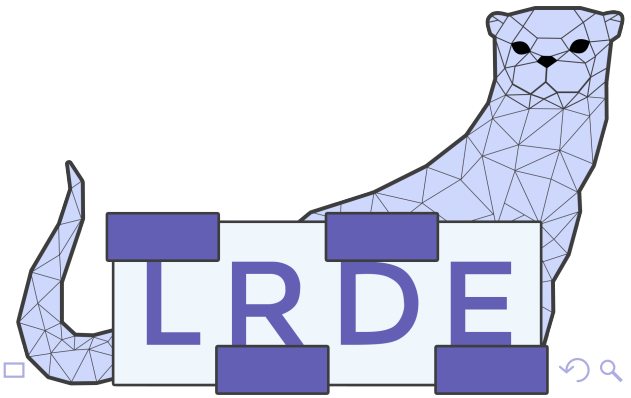
A Corpus Processing and Analysis Pipeline for Quickref

Antoine Hacquard & Didier Verna

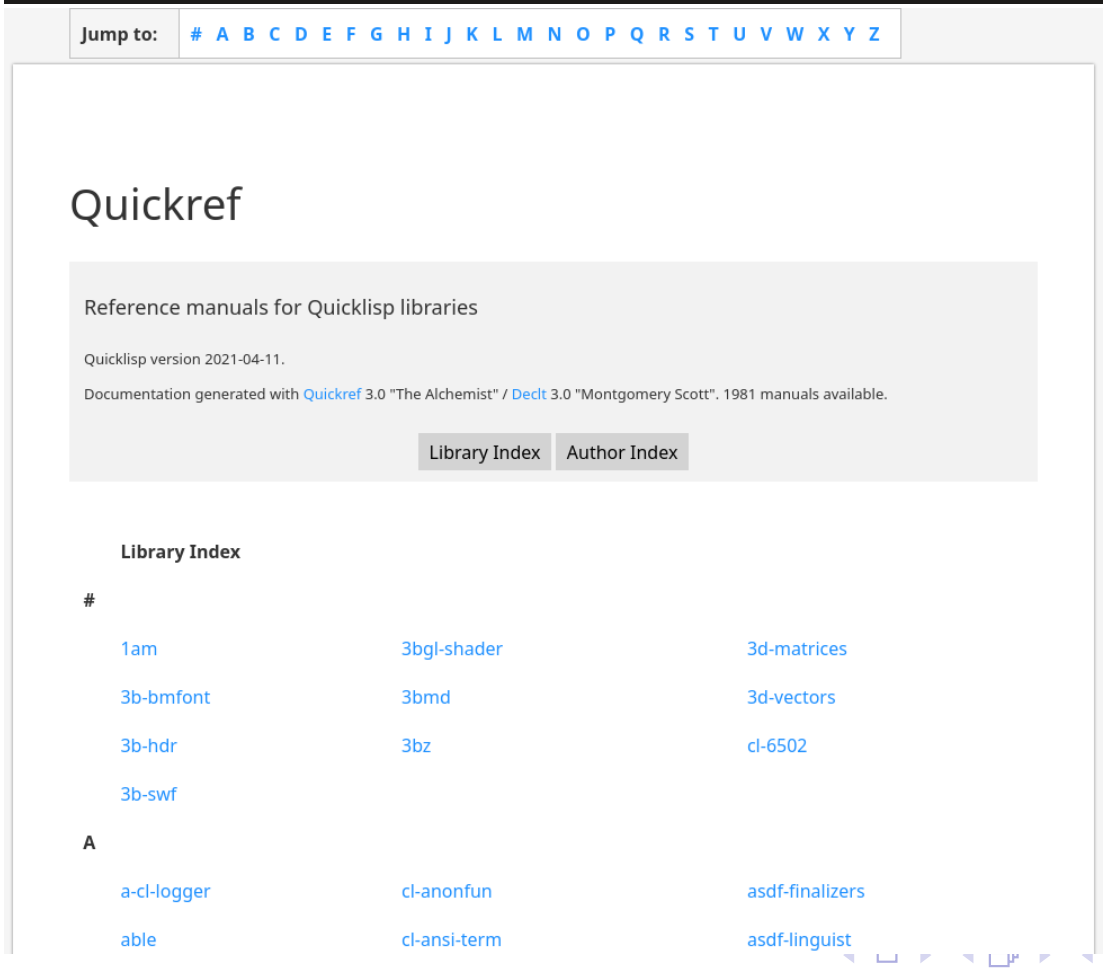
LRDE
EPITA Research and Development Laboratory

14th European Lisp Symposium, May 3–4 2021



Quicklisp & Quickref

```
* (ql:quickload :fr.epita.lrde.quickref)
To load "fr.epita.lrde.quickref":
  Load 1 ASDF system:
    fr.epita.lrde.quickref
; Loading "fr.epita.lrde.quickref"
.....
(:FR.EPITA.LRDE.QUICKREF)
```



Jump to: # A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Quickref

Reference manuals for Quicklisp libraries

Quicklisp version 2021-04-11.

Documentation generated with Quickref 3.0 "The Alchemist" / Declt 3.0 "Montgomery Scott". 1981 manuals available.

Library Index | Author Index

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

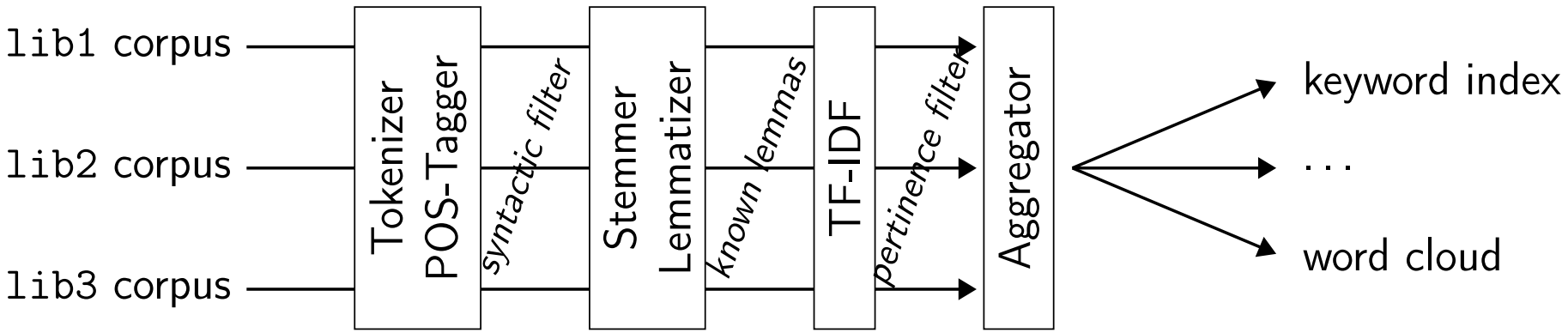
The project:

- A new keyword index for Quickref

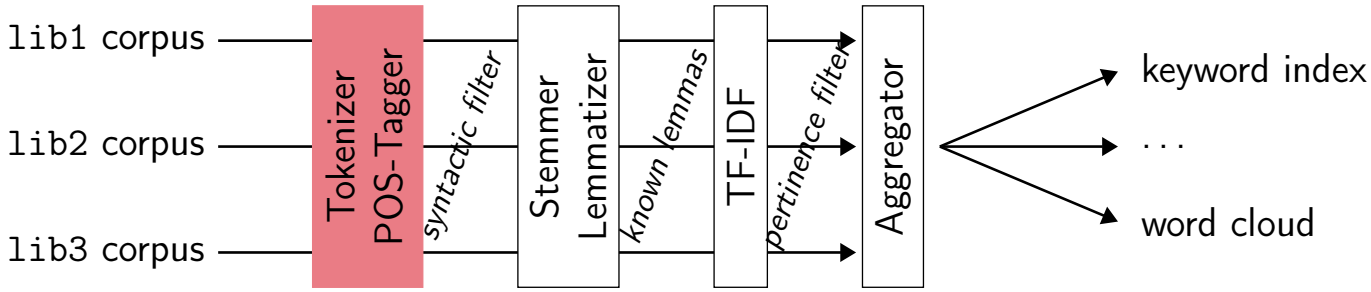
Why not just use a modern search engine?

- Favor Quicklisp availability
- Natural emphasis on libraries with *some* documentation
- Other potential applications (word cloud, statistical / topic analysis, *etc.*)

Pipeline Overview



Tokenizer & POS-tagger



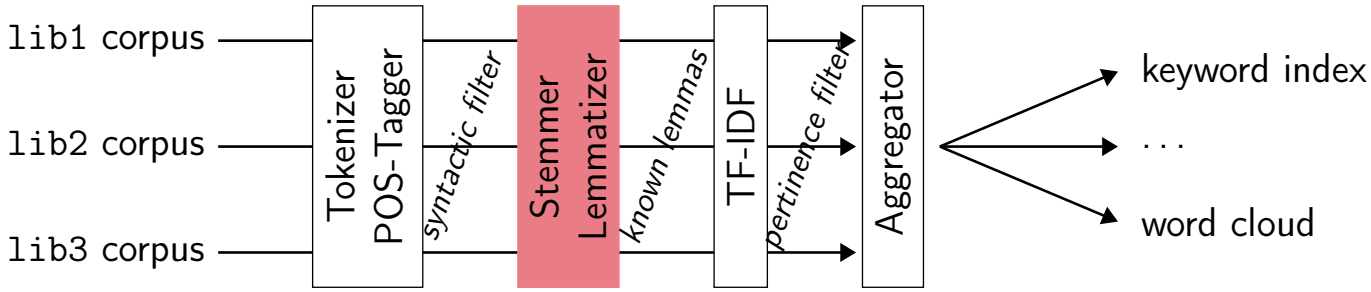
Tokenization :

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

POS-tagging :

THIS (det.) | CAN (common noun) | CAN (verb) | WALK (verb)

Stemmer & Lemmatizer



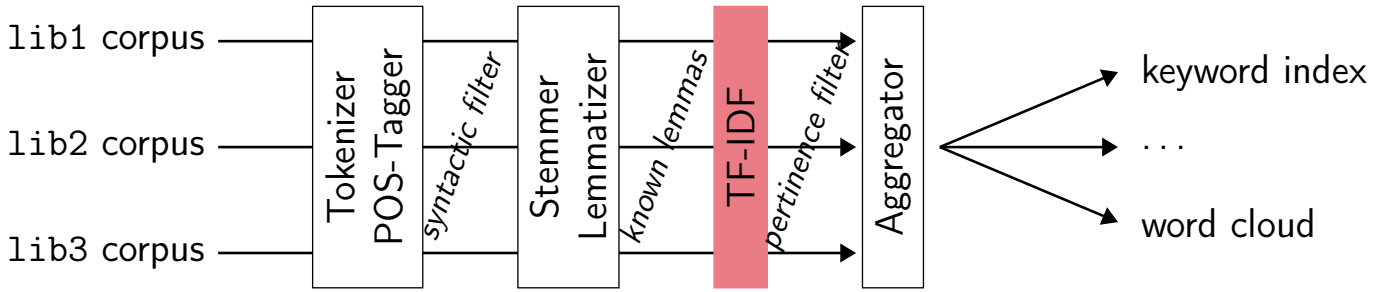
Stemming :

$$\text{argue} \mid \text{argued} \mid \text{argues} \mid \text{arguing} \Longrightarrow \text{argu}$$

Lemmatization :

$$\text{argue} \mid \text{argued} \mid \text{argues} \mid \text{arguing} \Longrightarrow \text{argue}$$

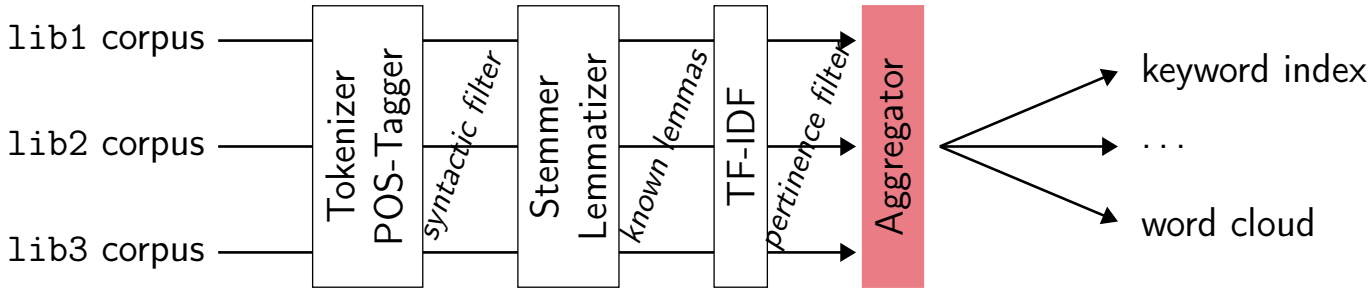
TF-IDF



$$\textbf{TF}(\text{'the'}) = 0.7;\ \textbf{IDF}(\text{'the'}) = 1.9;$$
$$\textbf{TD-IDF}(\text{'the'}) = \frac{TF}{IDF} = 0.37$$

$$\textbf{TF}(\text{'temperature'}) = 1.6;\ \textbf{IDF}(\text{'temperature'}) = 0.3;$$
$$\textbf{TD-IDF}(\text{'temperature'}) = \frac{TF}{IDF} = 5.33$$

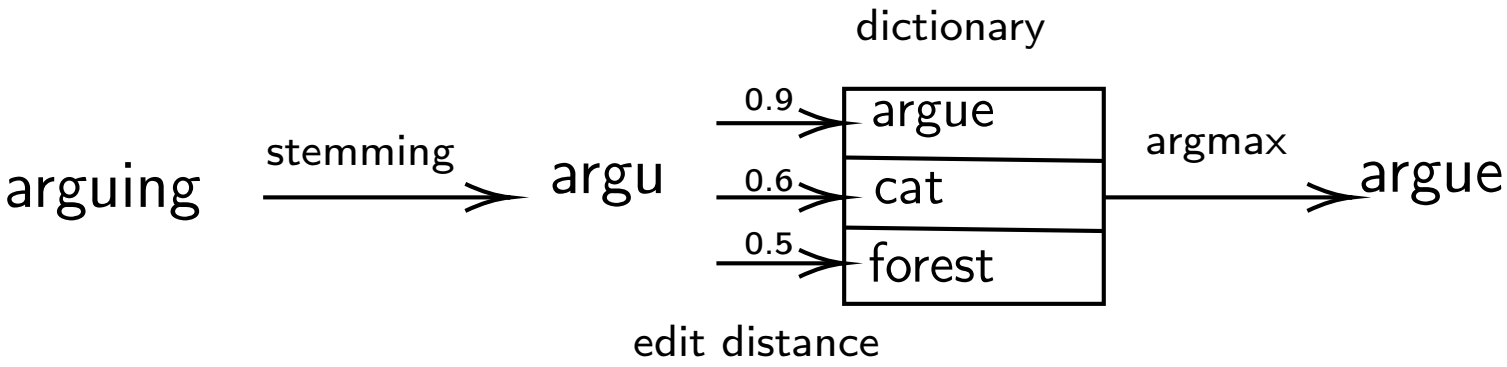
Aggregator



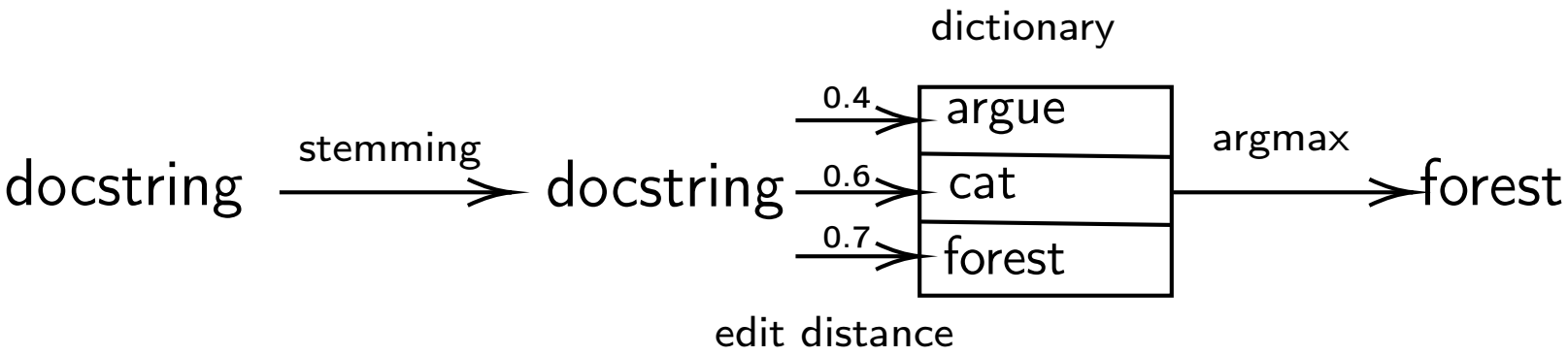
Keyword Index

- file
- document
- string
- test
- system
- stream
- client
- message

Out of Dictionary Words



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

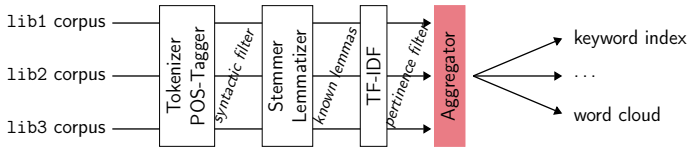
Pros:

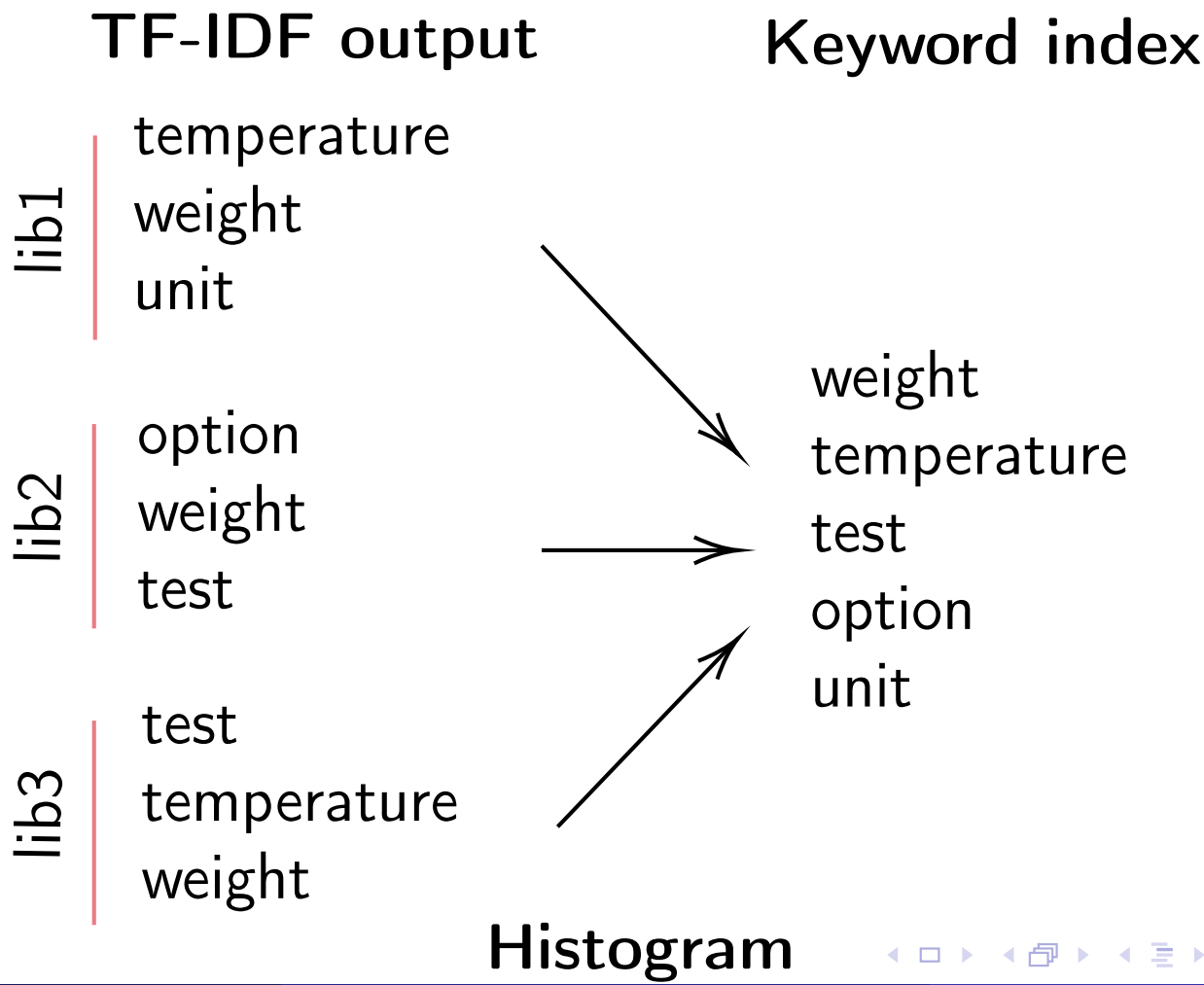
- Custom dictionary with words from our corpus *only*

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

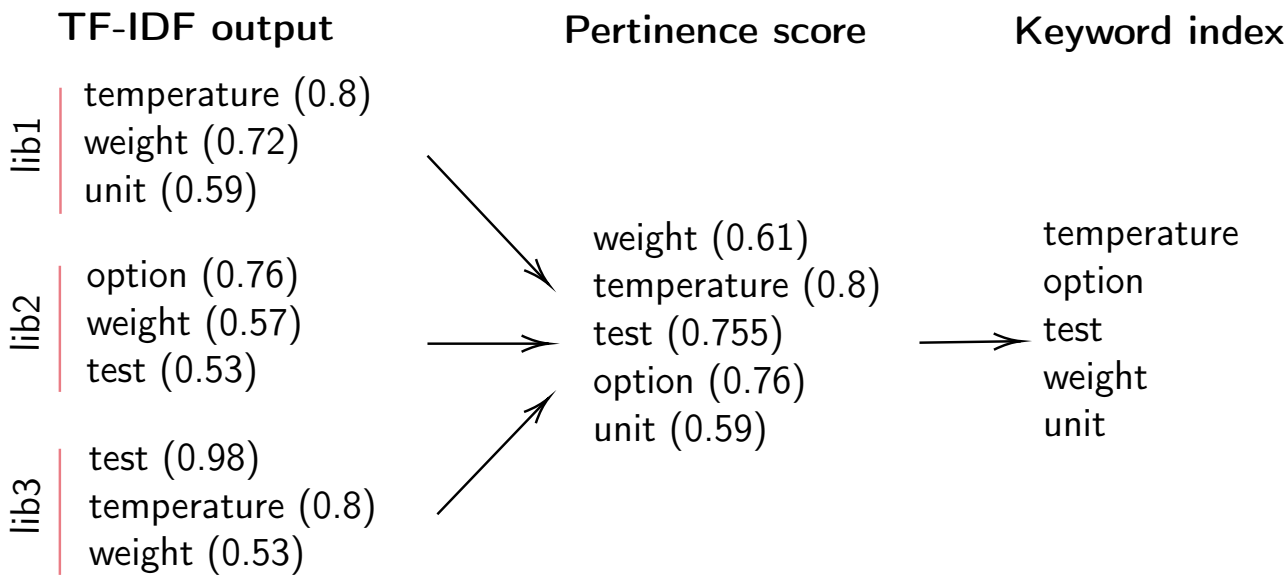
Experimentation with Aggregators





Other Potential Aggregators: Top-Down

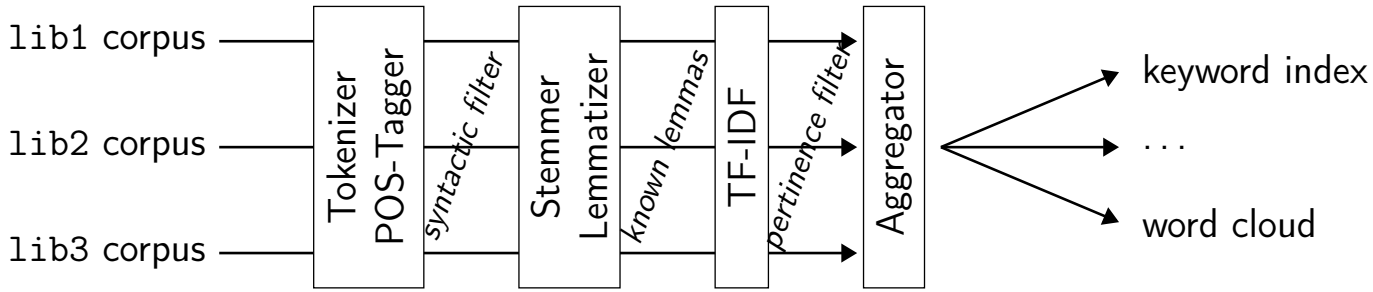
- Rank output of TF-IDF with a pertinence score (*e.g.* mean of TF-IDF values), and keep just enough keywords to reach full library coverage.



Other Potential Aggregators: Bottom-Up

- Start from keywords with the fewest associated libraries, and take until full library coverage is achieved.

Conclusion



- 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!