

Complementing Global and Local Contexts in Representing API Descriptions to Improve API Retrieval Tasks

Thanh Nguyen
Iowa State Univ., USA
thanhng@iastate.edu

Ngoc Tran
Univ. of Texas-Dallas, USA
ngoctran@utdallas.edu

Hung Phan
Iowa State Univ., USA
hungphd@iastate.edu

Trong Nguyen
Iowa State Univ., USA
trong@iastate.edu

Linh Truong
Univ. of Texas-Dallas, USA
linh.h.truong@utdallas.edu

Anh Tuan Nguyen
Axon Corp, USA
ntanhbk44@gmail.com

Hoan Anh Nguyen
Iowa State Univ., USA
hoan@iastate.edu

Tien N. Nguyen
Univ. of Texas-Dallas, USA
tnn160630@utdallas.edu

背景

- テキストとソースコードを関連付けることは、Software Engineeringにおいて重要なタスク
 - 開発者が与えたテキストクエリからコード片を検索するタスク[8, 11, 29, 52, 55].
 - APIクエリから、APIのチュートリアルを検索するタスク[22, 23, 41]
- 自然言語処理（NLP）技術は、SE文書においても使用され成功を収めている

背景

- 特に注目を集めている技術に**Word2vec**がある
 - Word2vecを利用することで単語をベクトル化することができる
- Word2vecは特にAPIドキュメントとソフトウェアライブラリのチュートリアルに適用される
 - ソースコードと英語をベクトル空間をベクトル空間に投影することで、テキストとソースコードの類似度を算出タスクなどに利用される

課題

- Word2vecでは単語個別のベクトルを利用し，テキスト間の類似度を算出する
 - ▶ 文書に出てくる単語の特徴ベクトルを平均し，その文書の特徴ベクトルとする.
- 同じ文書内の単語が独立したものとして扱われる
semantic（意味？）を考慮できない
- APIを利用したソースコードなどでは，コード間の順序関が重要
- 同じ単語でも文書によって使われ方が異なる場合もある

研究目的

- APIドキュメントとSoftware Engineering文書のベクトル化に対して、高いパフォーマンスを示すモデルを提案する
 - 文書のsemantic（意味？）をより良く捉えるために、**Doc2vec**を利用した手法を提案する

Word2vec, Doc2vec

Word2vec

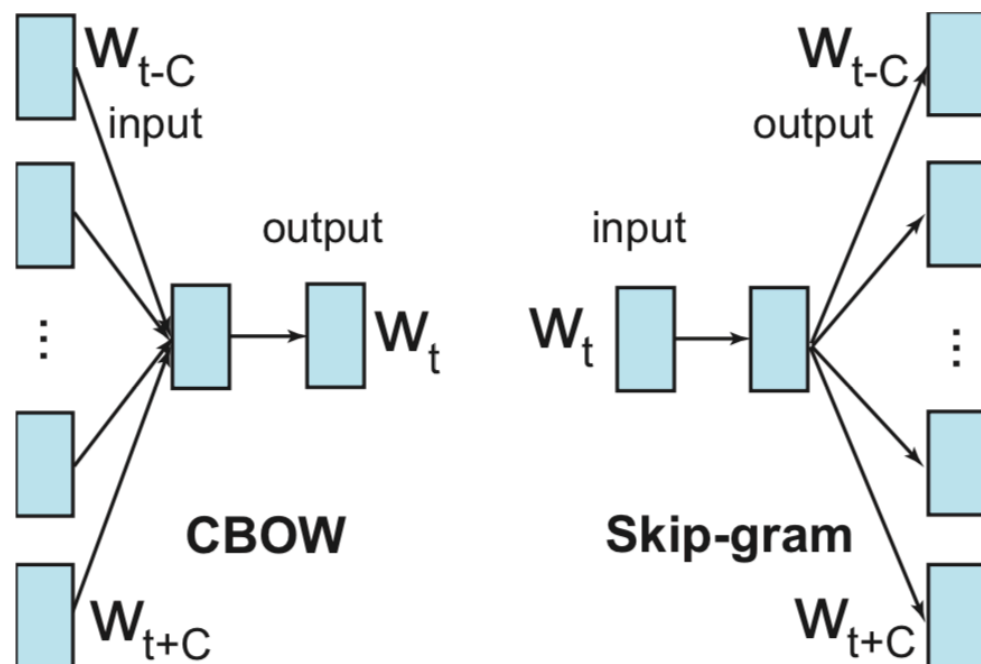


Figure 1: Word2vec with two different architectures

Doc2vec

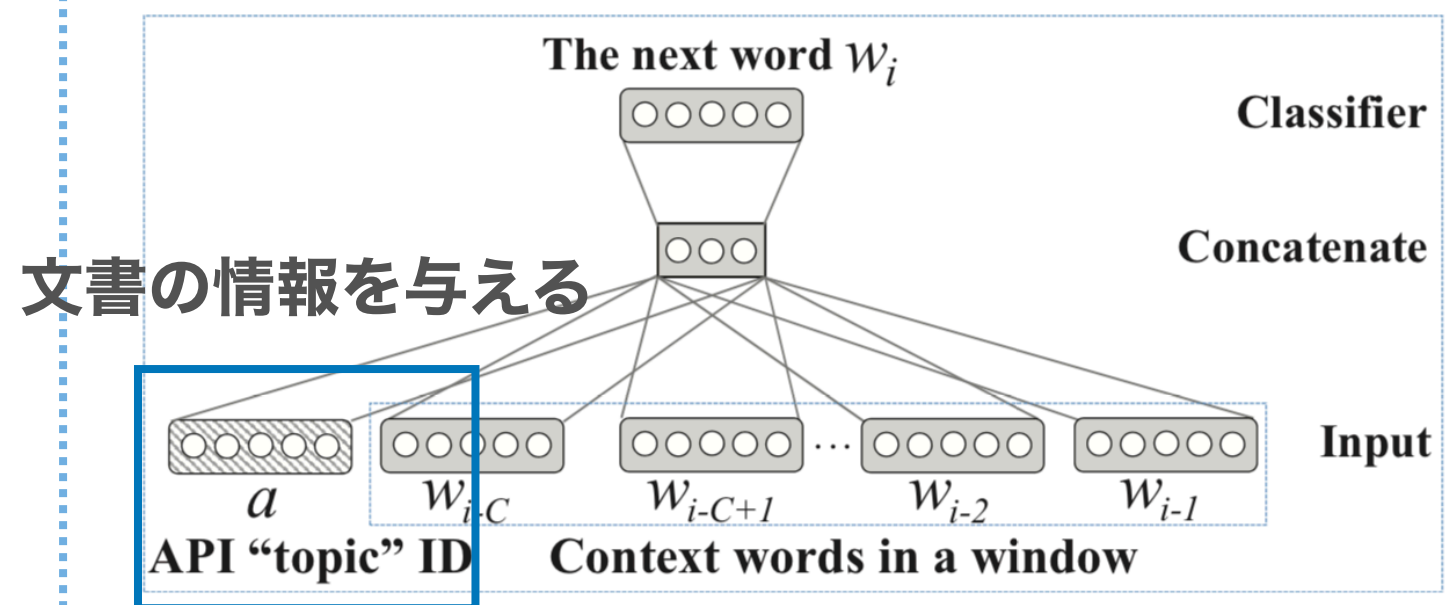


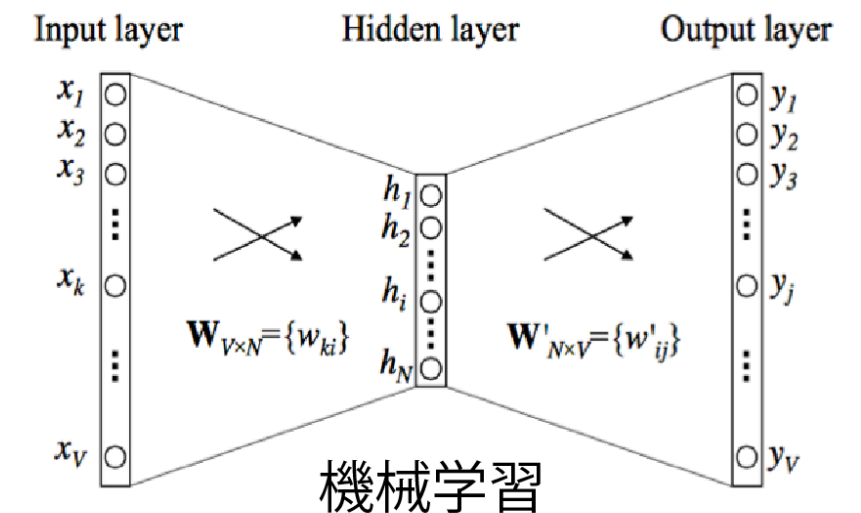
Figure 3: D2VEC model to learn vector representations for words and API elements from API documentation

分散表現でできること

東京都の
天気は
晴れです

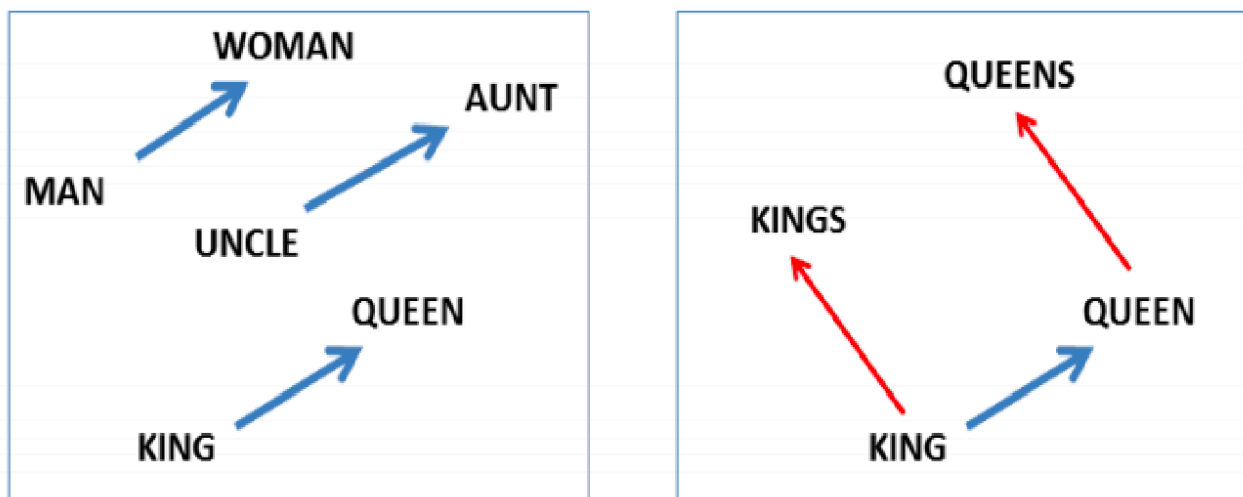
テキスト
整形処理

東京／都／の／
天気／は／
晴れ／です

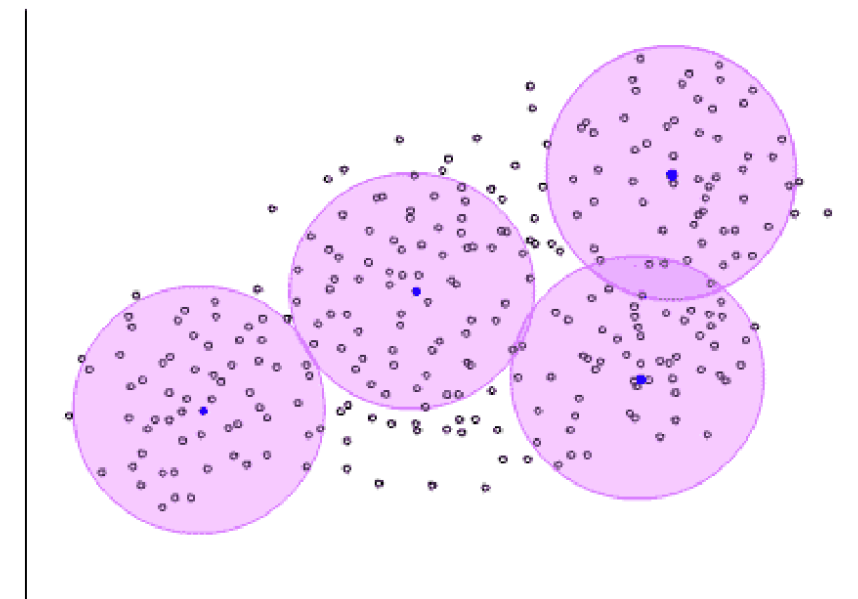


ベクトル化

KING-MAN+WOMAN=QUEEN



ベクトル空間形成



リサーチクエスチョン

- テキストからコード，コードからテキスト，コードからコードへ検索性能を調査

RQ1: テキストからのAPIコード検索において，既存の手法よりも

Doc2Vecは高い性能を示すか？ (text-to-code retrieval)

RQ2: APIから対応するAPIのチュートリアルを検索するタスクにおいて

Doc2Vecは高い性能を示すか？ (code to text retrieval)

RQ3: 2つの類似するAPIライブラリを見つけるタスクにおいてDoc2Vec

は高い性能を示すか？ (code-to-code retrieval)

Doc2Vecの学習

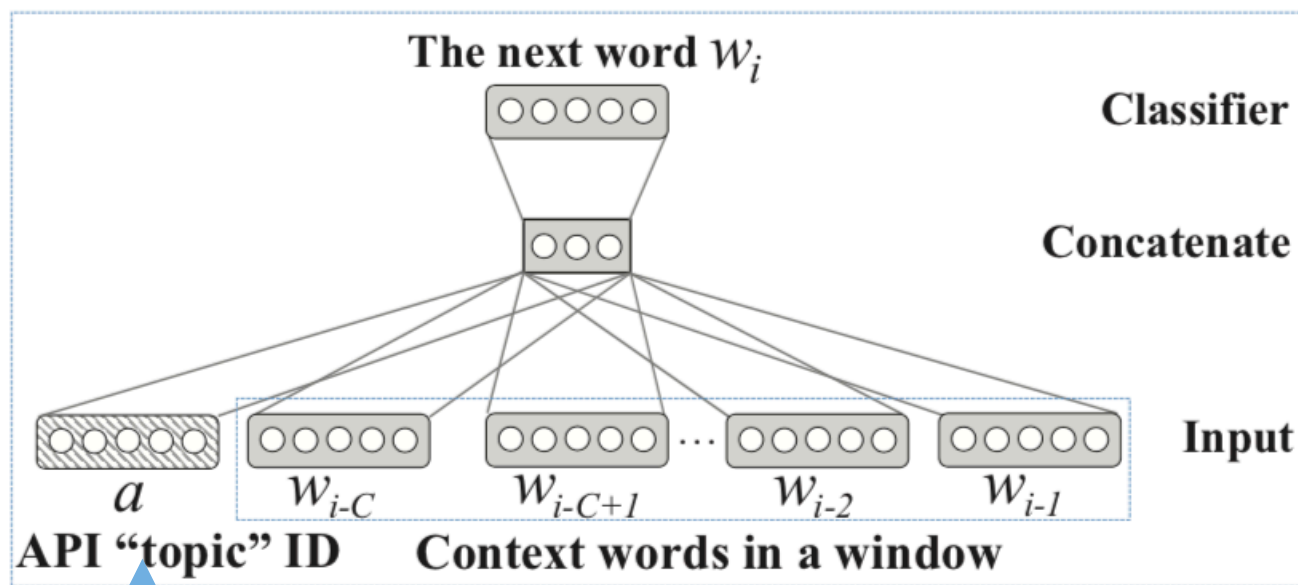


Figure 3: D2VEC model to learn vector representations for words and API elements from API documentation

APIのドキュメント

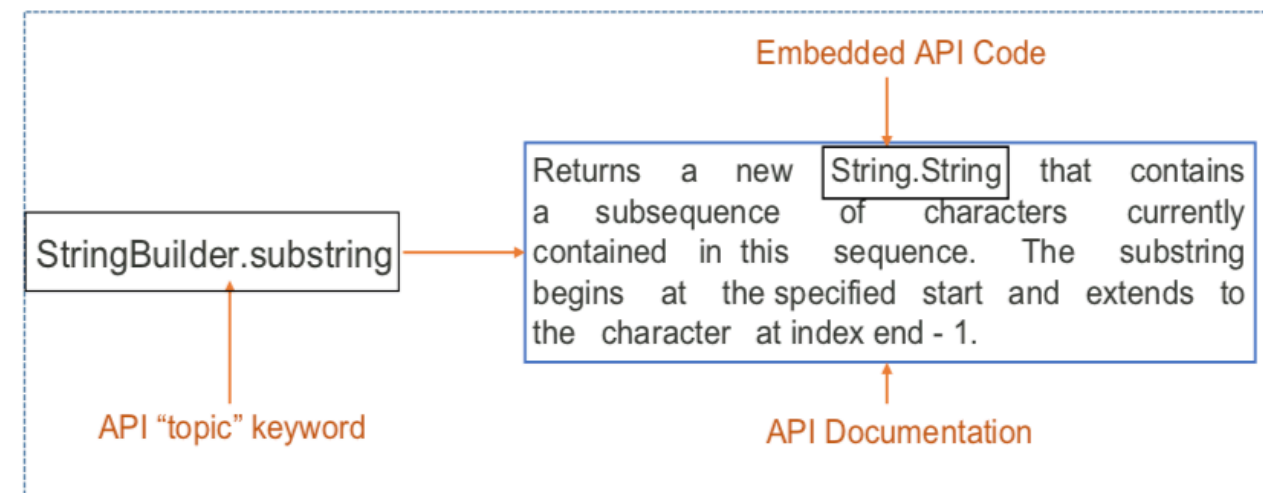


Figure 2: Documentation of JDK API `StringBuilder.substring`

どのAPIについての説明かといった情報を保持しながら学習を行う

評価実験

学習データ

- APIドキュメント (Java JDK Core, Apache Commons libraries)

Table 1: JDK and Apache Training Dataset

Number of distinct API elements	36,923
Number of distinct English words	10,258
Occurrence frequency of APIs	3.2
Occurrence frequency of English words	25.6
Vocabulary size	47,381
Number of APIs embedded within a documentation	2.7

RQ1:API EXAMPLE CODE SEARCH

- 実験の設定

- JDKのAPIのみを使用

- KodeJavaからデータセットを作成

KodeJava : Javaのサンプルコード集

JDKデータセットに含まれる
APIを参照して正解集合を作る

Table 1: JDK and Apache Training Dataset

Number of distinct API elements	36,923
Number of distinct English words	10,258
Occurrence frequency of APIs	3.2
Occurrence frequency of English words	25.6
Vocabulary size	47,381
Number of APIs embedded within a documentation	2.7

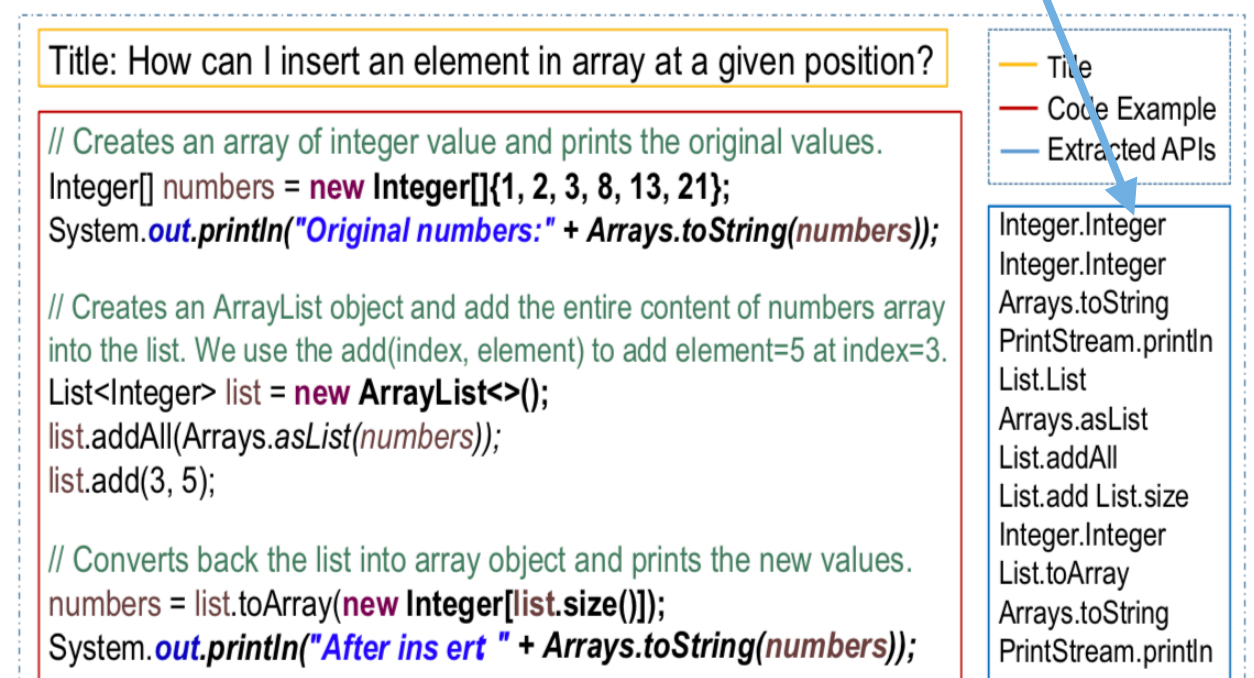


Figure 4: An API code example with query from KodeJava
KodeJavaの例

RQ1:結果

- rVSM : Word2vecの改良モデル（コード検索向け？）

Table 3: Accuracy comparison on API code search

	top-1	top-2	top-3	top-4	top-5
Word2Vec	11.7	18.5	22.7	26.3	29.5
D2VEC	19.9	28.8	33.9	38.2	41.2
revised Vector Space Model (rVSM)	19.1	23.0	25.4	27.4	35.2
rVSM+ Word2Vec	28.2	30.4	33.5	34.6	38.5
rVSM+D2VEC	35.5	46.7	52.7	58.1	60.9

RQ2. SEARCHING RELEVANT FRAGMENTS IN API TUTORIALS

- APIから対応するチュートリアルを検索する

APIコードと文書をベクトル化しコサイン類似で検索

Table 6: Tutorial Corpus

Tutorial	#APIs	Explanatory Fragment	Non-Expla. Fragment	Frag with APIs	Frag w/o APIs
Joda Time	36	19	10	21	8
Math Lib	73	31	10	16	25
Col. Official	59	31	26	17	40
Col. Jenkov	28	34	35	42	27
Smack	40	42	5	31	16

RQ2. 結果

- コードからドキュメント検索タスクにおいて, Doc2vecが最も良い性能を示す

Table 7: Accuracy in Ranking Relevant Tutorial Fragments for APIs

	Precision (%)			Recall (%)			F-score (%)		
Tutorial	FRAFT	Word2Vec	D2VEC	FRAFT	Word2Vec	D2VEC	FRAFT	Word2Vec	D2VEC
Joda Time	85.19	44.62	73.17	76.67	100.00	100	80.70	61.70	84.51
Math Lib	84.78	81.25	82.81	73.58	98.11	100.00	78.79	88.89	90.60
Col. Official	62.03	50.00	64.29	87.50	58.93	64.29	72.59	54.10	64.29
Col. Jenkov	61.19	48.57	57.69	97.62	100.00	88.24	75.23	65.38	69.77
Smack	77.94	87.88	87.88	94.64	100.00	100.00	85.48	93.55	93.55
Average	74.23	62.46	73.17	86.00	91.41	90.50	78.56	72.72	80.54

RQ3. MAPPING BETWEEN API LIBRARIES

- JDK CoreのAPIを入力として，同等の帰納のApache CommonsのAPIを探せるか？
- 結果（青：Word2vec，赤：Doc2vec）

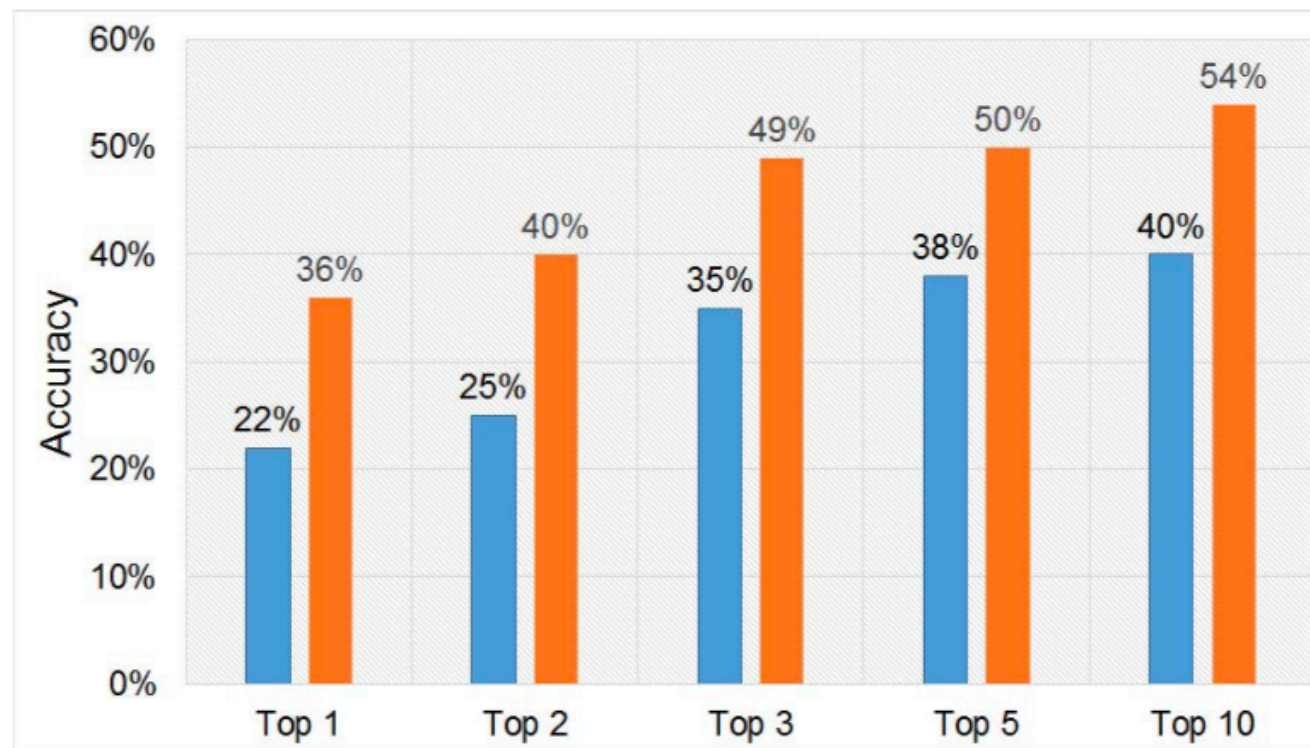


Figure 7: Accuracy comparison in deriving API mappings

まとめ

- まとめ
 - 文書をベクトル化するには, Word2vecよりもDoc2vecが有用
 - これまでのLDAとかが使われていた部分もDoc2vecに置き換わるかも
- 所感
 - 評価実験のやり方が考えられている