Nicholas Truong

Overview of Natural Language Tool Kit (NLTK)

One of the main frameworks and toolkit for Natural Language Processing (NLP) is Natural Language Tool Kit (NLTK). NLTK is one of the standards because of its rich access to many cutting-edge NLP techniques through simple function calls in Python. However, as I looked more into the field I realized that there are still so many other frameworks and tools that can be used such as Pytorch. Pytorch is a more general Deep Learning tool, but it has a section of it devoted to text analysis called TorchText. I will be analyzing how TorchText and NLTK are different and which to prefer.

There are a couple of areas where there is a big difference between TorchText and NLTK. One example of this is in n-gram language modeling. In TorchText, you can only load the dataset and the train/test split that you wish to train on via **torchtext.datasets**. These datasets include: WikiText-2, WikiText-103, and PennTreebank. While this is extremely useful, this does not directly help build the n-gram language model. TorchText does provide a **ngrams_iterator()** method that returns an iterator that yields the next token and its ngram; however, this is not as rich a function as that provided by NLTK. NLTK provides multiple functions to help build an n-gram language model such as **nltk.lm.MLE**, which provides the MLE ngram model score, and **nltk.lm.laplace**, which adds the Laplace smoothing to the model. In general it seems that NLTK provides more in-depth tools to generate n-gram language models.

Another major difference between NLTK and TorchText is the availability of datasets/corpuses to download and use. TorchText lists only 20 datasets to download from whereas NLTK has 108 datasets that can be downloaded. One of the biggest things that TorchText is missing from its corpuses is a large variety of language support. TorchText only has one Chinese dataset while NLTK has support for many more languages. This is a significant difference between the two tools and outlines a major difference between the rich-ness of the two.

There are some similarities between the two tools as they both do provide some of the same functionalities. For example, both tools allow us to easily calculate the BLEU score of two corpuses so that we can determine how accurate the machine translation is. In NLTK, it is a function called **ntlk.translate.bleu** and in TorchText it is a function called **torchtext.data.metrics.bleu_score.** There are also a variety of other utilities that are available in both tools that help with NLP.

When it comes to choosing between TorchText and NLTK, it is clear that NLTK provides more of the functionalities that we would need for any NLP project. TorchText does provide its own sets of functions, but it is a lot more limited when compared to NLTK's offerings. Thus, it is clear that NLTK is the better option for most tasks; however, there is one exception to this. If the specific NLP task requires more cutting-edge deep learning techniques, then we should prefer to use PyTorch (the overall library that contains TorchText) because PyTorch offers more Deep-Learning methods than NLTK does.

References

https://pytorch.org/text/stable/index.html

https://www.nltk.org/