NLP using TensorFlow

About TensorFlow, usage, advantages and comparison with other NLP tools

TensorFlow is an end-to-end open source platform for machine learning developed by Google. It has comprehensive and flexible tools, libraries, and community resources that lets researchers and developers push the state-of-the-art in Machine Learning. It provides stable Python and C++ APIs.
In this review, we discuss how TensorFlow has made basic text preprocessing tasks easier in NLP, TensorFlow's key features, its drawbacks and comparison with other NLP tools in the industry today.

To give a background, the main objective of NLP is to read, decipher, understand, and make sense of the natural human language in a way that is valuable.
TensorFlow is currently the leading open source software for deep learning, computer vision, natural language processing (NLP), speech recognition, and general predictive analytics.

TensorFlow applications in NLP

TensorFlow provides libraries for most of the traditional NLP text preprocessing tasks making the program robust and fast.

- Tokenization:
  We vectorize a text corpus, by turning each text into either a sequence of integers or into a vector. The coefficient for each token could be binary, based on word count and on tf-idf.
  ```
  from tensorflow.keras.preprocessing.text import Tokenizer
  tokenizer = Tokenizer(num_words,oov_token="<OOV>")
  tokenizer.fit_on_texts(sentences)
  ```

- Sequencing:
  We represent our sentences by a sequence of numbers in the correct order. This function transforms a string of text into a list of words while ignoring filters like punctuations.
  ```
  sequences = tokenizer.texts_to_sequences(sentences)
  ```

- Padding the sequences:
  Pads sequences to the same length.
  from tensorflow.keras.preprocessing.sequence.pad_sequences
  pad_sequences(sequences)

- Word Embedding:
  Words and associated words are clustered as vectors in a multi-dimensional space. Words are present in a sentence and often words with similar meanings are close to each other.
  tensorflow.keras.layers.Embedding(vocab_size, embedding_dim, input_length = max_length)

TensorFlow Key Features

Tensor flow is popular and ahead of race amongst other NLP tools.

- Keras is a high-level API for TensorFlow, and it's extended so that you can use all the advanced features of TensorFlow directly from tf.keras.
- TensorFlow offers high performance models for large datasets that require fast execution.
- In TensorFlow 2.0, eager execution is default. *Eager execution is a define-by-run interface where operations are executed immediately as they are called from Python.* You can take advantage of graphs even in eager context, which makes your debugging and prototyping, while TensorFlow takes care of performance
- Tensorflow has better computational graph visualizations.
- Tensorflow lets you execute subparts of a graph.You can introduce and retrieve discrete data and therefore offers a great debugging method.
- TensorFLow is highly parallel and designed to use various backends software (GPU, ASIC) etc.

New in TensorFlow

The TensorFlow team announced T5 Text-to-Text Transfer Transformer, a framework that applies transfer learning to natural language processing. Meena is a neural conversational model that learns to respond sensibly to a given conversational context.

TensorFlow Drawbacks

Given the numerous advantages of using TensorFlow, TensorFlow lacks on a few fronts.

- TensorFlow is incomprehensive and less user friendly than its counterpart PyTorch.
- TensorFlow uses static graphs for computation, so you need to define the computation graph statically before a model is run.
- Debugging isn’t that simple with TensorFlow. You need to use a special tool called tfdbg, which evaluates TensorFlow expressions at runtime.
- TensorFlow does not offer comfortable support for Windows.You need to install it within a conda environment or using the python package library, pip.

Comparison with other tools

- In keras, there is usually very less frequent need to debug simple networks. But in the case of Tensorflow, it is quite difficult to perform debugging. Pytorch on the other hand has better debugging capabilities as compared to the other two.
- Keras is usually used for small datasets as it is comparatively slower. On the other hand, TensorFlow and PyTorch are used for high performance models and large datasets that require fast execution.
- PyTorch framework is more tightly integrated with Python language and way more friendly and simpler than TensorFlow.
- In PyTorch things are way more dynamic.You can define, change and execute nodes as you go, no interfaces or placeholders required.
- TensorFlow uses static graphs for computation while PyTorch uses dynamic computation graphs.

Overall TensorFlow has helped us improve NLP performance by offering libraries to perform traditional and tedious NLP text preprocessing tasks. PyTorch and Keras give healthy competition to TensorFlow.