Gargi Deb
Text Information Systems
November 7, 2021

CS 410: Tech Review Paper

The NLTK python package is a robust and powerfully built package that can be utilized when working with text data. There are many handy tools that can be utilized by this package and can be applied to solve a variety of problems ranging from part of speech tagging to sentiment analysis. This review will give an overview of some of these effective tools and discuss its applications.

One of the most popular tools utilized in NLTK is for sentiment analysis. Doing sentiment analysis on text data is a very useful way to see whether text information, like reviews, are positive, negative, or neutral. This can help when deciding whether to purchase a product or a service. However, if the sentiment analysis is not accurate enough, that may cause the user to have an incorrect judgement of the data. So, if one wants to simply get the sentiment of their text data, they can utilize SentimentAnalyzer from the NLTK package to get the sentiment of the text (*NLTK Documentation: Sample Usage for Sentiment*). They first can split their dataset into train and test and utilize the NaiveBayesClassifier to classify the text as positive, or neutral. Though this will give a general baseline for classifying text by their sentiment, it may be more beneficial to create a more tailored model to the dataset and extract more specific features depending on the data. For example, using the same model on a movie review dataset may perform differently than a restaurant review dataset.

Another useful tool utilized in NLTK is for parsing text. If a user has a use case where they want to extract a feature where they would want to count how many nouns are in sentence, they can use NLTK's parse package. This package is charge of "producing tree structures that represent the internal organization of a text." (*NLTK*

Documentation: Nltk.parse Package). The main downfall for using the parser, is that it cannot distinguish between ambiguity in text. The parser module however does contains different kinds of specialized parsing depending on where the user would like to begin the parsing from in the text such as bottom-up, stepping chart, or probabilistic. All these different parsers could create different parse trees for the same text.

Some very important and necessary steps to take when analyzing any text data requires data cleaning steps. These steps can be done by leveraging some of the tools in NLTK. One of the important modules in the NLTK package that is widely utilized for data cleaning is tokenization. This brings all words to their root form, and ensures that all variations of a word is brought to their same root. There are different tokenizers within the tokenize package as well such as the treebank tokenizer and the regexp tokenizer (*NLTK Documentation: Sample Usage for Tokenize*). One very interesting use case is if someone wants to analyze tweets and perform a task like sentiment analysis for tweets, part of the cleaning step they can utilize a tokenizer available in NLTK that is specifically geared towards tokenizing tweets, called the tweet tokenizer. Another very similar and useful module that can be utilized in the cleaning step is the stem module. Similar to the tokenizer module, this module brings all words to their stem form.

Overall, the NLTK package has very practical and important use cases that can be applied to a variety of applications. The important thing to remember is that for many applications, the NLTK package provides a good baseline for performance, however after further analyzing and reflecting, sometimes it may be needed derive a specific module for its own use case. Though this is a great starting point, there is still a lot of work to do in this area. NLTK is mainly tailored to the English language and when it comes to doing similar analysis on different languages, there is still much work left to be done in that area. All in all, this is still a powerful package for the English language and if utilized for the apt use cases, it can have favorable results.

Works Cited

Garrette, Dan, et al. “NLTK Documentation: Nltk.parse Package.” *NLTK*, 19 Oct. 2021, https://www.nltk.org/api/nltk.parse.html.

Garrette, Dan, et al. “NLTK Documentation: Sample Usage for Sentiment.” *NLTK*, 19 Oct. 2021, https://www.nltk.org/howto/sentiment.html.

Garrette, Dan, et al. “NLTK Documentation: Sample Usage for Tokenize.” *NLTK*, 19 Oct. 2021, https://www.nltk.org/howto/tokenize.html.