NOUNS ARE BETTER THAN N-GRAMS

Asoka Diggs - Data Scientist

June 2018

IT@INTEL

Legal Notices

This presentation is for informational purposes only. INTEL MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS SUMMARY.

For more complete information about performance and benchmark results, visit www.intel.com/benchmarks

About Me

- Lots of database design and architecture experience
- Text analytics class during my Master's degree in Predictive Analytics
- Preparing text for analysis has been a theme of just about every project I've been involved in for the last 4+ years

Agenda

- Overview of using part-of-speech tagging to extract tokens from text
- Python notebooks showing some different extraction approaches
  - Comparison of standard pipeline tokens to noun phrase extraction
- Pseudo code example of scaling the method to a Spark cluster
- Q&A

Begin With the End in Mind

- A part-of-speech based token extraction pipeline for text:
  - Can be straightforward to implement
  - Generates tokens that are more human readable
  - Can find phrases naturally (replacement for n-grams generation)
  - Removes many stopwords “for free”
- Lemmatizing or singularizing terms can collapse terms that are different because of plural / singular differences

Typical pipeline

- Tokenize the text
- Convert all tokens to lower case
- Apply a stemmer to reduce tokens to common roots
  - E.g. boil, boils, boiler, boiled, boiling → boil
- Remove stop words
  - Common English words (the, of, to, in, and, or, which)
- Might also include n-gram generation

Challenge

- Standard pipeline for preparing text for analysis yields tokens that can be of low value
- These tokens can be difficult to explain to business users – what do the tokens mean?
- Uses many rules for handling text, and text (nearly) always has exceptions to each rule.

Solution

- Replace the standard tokenize-stem-stop-n_gram pipeline with a part-of-speech based pipeline
- Extract words and phrases with the desired parts-of-speech
- Use singularization as a different approach to stemming to bring singular and plural forms of words together

Result is not perfect, but better quality than the tokenization approach.

Benefit

The resulting tokens naturally:

- Include phrases (n-gram replacement)
- Remove the standard English stop words. They have a part-of-speech that isn't typically used in analysis
- Supports readability and don't need stemming to collapse similar words to a common root

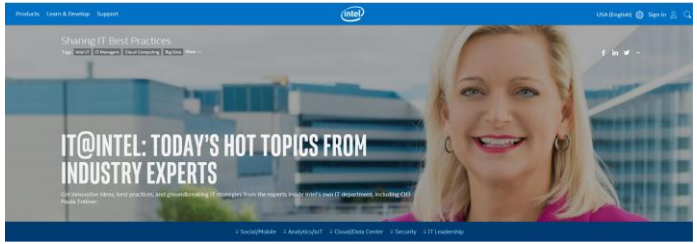
DEMO

Resources and Links

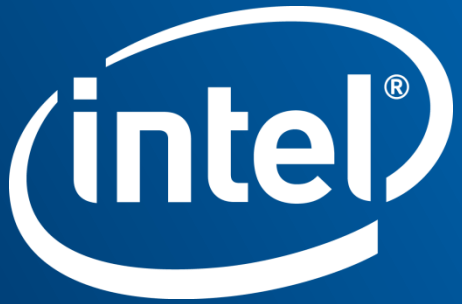
- Pattern
  - For Python* 3 compatibility, I had to install a development branch of the library. See (link) for details
    - Something like:
      git clone –b development https://github.com/clips/pattern
      cd pattern
      sudo python setup.py install
- NLTK
- Penn-Treebank (on Wikipedia) and Tags
- Intel® Distribution for Python* (link)
- Others: RDRPOSTagger, spaCy, rakutenma, TextBlob

Q&A

IT@INTEL: Sharing Intel IT Best Practices With the World



Learn more about Intel IT's initiatives at: **www.intel.com/IT**


intel®