Exploring API Embedding for API Usages and Applications

By Nguyen T D, Nguyen A T, Phan H D, et al.

What is API

- Application Programming Interface
- Allows two applications to talk to each other

Key Point

- Exploring API Embedding for API Usages and Applications ?
- Exploring Word Embedding for Word Usages and Sentence
- API->Word

Why?

- API usage patterns
- Programming Language->Natural Language
- F.open, F.close->Hello, Bye

Why?

- API usage patterns
- Programming Language->Natural Language
- F.open, F.close->Hello, Bye
- NLP!

WORD2VEC

- Neural network model
- Like -> (1,1,0,0,1,1,0)
- First 1 represents
- Second 1 represents

CBOW

- Input Layer: a window of n words preceding and succeeding current word wi, one-hot encoding

- Output Layer: the word2vec vector of the predicted word w

- lower but more meaningful demension

- Training: matrix in hidden layer

API2Vec

- Similar contexts
- Word2Vec->API2Vec
- A window of n words preceding and succeeding current word wi
- A window of n APIs preceding and succeeding current word APIi

RQ1

- In a vector space produced by API2VEC on API elements, do nearby vectors represent the APIs that have similar usage contexts (defined as similar surrounding API elements of those APIs)?

nearby vectors?

- Like -> (1,1,0,0,1,1,0)
- Love -> (1,1,0,0,1,1,1)
- Similar usage contexts
- StringBuffer and StringBuilder

RQ2

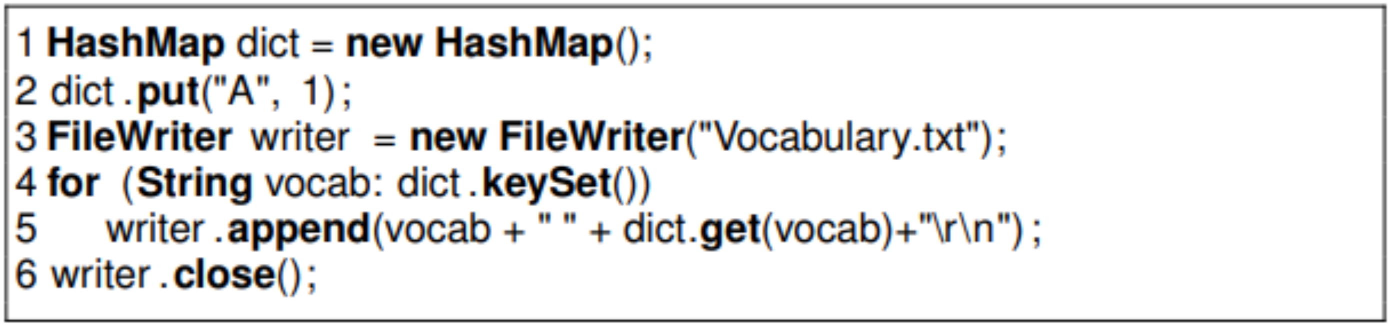
- By vector offsets, can API2VEC reveal similar usage relations between API elements (defined as co-occurring relations between API elements in API usages)?

Word/API Pair

- Vise and Versa Pros and Cons
- (1,1,1) and (0,0,0) (1,1,0) and (0,0,1)
- Offset:(1,1,1)

Building API Sequence

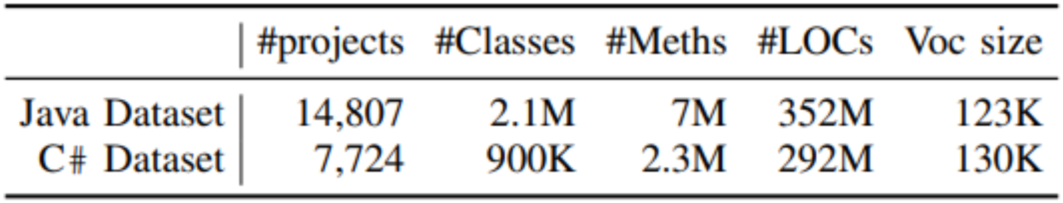
- AST
- Literal, Identifier, Method call, Constructor call or field access, Variable declaration, Array access, Statements.
- Nature



```
HashMap dict = new HashMap();
dict.put("A", 1);
FileWriter writer = new FileWriter("Vocabulary.txt");
for (String vocab: dict.keySet())
     writer.append(vocab + " " + dict.get(vocab)+"\r\n");
writer.close();
```

- HashMap#var HashMap.new String#ret HashMap#rec HashMap.put String#arg Integer#arg FileWriter#var FileWriter.new String#arg for String#var String[]#ret HashMap#rec HashMap.keySet String#ret HashMap#rec HashMap.get String#arg FileWriter#rec FileWriter.append String#arg FileWriter#rec FileWriter.close

Dataset



	#projects	#Classes	#Meths	#LOCs	Voc size
Java Dataset	14,807	2.1M	7M	352M	123K
C# Dataset	7,724	900K	2.3M	292M	130K

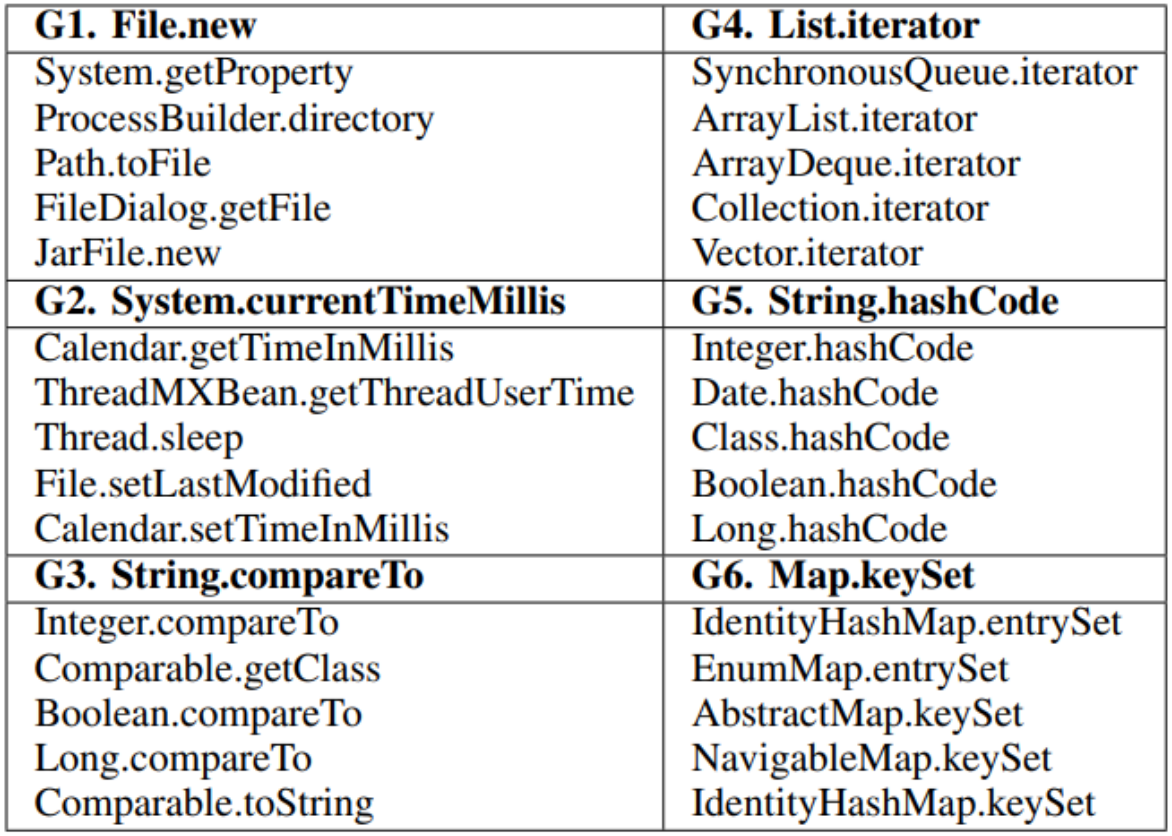
RQ1

- Randomly selected 1,000 JDK API methods and fields
- Top-5 API method calls and field accesses that are closest to that API
- Threshold: 80%

RQ1

- 4,632 pairs (92.64% of them) have similar surrounding

RQ1



G1. File.new	**G4. List.iterator**
System.getProperty	SynchronousQueue.iterator
ProcessBuilder.directory	ArrayList.iterator
Path.toFile	ArrayDeque.iterator
FileDialog.getFile	Collection.iterator
JarFile.new	Vector.iterator
G2. System.currentTimeMillis	**G5. String.hashCode**
Calendar.getTimeInMillis	Integer.hashCode
ThreadMXBean.getThreadUserTime	Date.hashCode
Thread.sleep	Class.hashCode
File.setLastModified	Boolean.hashCode
Calendar.setTimeInMillis	Long.hashCode
G3. String.compareTo	**G6. Map.keySet**
Integer.compareTo	IdentityHashMap.entrySet
Comparable.getClass	EnumMap.entrySet
Boolean.compareTo	AbstractMap.keySet
Long.compareTo	NavigableMap.keySet
Comparable.toString	IdentityHashMap.keySet

RQ1

- Cosine distances

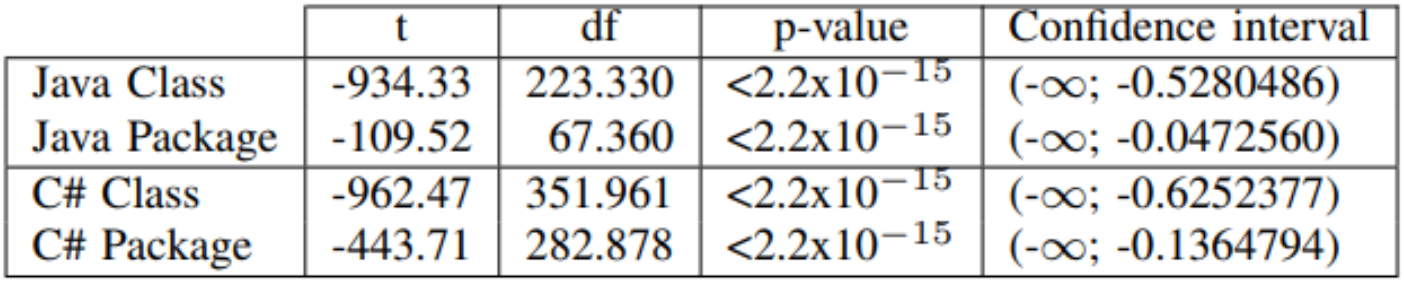
- independent-samples t-test with significance level α = 0.99.

RQ1

- Cosine distances

- independent-samples t-test with significance level α = 0.99.

RQ1



	t	df	p-value	Confidence interval
Java Class	-934.33	223.330	<2.2x10^{-15}	(-∞; -0.5280486)
Java Package	-109.52	67.360	<2.2x10^{-15}	(-∞; -0.0472560)
C# Class	-962.47	351.961	<2.2x10^{-15}	(-∞; -0.6252377)
C# Package	-443.71	282.878	<2.2x10^{-15}	(-∞; -0.1364794)

RQ2

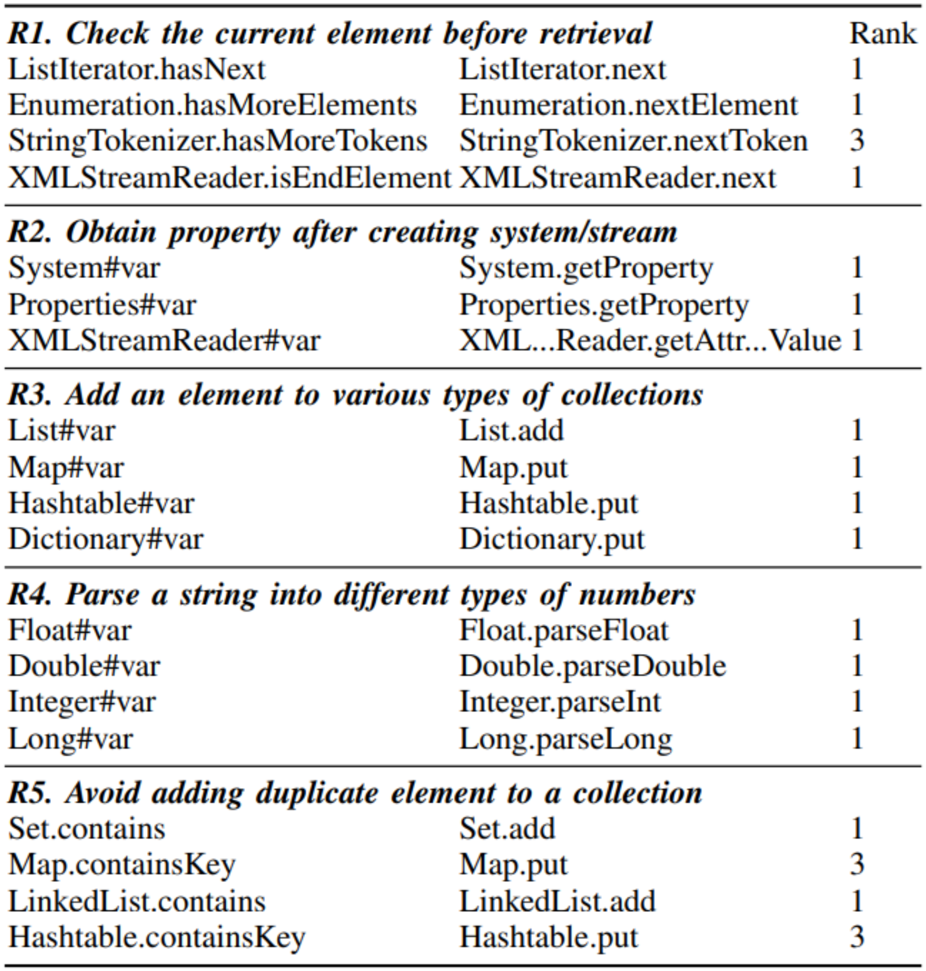
- Mining frequent pairs of APIs

- X = V(List.add) – V(List#var) + V(Map#var)

RQ2

- 94.2% : in the top-5 candidate list
- 74.1% : top one

RQ2



R1. Check the current element before retrieval		Rank
ListIterator.hasNext	ListIterator.next	1
Enumeration.hasMoreElements	Enumeration.nextElement	1
StringTokenizer.hasMoreTokens	StringTokenizer.nextToken	3
XMLStreamReader.isEndElement	XMLStreamReader.next	1
R2. Obtain property after creating system/stream		
System#var	System.getProperty	1
Properties#var	Properties.getProperty	1
XMLStreamReader#var	XML...Reader.getAttr...Value	1
R3. Add an element to various types of collections		
List#var	List.add	1
Map#var	Map.put	1
Hashtable#var	Hashtable.put	1
Dictionary#var	Dictionary.put	1
R4. Parse a string into different types of numbers		
Float#var	Float.parseFloat	1
Double#var	Double.parseDouble	1
Integer#var	Integer.parseInt	1
Long#var	Long.parseLong	1
R5. Avoid adding duplicate element to a collection		
Set.contains	Set.add	1
Map.containsKey	Map.put	3
LinkedList.contains	LinkedList.add	1
Hashtable.containsKey	Hashtable.put	3

API MAPPINGS BETWEEN JAVA AND C#

- API Mapping -> Language Translation
- Hello -> Bon Jour
- System.out.println -> Console.Writeln

API2API

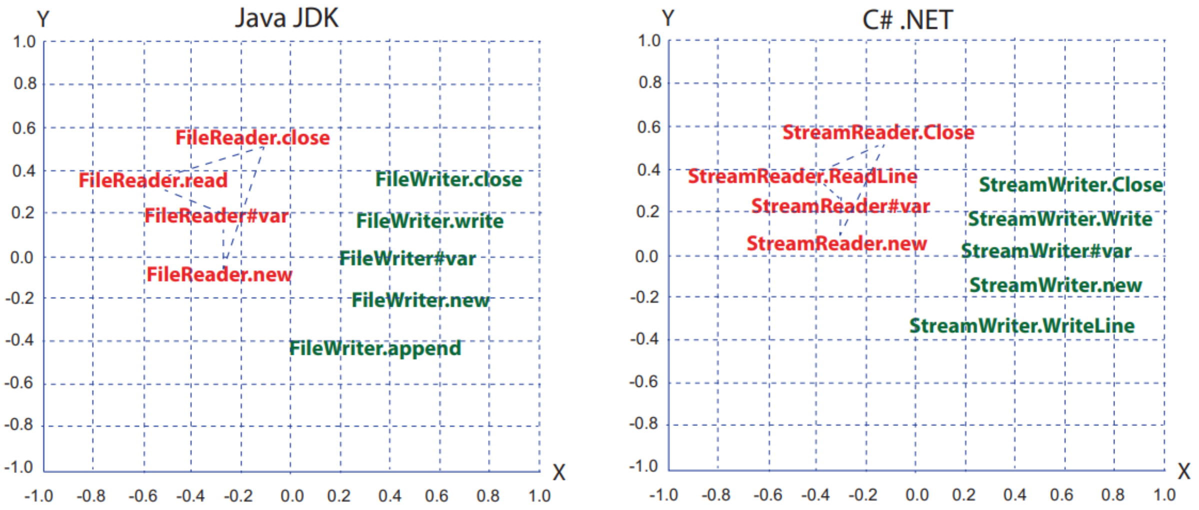
- Semantic relations among APIs in their usages are observed in the two vector spaces for the two languages as similar geometric arrangements among their vectors. .

API2API

- Semantic relations among APIs in their usages are observed in the two vector spaces for the two languages as similar geometric arrangements among their vectors. .

API2API

- FileReader and FileWriter



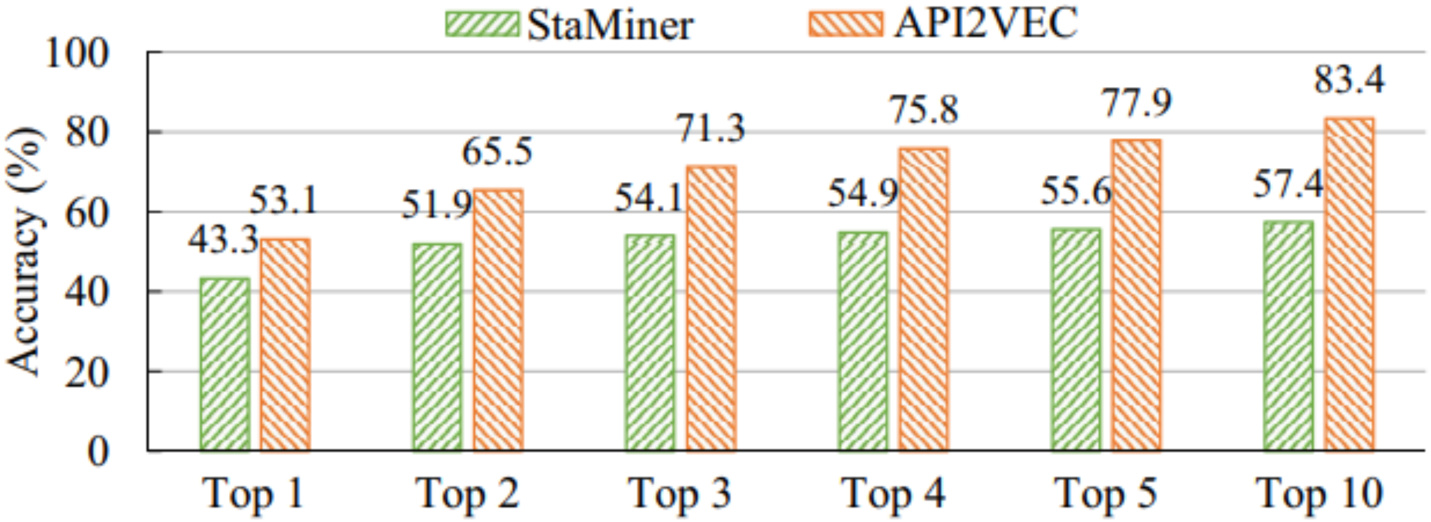
API2API

- Training dataset: a set of API mappings that was provided as part of the migration tool Java2CSharp

- API2Vev vectors

- Minimizing the Least Square

Quantitative Comparison



Qualitative Comparison

- API2API performs better than StaMiner with 34,628 pairs of respective methods

Impacts of Factors on Accuracy,

- Selecting different packages of API mapping pairs to train the transformation matrix
- Varying Numbers of Dimensions of Vector Spaces
- …

Conclusion

- Word2Vec for APIs can capture the regularities of the relations of APIs in API usages

- Propose an approach to automatically mine API mappings by learning the transformation between the two vector spaces of APIs in the source and target languages.

My ideas

- Offset->relationship and relationship -> offset?
- Automatic programming?
- Object Oriented Programming?

Thanks!