A Corpus Processing and Analysis Pipeline for Quickref

Antoine Hacquard & Didier Verna

LRDE
EPITA Research and Development Laboratory

14th European Lisp Symposium, May 3–4 2021
Quicklisp & Quickref

Quickref

Reference manuals for Quicklisp libraries

Quicklisp version 2021-04-11
Documentation generated with Quickref 3.0 "The Alchemists" / Deict 3.0 "Montgomery Scott"; 1981 manuals available.

Library Index

#

1am  3bgl-shader  3d-matrices
3b-bmfont  3bmd  3d-vectors
3b-hdr  3bz  cl-6502
3b-swf

A

a-cl-logger  cl-anonfun  asdf-finalizers
able  cl-ansi-term  asdf-linguist
Motivation

The project:
- A new keyword index for Quickref

Why not just use a modern search engine?
- Favor Quicklisp availability
- Natural emphasis on libraries with some documentation
- Other potential applications (word cloud, statistical / topic analysis, etc.)
Pipeline Overview

lib1 corpus

lib2 corpus

lib3 corpus

Tokenizer

POS-Tagger

Syntactic filter

Stemmer

Lemmatizer

Known lemmas

TF-IDF

Pertinence filter

Aggregator

keyword index

word cloud
Tokenization:

"this can can walk" $\rightarrow$ THIS | CAN | CAN | WALK

POS-tagging:

THIS (det.) | CAN (common noun) | CAN (verb) | WALK (verb)
Stemmer & Lemmatizer

Stemming:

argue | argued | argues | arguing $\rightarrow$ argu

Lemmatization:

argue | argued | argues | arguing $\rightarrow$ argue
TF-IDF

\[ TF('the') = 0.7; \quad IDF('the') = 1.9; \quad TD-IDF('the') = \frac{TF}{IDF} = 0.37 \]

\[ TF('temperature') = 1.6; \quad IDF('temperature') = 0.3; \quad TD-IDF('temperature') = \frac{TF}{IDF} = 5.33 \]
Aggregator

lib1 corpus → Tokenizer → POS-Tagger → Syntactic filter → Lemmatizer → Known lemmas → Pertinence filter → Aggregator → Keyword index

lib2 corpus → Stemmer → Lemmatizer → Known lemmas → Pertinence filter → Aggregator → …

lib3 corpus → TF-IDF → Aggregator → Word cloud

Keyword Index

- file
- document
- string
- test
- system
- stream
- client
- message
Out of Dictionary Words

arguing → stemming → argu

argu → dictionary

argu → argmax

argu → argue

dictionary

argue

cat

forest

edit distance

0.9

0.6

0.5
Out of Dictionary Words

Words absent from the dictionary will match awkwardly!
Custom Dictionary Generation

- Grab the *whole* corpus
- Lemmatize with an external lemmatizer (NLTK in our case)
- Use this as new dictionary
Pros & Cons

Pros:
- Custom dictionary with words from our corpus *only*

Cons:
- Words are potentially badly lemmatized
  - Potential solution: test and incorporate CLHS glossary
- Requires an external lemmatizer
  - But just once for every other pipeline run
Experimentation with Aggregators

The Quickref NLP Pipeline

TF-IDF output
- temperature
- weight
- unit

Keyword index
- weight
- temperature
- test
- option
- unit

Histogram
Other Potential Aggregators: Top-Down

- Rank output of TF-IDF with a pertinence score (e.g. mean of TF-IDF values), and keep just enough keywords to reach full library coverage.

<table>
<thead>
<tr>
<th>TF-IDF output</th>
<th>Pertinence score</th>
<th>Keyword index</th>
</tr>
</thead>
<tbody>
<tr>
<td>lib1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>temperature (0.8)</td>
<td></td>
<td>temperature</td>
</tr>
<tr>
<td>weight (0.72)</td>
<td></td>
<td>option</td>
</tr>
<tr>
<td>unit (0.59)</td>
<td></td>
<td>test</td>
</tr>
<tr>
<td>lib2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>temperature (0.8)</td>
<td></td>
<td>unit</td>
</tr>
<tr>
<td>weight (0.72)</td>
<td></td>
<td>test</td>
</tr>
<tr>
<td>unit (0.59)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lib3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>temperature (0.8)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>weight (0.72)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>unit (0.59)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Other Potential Aggregators: Bottom-Up

- Start from keywords with the fewest associated libraries, and take until full library coverage is achieved.
A 4-stages modular NLP pipeline for Quickref

- First 3 blocks completed, to be released as standalone open-source libraries
- Aggregation block still work in progress

Suggestions / ideas welcome!

Thank you!