Browsing NPM packages more effectively with Code Compass

Tom Van Cutsem
@tvcutsem
Fact: software ecosystems are rapidly expanding

(source: GitHub Octoverse and modulecounts.com, May 2019)
NPM dominates

(source: modulecounts.com, June 2019)
That’s “awesome”, but...

**New problem**: how do you find relevant libraries for your development needs?

- **Today**: manual way of dealing with this
  - **“Awesome” Lists** – community-curated lists of categorized libraries
  - **LibHunt** – website built on top of awesome lists
    - Indexed 24K libraries
    - Into 3K categories
- **But hardly Scalable**...
  - Top 6 languages have over **1.5 Million libraries**
  - Only 1.6% is covered by manual indexing efforts
Code Compass to the rescue
Unsupervised learning from Big Code

**Extract** library dependency data from code

**Learn** vector representation

**Compute** nearest neighbors
Doesn’t Code Compass just recommend popular combo’s? **No!**

<table>
<thead>
<tr>
<th>Search anchor</th>
<th>Code Compass top results</th>
<th>Most Popular combos</th>
</tr>
</thead>
<tbody>
<tr>
<td>mysql</td>
<td>pg (#299) redis (#117) knex (#343) mongodb (#97) nodemailer (#153)</td>
<td>express (#3) body-parser (#13) async (#25) lodash (#12) request (#17)</td>
</tr>
<tr>
<td>gm (graphicsmagic)</td>
<td>imagemagick (#1517) sharp (#1040) connect-busboy (#1913) jump (#1010) canvas (#350)</td>
<td>async (#25) request (#17) express (#3) lodash (#12) crypto (#16)</td>
</tr>
</tbody>
</table>
Mapping the JavaScript library ecosystem

2D projection of a 100D space

express
body-parser
vue
react

Vue
React
Koa
Gulp
Webpack
Local neighborhoods reveal meaningful similarities between libraries
Local neighborhoods reveal meaningful similarities between libraries
Data

GitHub+NPM JS/TS projects crawled 764K
Source files *.js, ts* crawled 20.4M
Unique require/import statements crawled 216K

Libraries indexed by import2vec 88.4K
Fun fact: how many modules get imported in a typical JS file?
Fun fact: how many modules get imported in a typical JS file?

- Answer: about 4 on average
Fun fact: module imports follow Zipf’s Law

- The 2\textsuperscript{nd} most popular module gets imported only half as much as the most popular one.
- The \( n \)\textsuperscript{th} most popular module gets imported only \( \sim \frac{1}{n} \) as much as the most popular one.

Similar to frequency of letters in alphabet, words in text documents, ...
Hunger for more details? Read our paper

**Import2vec**
Learning Embeddings for Software Libraries

<table>
<thead>
<tr>
<th>Bart Theeten</th>
<th>Frederik Vandeputte</th>
<th>Tom Van Cutsem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nokia Bell Labs</td>
<td>Nokia Bell Labs</td>
<td>Nokia Bell Labs</td>
</tr>
<tr>
<td>Antwerp, Belgium</td>
<td>Antwerp, Belgium</td>
<td>Antwerp, Belgium</td>
</tr>
<tr>
<td><a href="mailto:bart.theeten@nokia-bell-labs.com">bart.theeten@nokia-bell-labs.com</a></td>
<td><a href="mailto:frederik.vandeputte@nokia-bell-labs.com">frederik.vandeputte@nokia-bell-labs.com</a></td>
<td><a href="mailto:tom.van.cutsem@nokia-bell-labs.com">tom.van.cutsem@nokia-bell-labs.com</a></td>
</tr>
</tbody>
</table>

**Abstract:** We consider the problem of developing suitable learning representations (embeddings) for library packages that capture semantic similarity among libraries. Such representations are known to improve the performance of downstream learning tasks (e.g., classification) or applications such as contextual search and analogical reasoning.

We apply word embedding techniques from natural language processing (NLP) to train embeddings for library packages (“library vectors”). Library vectors represent libraries by similar context of use as determined by import statements present in source code. Experimental results obtained from training such embeddings on three large open source software corpora reveal that library vectors capture semantically meaningful relationships among software libraries, such as the relationships between frameworks and their plug-ins and libraries commonly used together within ecosystems such as big data infrastructure projects (in Java), front-end and back-end web development frameworks (in JavaScript) and data science toolkits (in Python).

**Index Terms:** machine learning, software engineering, information retrieval

The size and scale of today’s software ecosystem suggests that a machine learning approach could help us build tools that help developers more effectively navigate them. However, for most learning algorithms to be applied successfully to this problem, we require a mathematical representation of libraries, preferably one that represents similar libraries by similar representations.

This paper addresses the question whether we can leverage techniques from natural language processing, in particular word embeddings, to learn meaningful distributed representations of software libraries from large codebases. Just like word embeddings learn to represent similar words by similar dense vector representations based on the words’ similar context of use, we aim to learn a dense vector representation of libraries.


(google “import2vec”)
Give Code Compass a try. Thanks!

- bell-labs.com/code-compass
- github.com/nokia/code-compass
- @tvcutsem