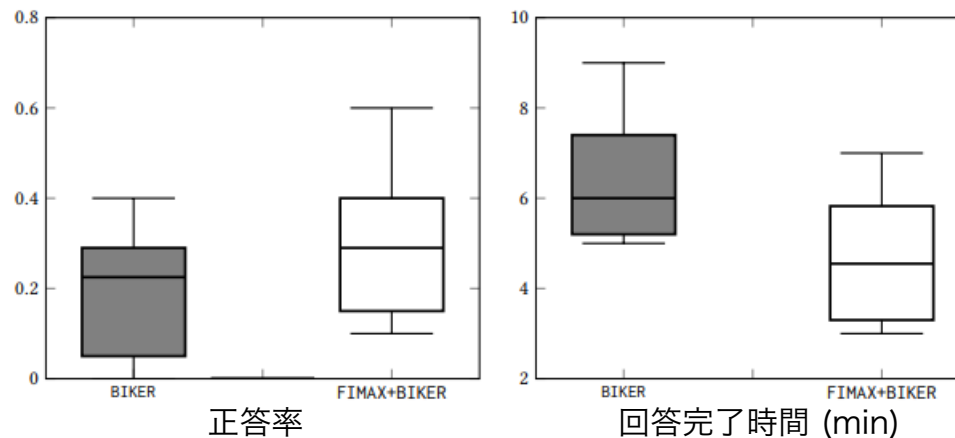
図：FIMAXの概要

ユーザ調査

- FIMAXが開発者がより効率的かつ正確に正しいAPIを見つけるのに役立つかどうかを調査.
 1. テストデータセットからランダムに20問を選択.
 2. それぞれの質問に対し, 質問のタイトルをタスクの記述としてAPI検索タスクを作成.
 3. 参加者の開発経験年数が均等になる様2つのグループに分け, タスクもランダムに2つのグループに分類.
 4. 2グループそれぞれがFIMAX+BIKERとBIKERを使って片方のタスクをクリア.
 5. ツールを交換し, 残りのタスクをクリア.

ユーザ調査

- 正答率と回答完了時間で評価.
 - FIMAX+BIKERを使用した場合、オリジナルのBIKERを使用した参加者に比べより正確に、より短時間でタスクを完了.
 - 差の統計的有意性を検証するために、Wilcoxon符号順位検定を使用.
 - 正答率と完了時間のp値はともに0.05より小さい.
 - 正答率と完了時間におけるFIMAX+BIKERとBIKERの差は統計的に有意.



図：BIKERとDeepAPIにおけるFIMAXの性能

結論

- API推薦アプローチであるBIKERとDeepAPIの有効性を調査した結果、下記の理由で性能が低下していた。
 - PythonベースのML質問はより多くのAPIを必要とする.
 - 既存のアプローチは、一般的なAPIの使用法を考慮していない.
- 既存の API推薦手法を拡張するFIMAXを提案
 - FIMAXは既存のAPI推薦アプローチを大幅に強化.
 - ユーザー調査により、FIMAXのAPI推薦における実用的な価値も実証.
- 今後、他のAPI推薦手法や他のドメイン・言語のプログラミングタスクに対するFIMAXの有効性を調査する予定.