Bridging Semantic Gaps between Natural Languages and APIs with Word Embedding

Authors: Xiaochen Li¹, He Jiang¹, Yasutaka Kamei¹, Xin Chen²
¹Dalian University of Technology, ²Kyushu University, ³Hangzhou Dianzi University
A simple requirement “Read a File” may call many APIs to implement.

```java
public void readFile(String path) throws IOException {
    File file = new File(path);
    FileReader fr = new FileReader(file);
    BufferedReader br = new BufferedReader(fr);
    String line = "";
    while (null != (line = br.readLine())) {
        System.out.println(line);
    }
    br.close();
}
```
BACKGROUND

Semantic gaps negatively affect SE processes

Hinder developers from comprehending APIs and bring thousands of defects in API documents

A tool only returns 25.7% to 38.4% useful code snippets in top-10 results for user queries

Y. Zhou et al. “Analyzing APIs documentation and code to detect directive defects,” ICSE’17
F. Lv, et al. “Codehow: Effective code search based on API understanding and extended boolean model,” ASE’15
BACKGROUND

Semantic gaps negatively affect SE processes

Hinder developers from comprehending APIs and bring thousands of defects in API documents.

A tool only returns 25.7% to 38.4% useful code snippets in top 10 results for user queries.

How to bridge the gaps?

Semantic Estimation

Y. Zhou et al. “Analyzing APIs documentation and code to detect directive defects,” ICSE’17
F. Lv, et al. “Codehow: Effective code search based on API understanding and extended boolean model,” ASE’15
Classical algorithms to bridge the gaps

Calculating semantic relatedness / similarity between a word and an API or a set of words and APIs

**WordNet thesaurus analysis**

**Latent semantic analysis (LSA or LSI)**

**Co-occurrence analysis**

Section: Working with views and editors
Topic: Maximizing and minimizing in the eclipse presentation
Content: The **minimize** behavior for the Editor Area is somewhat different; **minimizing** the Editor Area results in a trim stack containing only a **placeholder icon** representing the entire editor area rather than icons for each open editor...
Motivation

Word embedding for word representing

A co-occurrence based algorithm. Generating low-dimensional vectors for words or APIs.

Prepare a dataset

Prepare word2vec tool

Run CBOW or Skip-gram model

Tool for computing continuous dis
Continuous Bag-Of-Words model CBOW

Motivation

Generate vectors of center words with their surrounding context

Mathematical formulation:

\[ L_M = \frac{1}{X} \sum_{x=1}^{X} \log p(w_x | W_x^d) \]

Example one-hot vector for the word “current”:

\{begin, xx, ..., current, ..., xx, end\}
{ 0, 0, ..., 1, ..., 0, 0 }
MOTIVATION

There are challenges for word-API learning

**Acquisition challenge**
collect large numbers of documents that contain diversity words and APIs

**Alignment challenge**
align words and APIs to fully mine their overall relationship in a window

---

Interface IPagination
Description: A page layout defines the initial layout for a perspective within a page in a workbench window... View placeholders may also have a secondary id. ... For example, the placeholder “someView:*” will match any occurrence of the view that has primary id “someView” and that also has some non-null secondary id. Note that this placeholder will not match the view if it has no secondary id...

DocumentView#getDefaultView()?
ComponentView#new()?

No word-API collocations
FRAMEWORK

Address the challenges by our model Word2API

A. Collect source code & APIs from GitHub (Acquisition)

B. Pre-process words & APIs with NLP and AST (Alignment)

C. Shuffle words & APIs in a method (Alignment)

D. Run word embedding to generate vectors

Word2API Model

Data acquisition

Word-API tuple construction

Training set creation

Vector generation
**FRAMEWORK**

**A. Collect source code & APIs from GitHub**

GitHub from 2008-2016
391,690 Java projects
31,211,030 source code files

Source code and comments

```java
/**
 * Open a file and output the contents
 * @param path
 * @return
 * @throws IOException
 */
public void readFile(String path) throws IOException {
    File file = new File(path);
    FileReader fr = new FileReader(file);
    BufferedReader br = new BufferedReader(fr);
    String line = "";
    while (null != (line = br.readLine())) {
        System.out.println(line);
    }
}
```

Extract words in the **method comment** and API calls in the **method body**.

These words and APIs are widely used by developers.
B. Pre-process words & APIs with NLP and AST

NLP for comments
/**
 * Open a file and output the contents
 * @param path
 * @return
 * @throws IOException
 */

Word Sequence: open -> file -> output...

API Sequence: File#new -> ... -> BufferedReader#readLine -> ...

Word sequence
< open, file, output, content >

13,883,230 word-API tuples
<word1, word2, ..., API1, API2...>

Tokenization
Stop word removal
Stemming

AST
Find API fully qualified names

expr
+ term println("+")
2 println("2")

expr

expr

expr

AST for source code
C. Shuffle words & APIs in a method

Problem of word-API tuples:
Words & APIs do not appear within each other’s window

<open, file, output, contents, ..., BufferedReader#readLine>

Shuffling strategy:
Words & APIs in the same word-API tuple contain valuable semantic information (relatedness) for mining
C. Shuffle words & APIs in a method

Word-API tuples
\(<\text{open, file, output, contents, …, BufferedReader}\#\text{readLine}>\>

- Shuffled Results
  \(<\text{open, File}\#\text{new, file, BufferedReader}\#\text{readLine}…>\>
  \(<\text{open, file, File}\#\text{new, BufferedReader}\#\text{readLine}…>\>
  \(<\text{file, File}\#\text{new, open, BufferedReader}\#\text{readLine}…>\>
  \(<\text{BufferedReader}\#\text{readLine, file, open, File}\#\text{new}…>\>
  \ldots\ldots

Increase the information interaction (collocations of words & APIs)

Help word embedding learn the knowledge of word-API (the overall knowledge of each tuple)

138,832,300 shuffled results (>30 GB)
D. Run word embedding to generate vectors

- 87,270 word vectors
- 37,431 API vectors

Semantic estimation with these vectors

- Word-API similarity
  \[ \text{sim}(w, a) = \frac{\vec{V}_w \cdot \vec{V}_a}{|\vec{V}_w| |\vec{V}_a|}. \]

- Words-APIs similarity
  \[ \text{sim}(W, A) = \frac{1}{2} \left( \sum \left( \text{maxSim}(w, A) \times \text{idf}(w) \right) \right) \left( \sum \text{idf}(w) \right) + \frac{1}{2} \left( \sum \left( \text{maxSim}(a, W) \times \text{idf}(a) \right) \right) \left( \sum \text{idf}(a) \right), \]
EVALUATION

Recommend APIs by a query word

- Random selected 50 Noun. & Verb.

<table>
<thead>
<tr>
<th>#</th>
<th>Word</th>
<th>#</th>
<th>Word</th>
<th>#</th>
<th>Word</th>
<th>#</th>
<th>Word</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>agent</td>
<td>11</td>
<td>delete</td>
<td>21</td>
<td>key</td>
<td>31</td>
<td>random</td>
</tr>
<tr>
<td>2</td>
<td>average</td>
<td>12</td>
<td>display</td>
<td>22</td>
<td>length</td>
<td>32</td>
<td>remote</td>
</tr>
<tr>
<td>3</td>
<td>begin</td>
<td>13</td>
<td>environment</td>
<td>23</td>
<td>mp3</td>
<td>33</td>
<td>request</td>
</tr>
<tr>
<td>4</td>
<td>buffer</td>
<td>14</td>
<td>file</td>
<td>24</td>
<td>next</td>
<td>34</td>
<td>reserve</td>
</tr>
<tr>
<td>5</td>
<td>capital</td>
<td>15</td>
<td>filter</td>
<td>25</td>
<td>node</td>
<td>35</td>
<td>scale</td>
</tr>
<tr>
<td>6</td>
<td>check</td>
<td>16</td>
<td>graphics</td>
<td>26</td>
<td>object</td>
<td>36</td>
<td>select</td>
</tr>
<tr>
<td>7</td>
<td>classname</td>
<td>17</td>
<td>http</td>
<td>27</td>
<td>open</td>
<td>37</td>
<td>session</td>
</tr>
<tr>
<td>8</td>
<td>client</td>
<td>18</td>
<td>input</td>
<td>28</td>
<td>parse</td>
<td>38</td>
<td>startup</td>
</tr>
<tr>
<td>9</td>
<td>current</td>
<td>19</td>
<td>interrupt</td>
<td>29</td>
<td>port</td>
<td>39</td>
<td>string</td>
</tr>
<tr>
<td>10</td>
<td>day</td>
<td>20</td>
<td>iter</td>
<td>30</td>
<td>post</td>
<td>40</td>
<td>system</td>
</tr>
</tbody>
</table>

- Comparative algorithm

- LSA (Latent Semantic Analysis)
- PMI (Pointwise Mutual Information)
- NSD (Normalized Software Distance)
- HAL (Hyperspace Analogue to Language)
EVALUATION

Word2API captures the relatedness of words & APIs

- Word2API **outperforms the baseline algorithms**;
- Volunteers judgement the relatedness between the words and the recommended APIs by different algorithms.
**Evaluation**

Window size and the shuffling strategy

- Increase window size, but performance drops.
- Cannot learn word embedding by simply increasing window size $(w)$.
- Sequence-$w5$: no shuffling, window size $= 5$.
- Shuffling is significantly better than no shuffling.

(a) Precision@$k$

(b) Precision@$k$ (different models)
EVALUATION

Window size and the shuffling strategy

Input  Hidden layer  Output

$w_{x-d}$  $w_{y_{-c}}$  $w_x$

$w_{x-1}$  $w_{x+1}$  $w_x$

$w_{x+d}$

Example of one-hot vector for word “current”
\{begin, xx, ..., current, ..., xx, end\}
\{0, 0, ..., 1, ..., 0, 0\}

Sequence-w5: no shuffling, window_size=5

Shuffling is significantly better than no shuffling
APPLICATION 1

Expand user query into an API vector for API sequences recommendation

Query

“read a file”

API Sequence

- `java.io.File#new`
- `java.io.FileReader#new`
- `java.io.BufferedReader#new`
- `java.io.BufferedReader#readLine`

These sequences are retrieved from source code corpus, e.g. GitHub corpus.
**APPLICATION 1**

Expand user query into an API vector for API sequences recommendation

**Query**

"read a file"

**API vector**

```
API_1  API_2  API_1  API_4  API_3
```

**API Sequence**

- java.io.File#new,
- java.io.FileReader#new,
- java.io.BufferedReader#new,
- java.io.BufferedReader#readLine

These sequences are retrieved from source code corpus, e.g. GitHub corpus.

- **SWIM**: Word Alignment based Augmentation
- **CodeHow**: API Description based Augmentation
- **Word2API** based Augmentation
## APPLICATION 1

TABLE III: Performance of query augmentation algorithms over 30 human written queries.

<table>
<thead>
<tr>
<th>ID</th>
<th>Query</th>
<th>FR</th>
<th>RR5</th>
<th>RR10</th>
<th>FR</th>
<th>RR5</th>
<th>RR10</th>
<th>FR</th>
<th>RR5</th>
<th>RR10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>convert int to string</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0.2</td>
<td>0.1</td>
</tr>
<tr>
<td>Q2</td>
<td>convert string to int</td>
<td>1</td>
<td>1</td>
<td>0.5</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>Q3</td>
<td>append string</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Q4</td>
<td>get current time</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Q5</td>
<td>parse datetime from string</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.7</td>
</tr>
<tr>
<td>Q6</td>
<td>test file exists</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.8</td>
<td>0.7</td>
<td>1</td>
<td>0.4</td>
<td>0.7</td>
</tr>
<tr>
<td>Q7</td>
<td>open a url</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.8</td>
<td>0.9</td>
<td>1</td>
<td>1</td>
<td>0.9</td>
</tr>
<tr>
<td>Q8</td>
<td>open file dialog</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Q9</td>
<td>get files in folder</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Q10</td>
<td>match regular expressions</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Q11</td>
<td>generate md5 hash code</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Q12</td>
<td>generate random number</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Q13</td>
<td>round a decimal value</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>Q14</td>
<td>copy a file and save it to your destination path</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.8</td>
<td>0.9</td>
</tr>
<tr>
<td>Q15</td>
<td>delete files and folders in a directory</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.9</td>
</tr>
<tr>
<td>Q16</td>
<td>reverse a string</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.9</td>
</tr>
<tr>
<td>Q17</td>
<td>create socket</td>
<td>1</td>
<td>0.6</td>
<td>0.4</td>
<td>4</td>
<td>0.4</td>
<td>0.4</td>
<td>4</td>
<td>0.4</td>
<td>0.4</td>
</tr>
<tr>
<td>Q18</td>
<td>rename a file</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.9</td>
</tr>
<tr>
<td>Q19</td>
<td>download file from url</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.9</td>
</tr>
<tr>
<td>Q20</td>
<td>serialize an object</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.9</td>
</tr>
<tr>
<td>Q21</td>
<td>read binary file</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.9</td>
</tr>
<tr>
<td>Q22</td>
<td>save an image to a file</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.9</td>
</tr>
<tr>
<td>Q23</td>
<td>write an image to a file</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.9</td>
</tr>
<tr>
<td>Q24</td>
<td>parse xml</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.9</td>
</tr>
<tr>
<td>Q25</td>
<td>play audio</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.9</td>
</tr>
<tr>
<td>Q26</td>
<td>play the audio clip at the specified absolute URL</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.9</td>
</tr>
<tr>
<td></td>
<td><strong>Average scores over 30 queries</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>FR</td>
<td>5.633</td>
<td>0.513</td>
<td>0.463</td>
<td>4.467</td>
<td>0.547</td>
<td>0.533</td>
<td>1.933</td>
<td>0.680</td>
<td>0.677</td>
</tr>
</tbody>
</table>

Position of first correct API seq.: lower is better
Ratio of correct API seq.: higher is better
**Question**: "Are there any good CachedRowSet implementations other than the proprietary Sun one?"

**Answer**: You shouldn't be directly instantiating implementation of CachedRowSet -- use its Provider to obtain an instance: see [http://docs.oracle.com/javase/7/docs/api/javax/sql/rowset/RowSetProvider.html](http://docs.oracle.com/javase/7/docs/api/javax/sql/rowset/RowSetProvider.html) (available since JDK7).

In fact, CachedRowSet's interface and related factory are standard/portable.
APPLICATION 2

Word2API for API Documents Linking

● Collect words in the question "Are there any good CachedRowSet implementations other than the proprietary Sun one?"

1. Transform words and APIs into vectors with Word2API
2. Rank API documents by words-APIs similarity

● Collect APIs in each API document
  • `javax.sql.rowset.RowSetProvider#newFactory`
  • `javax.sql.rowset.RowSetProvider#createCachedRowSet`
  • .......

\[
sim(W, A) = \frac{1}{2} \left( \frac{\sum \maxSim(w, A) \times idf(w)}{\sum \idf(w)} + \frac{\sum \maxSim(a, W) \times idf(a)}{\sum \idf(a)} \right).
\]
APPLICATION 2

Word2API can bridge the gaps of documents

- MAP: Mean Average Precision
- MRR: Mean Reciprocal Rank

Algorithms
- VSM: vector space model
- Embedding: previous work
- VSM+XXX: combined

Word embedding is better than VSM
We can combine Word2API with other techniques for better results

Ye et al. From Word Embeddings To Document Similarities for Improved Information Retrieval in Software Engineering (ICSE16)
CONCLUSION

- We propose Word2API to solve the problem of constructing low-dimensional representations for both words and APIs simultaneously.

- With Word2API, we generate 126,853 word and API vectors to bridge the semantic gaps between natural language words and APIs.

- We show two applications of Word2API. Word2API improves the performance of two typical software engineering tasks, i.e., API sequences recommendation and API documents linking.
Bridging Semantic Gaps between Natural Languages and APIs with Word Embedding

Thanks

Reporters: Xiaochen Li
Dalian University of Technology, China

Authors: Xiaochen Li\textsuperscript{1}, He Jiang\textsuperscript{1}, Yasutaka Kamei\textsuperscript{1}, Xin Chen\textsuperscript{2}
\textsuperscript{1}Dalian University of Technology, \textsuperscript{2}Kyushu University, \textsuperscript{3}Hangzhou Dianzi University