Chris Zhu
cjzhu2
CS 410

Introduction to NLP in TensorFlow

As one of the most cutting-edge libraries available for machine learning applications, it is natural that TensorFlow has many available resources for natural language processing purposes. As such, there are a number of notable techniques and tools to become familiar with the basics of using TensorFlow for NLP. Thus, the following is a brief guide and overview on how to get started.

To get started, it is important to know exactly what TensorFlow is. Created by Google, this free and open-source library for machine learning provides tools for use with all of the most popular programming languages, and a higher-level API called Keras for easy access to said tools. Here, we will use these tools to create a basic classifier on text, one of the most elementary tasks in the space of NLP. While image classifiers can easily use the number values associated with pixels, this technique does not translate very easily to language and words. One approach that does not work particularly well is using the ASCII values assigned to individual letters, as this causes words with the same letters to be classified identically. Instead, we can assign each word in our vocabulary a number to be used as a unique identifier. TensorFlow and Keras provide an easy to use framework to do this called *Tokenizer*.

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
sentences = [
'I love my dog',
'I love my cat',
'I like my garden',
'My cat plays in my garden'
]
tokenizer = Tokenizer(num_words=100)
tokenizer.fit_on_texts(sentences)
word_index = tokenizer.word_index
print(word_index)
padded_sequences = pad_sequences(sequences, max_length=6)
print(padded_sequences)
```

(Source 1)

The above code successfully uses *Tokenizer* to fit on the example data in *sentences*. It then takes the *word_index* variable of *tokenizer*, which is a dictionary of *word : identifier* key value pairs. Every time a sentence is added to the data, all unseen words are assigned a new identifier number within this word index. Finally, in order to normalize the lengths of the sentences to make it easier to train the neural network on them, the number sequences representing the sentences are *padded* with 0's at the beginning if they do not meet the minimum length threshold. With all of this applied, the above code returns the following for *padded_sequences:*

```
[[0,0,2,3,4,5], [0,0,2,3,4,6], [0,0,2,7,4,8], [4,6,9,10,4,8]]
```
(Source 1)

This result may not have immediately obvious applications, but these kinds of row vectors can be applied in conjunction with many other techniques in order to automate and learn about the meanings of sentences. For example, if we compare the differences in values between sentences 1 and 2, versus the same number for sentences 3 and 4, we can clearly see that the first comparison has much closer values, and is thus likely more similar in meaning. This type of logic can extend much farther when using more mathematical approaches like linear algebra or statistics to form such conclusions. Furthermore, seeing as this is a very elementary approach, extending this technique beyond simple word tagging, such as using tags that can denote similarity between words of similar meaning can make the usefulness of doing this much more extensive. It is clear that this rather simple building block can be extended in many different ways.

In conclusion, the above example is a fairly trivial application of TensorFlow and Keras that still highlights the power and robustness of the library. With very few lines of code and basic understanding of the theory behind the techniques being used, one can very quickly and easily implement machine learning for NLP on fairly large datasets with little to no hassle involved. It is here that the true value of such a library becomes clear, as the programmer themself no longer needs to get their hands dirty with the implementation of the models involved. Simply put, one can use the work done by Google and others to get their models up and running with very little effort.

Works Cited:

https://medium.com/@aqsakausar30/nlp-in-tensorflow-all-you-need-for-a-kickstart-3293d7d2630e

https://towardsdatascience.com/natural-language-processing-with-tensorflow-e0a701ef5cef

https://en.wikipedia.org/wiki/TensorFlow#Features