Browsing NPM packages more effectively with Code Compass

Tom Van Cutsem

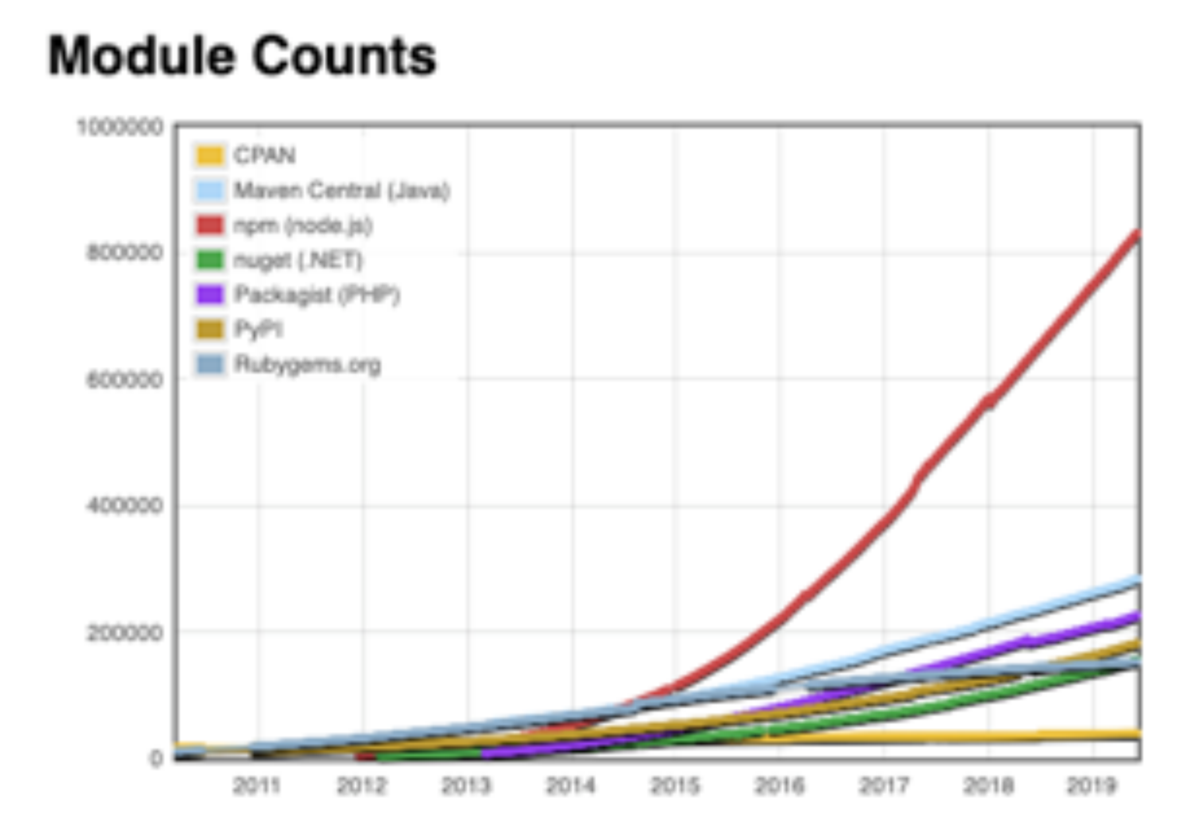
@tvcutsem

Fact: software ecosystems are rapidly expanding

JavaScript	Java	Python
820K+ packages	280K+ packages	180K+ packages
+454/day	+275/day	+130/day

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

NPM dominates

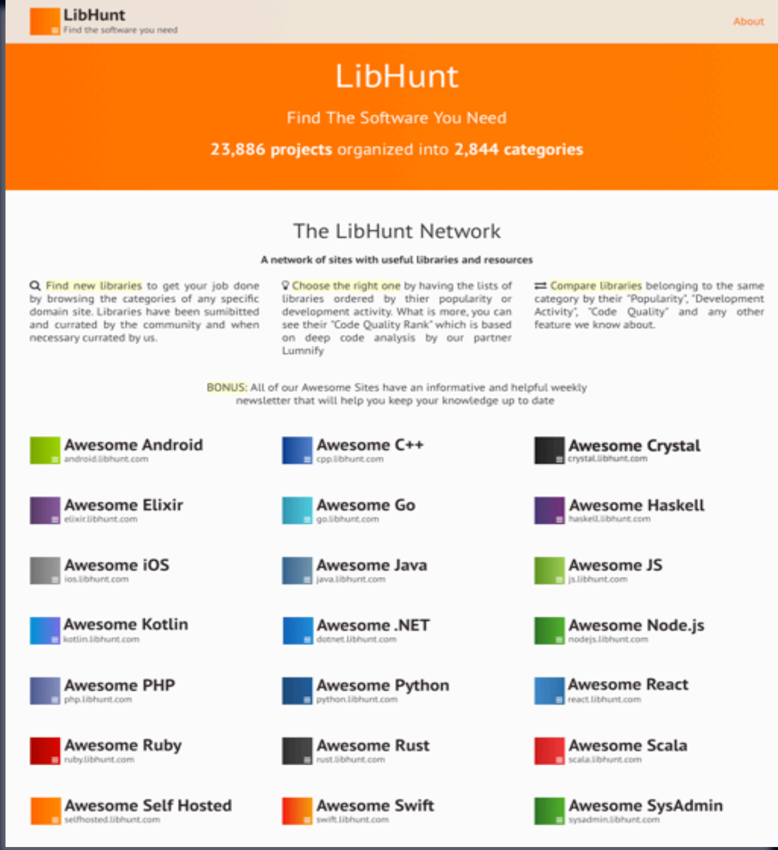


(source: modulecounts.com, June 2019)

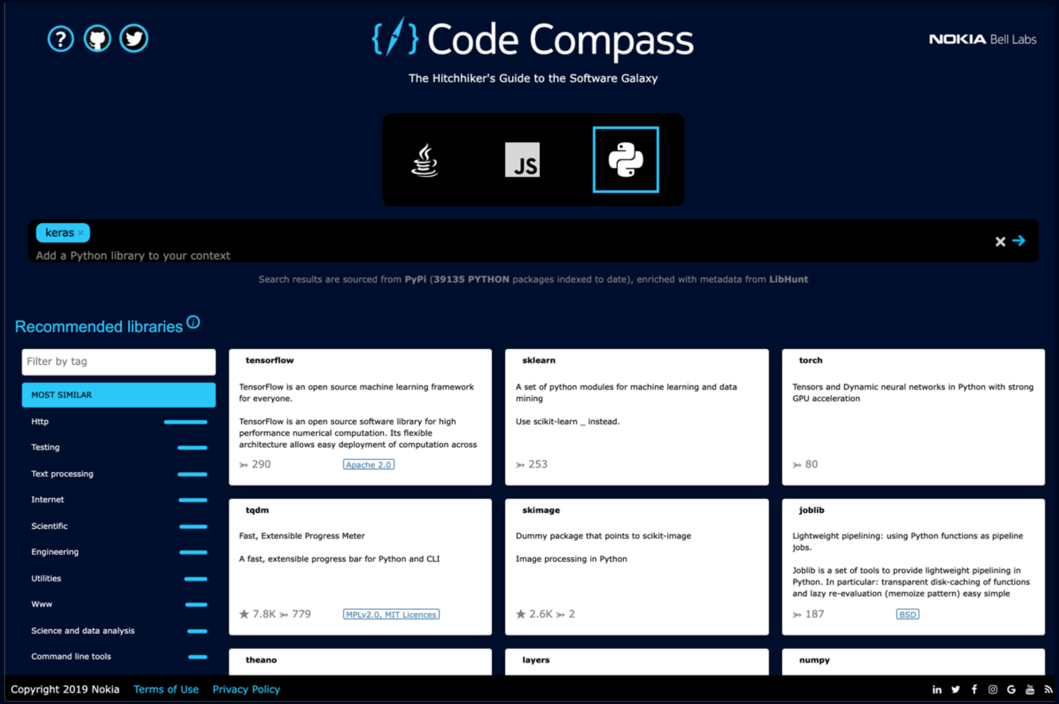
That’s “awesome”, but…

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 awesome
  - **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

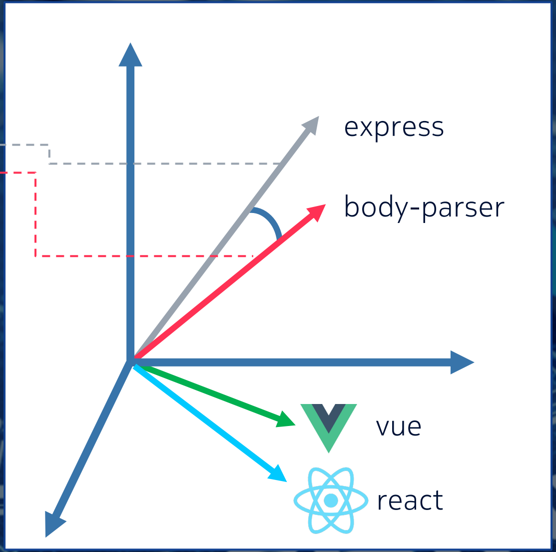


Unsupervised learning from Big Code

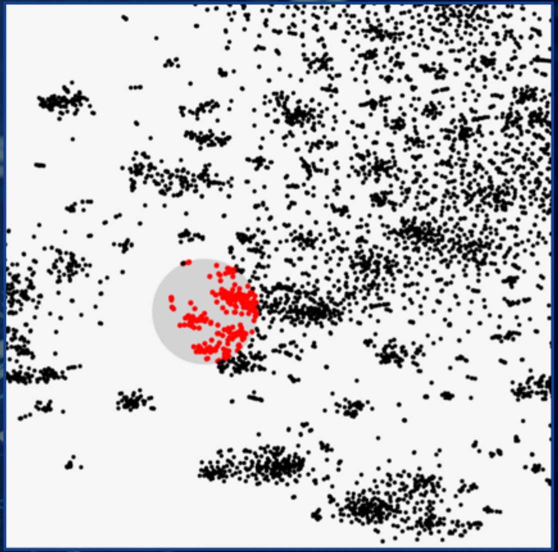
Extract library dependency data from code

```
// Dependencies
// ===============================================================
var express = require('express');
var body-parser = require('body-parser');
var path = require('path');

// Sets up the Express App
// ===============================================================
var app = express();
var PORT = 3000;

// Tells body-parser what type of content to receive
app.use(body-parser.json());
app.use(body-parser.urlencoded({ extended: true }));
app.use(body-parser.text());
app.use(body-parser({ type: 'application/vnd.api+json' }));
```

Learn vector representation



Compute nearest neighbors

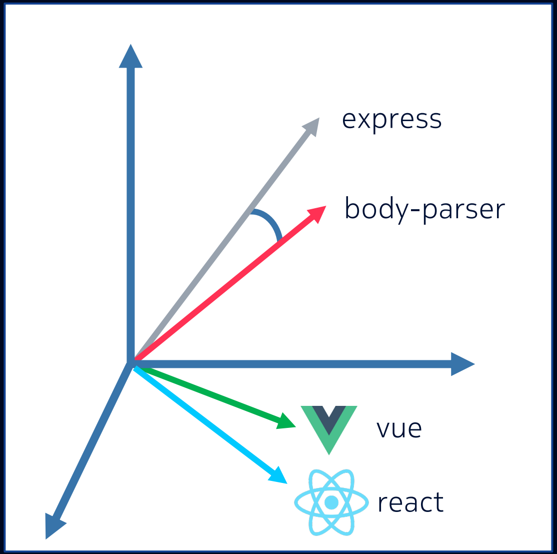


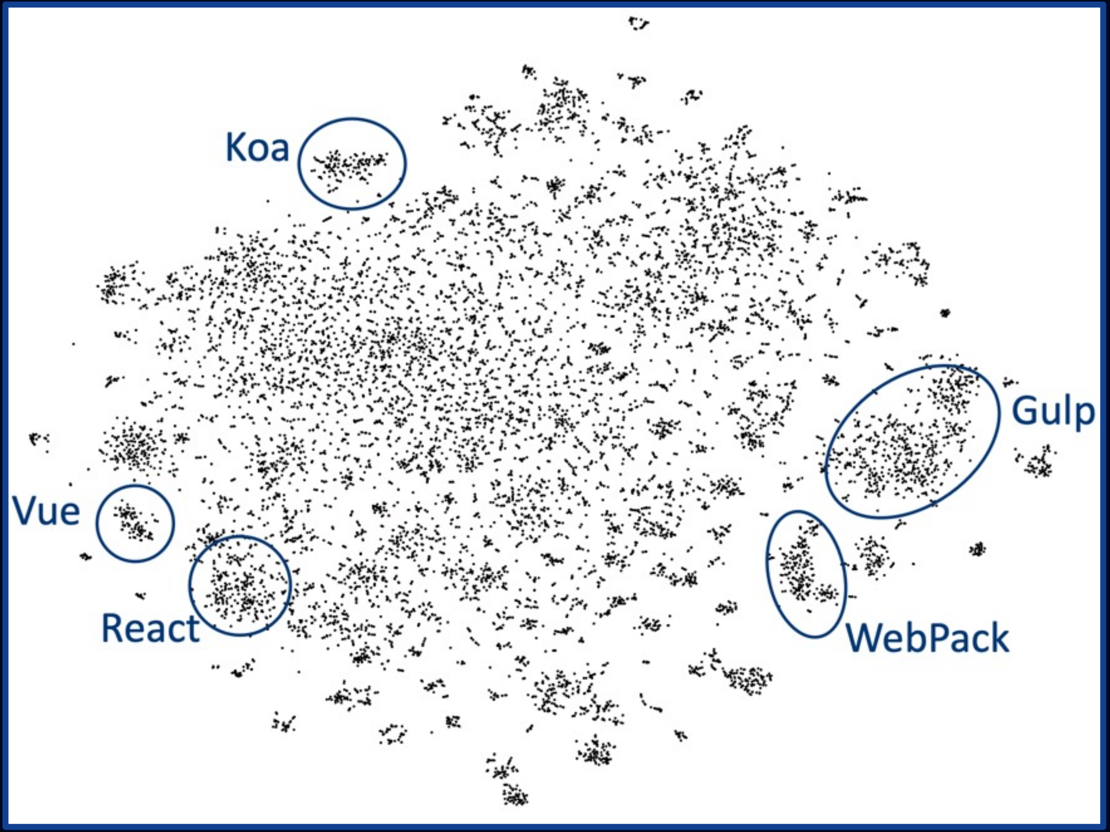
Doesn't Code Compass just recommend popular combo's? No!

Search anchor	Code Compass top results	Most Popular combos
`mysql`	`pg` (#299) `redis` (#117) `knex` (#343) `mongodb` (#97) `nodemailer` (#153)	`express` (#3) `body-parser` (#13) `async` (#25) `lodash` (#12) `request` (#17)
`gm` (graphicsmagic)	`imagemagick` (#1517) `sharp` (#1040) `connect-busboy` (#1913) `jimp` (#1010) `canvas` (#350)	`async` (#25) `request` (#17) `express` (#3) `lodash` (#12) `crypto` (#16)

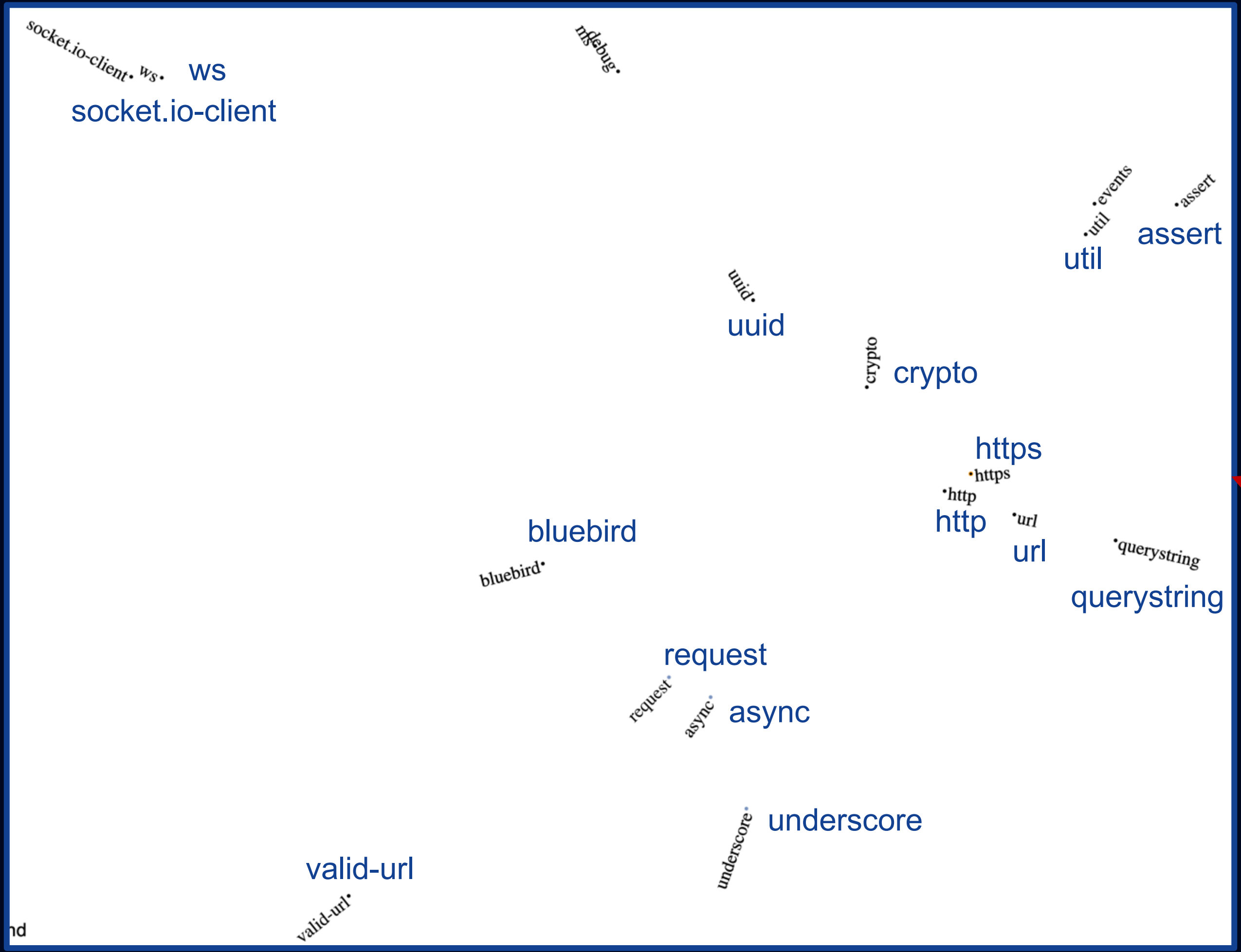
Mapping the JavaScript library ecosystem

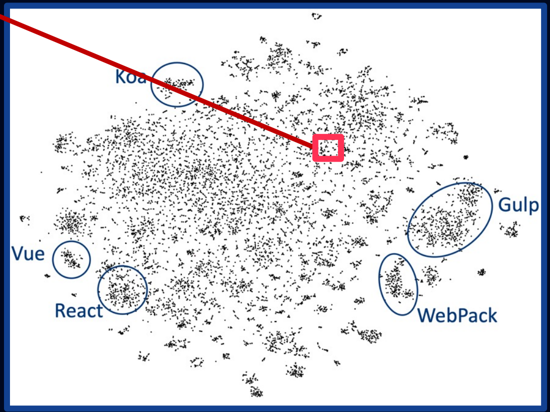
2D projection of a 100D space



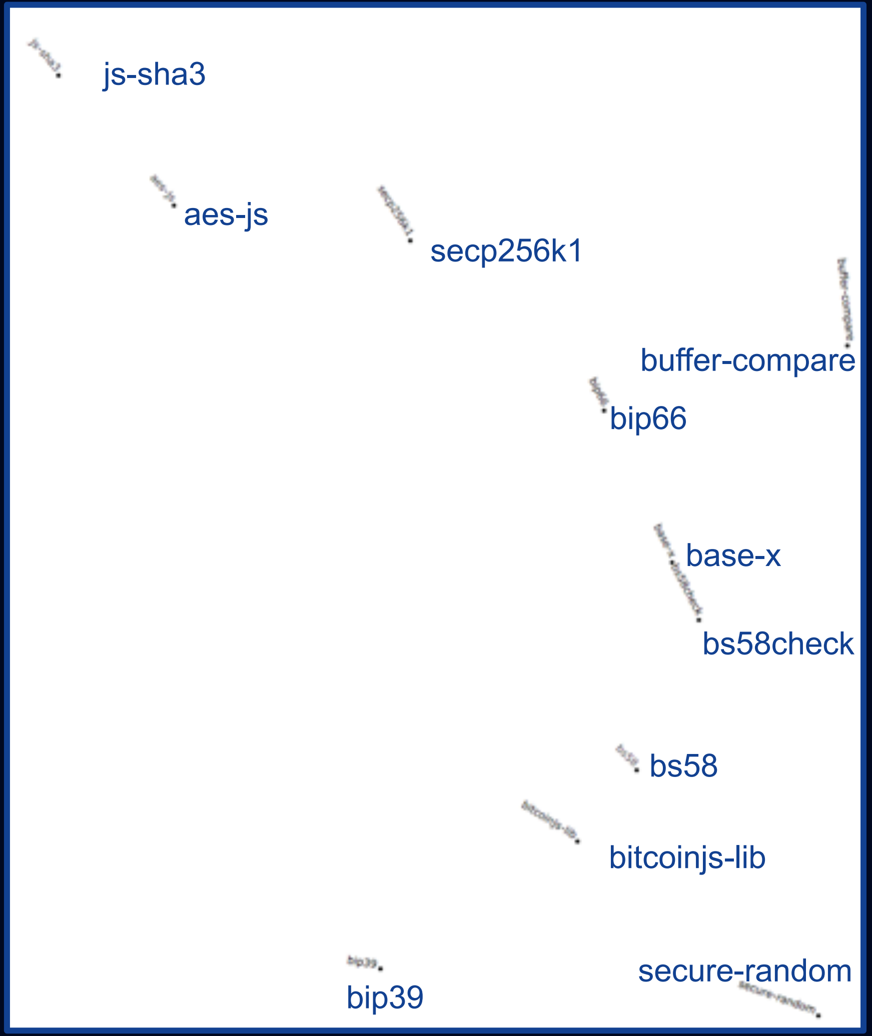


Local neighborhoods reveal meaningful similarities between libraries

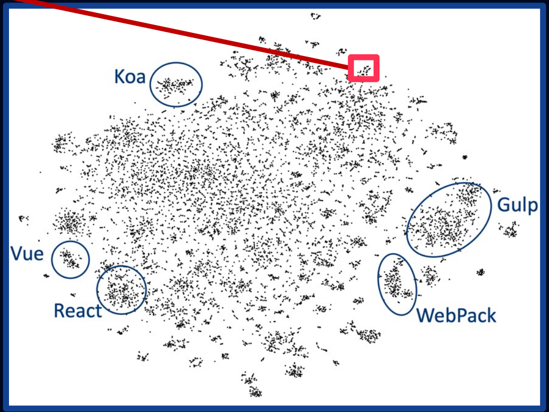




NOKIA Bell Labs



Local neighborhoods reveal meaningful similarities between libraries



NOKIA Bell Labs

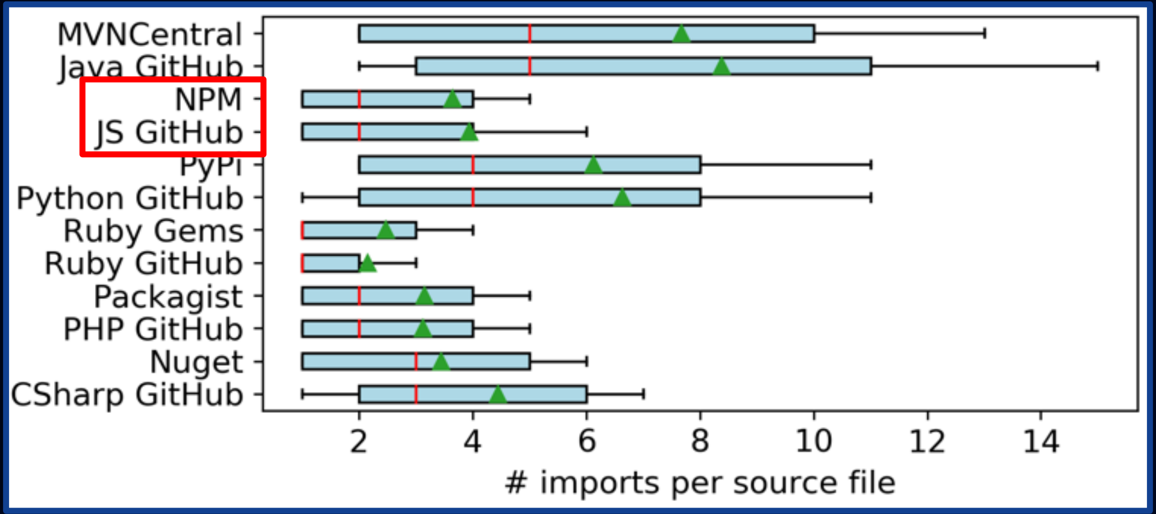
Data

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?

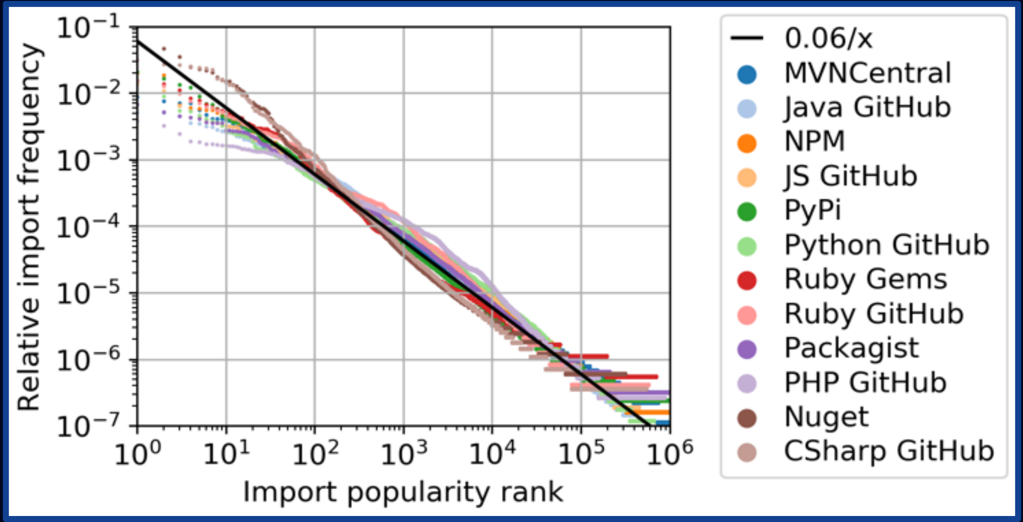
Fun fact: how many modules get imported in a typical JS file?

- Answer: about 4 on average



Fun fact: module imports follow Zipf's Law

- The 2nd most popular module gets imported only half as much as the most popular one
- The *n*th most popular module gets imported only ***~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

Bart Theeten
Nokia Bell Labs
Antwerp, Belgium
bart.theeten@nokia-bell-labs.com

Frederik Vandeputte
Nokia Bell Labs
Antwerp, Belgium
frederik.vandeputte@nokia-bell-labs.com

Tom Van Cutsem
Nokia Bell Labs
Antwerp, Belgium
tom.van_cutsem@nokia-bell-labs.com

***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 reveals that library vectors capture semantically meaningful relationships among software libraries, such as the relationship 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**

well exceeds over 1 million packages (data as of January 2019) and so most packages in the "long tail" remain undiscoverable through manual curation.

The size and scale of today's software ecosystems 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

https://arxiv.org/abs/1904.03990

(google "import2vec")

Code Compass

Contextual search for code

Give Code Compass a try. Thanks!

bell-labs.com/code-compass

github.com/nokia/code-compass

@tvcutsem

NOKIA Bell Labs

NOKIA