TikaEvalMetrics

tika-eval metrics

To be filled in.

Profiling Metrics

Common Words/Out of Vocabulary (OOV)

We gathered the top 30k most common words for ~120 languages from wikipedia or the Leipzig corpus. These word lists are available in the common_tokens directory.

When text is extracted for a given document, we run automatic language detection (thank you OpenNLP!) on the string and then count the number of common words for that detected language in the extracted text. We then the number of "common tokens" by the number of alphabetic words extracted. This gives us a percentage of common words or the inverse (1 - (commonTokens/alphabeticTokens), the Out of Vocabulary (OOV) statistic. This is some indication of how "languagey" the extracted text is.

If the assumption is that your documents generally contain natural language (e.g., not just parts lists or numbers), the OOV% statistic can identify outliers for manual review. For example, when we were testing the Tika 1.15 release candidate against the 1.14 version, we found that the text extracted by 1.14 had a 100% out of vocabulary (language id was Chinese, and the top 10 tokens were: : 18 | : 14 | : 14 | m: 11 | : 11 | : 11 | : 11 | : 10 | : 9 | : 9), however, FOR THE SAME FILE, the text extracted by the 1.15 release candidate had an OOV of 54% (language id was German and the top 10 tokens were: die: 11 | und: 8 | von: 8 | deutschen: 7 | deutsche: 6 | 1: 5 | das: 5 | der: 5 | finanzministerium: 5 | oder: 5). In short, text that has a high OOV *might* indicate a failed parse. Or, when comparing two different extraction methods, it is likely that the text with the lower OOV is better (assuming the extractors aren't hallucinating common words).

Tilman Hausherr originally recommended this metric as a comparison metric when comparing the output from different versions of PDFBox. For our initial collaboration with PDFBox, we found a list of common English words and removed those that had fewer than four characters. The intuition is that if tool A extracts 500, but tool B extracts 1,000, there is *some* indication that tool B may have done a better job.

Implementation Details

For now, we've set up an Analyzer chain in Lucene that:

- Filters out tokens that don't contain an alphabetic or ideographic character.
- Maps urls to "url" and emails to "email" (We don't want to penalize documents with urls and emails).
- Requires that a token be at least 4 characters long *unless* it is comprised entirely of CJK characters.

But wait, what's a word for non-whitespace (e.g. Chinese/Japanese) languages? We've followed common practice for non-whitespace languages of tokenizing bigrams...this is linguistically abhorrent, but it is mildly useful if inaccurate for our purposes.

Benefits: Easy to implement.

Risks:

- If an OCR engine relies solely on dictionary lookup and does not allow for out-of-vocabulary terms, the generated text will contain only known words, and the "common words" score will be incorrectly high. Yes, the text contains known words, but they might **not** reflect the correct text.
- If a document contains part numbers or other non-natural language tokens, then this metric will not accurately reflect success.
- Multi-lingual documents can cause challenges for interpretation. If the language id component "detects" English, even though the majority of the document is in Chinese, this metric will be misleading.

Comparison Metrics