Tech Review - TensorFlow for NLP

By Joshua M. Smith

University of Illinois Urbana-Champaign
CS410, Professor ChengXiang Zhai, Fall 2021

Background

Natural Language Processing (NLP) is an area of Artifical Intelligence (AI) which focuses on linguistics principles and how humans and computers relate to each other. Specifically, NLP focuses on how computers can be made useful for human language tasks, such as understanding documents and word terms, organizing them, and then providing meaning in data at scale, or allowing simple search and browsing. NLP is a broad topic, and there have been many tools developed to assist engineers and scientists with practical implementations.

Introduction to TensorFlow

Tensorflow is a free and open-source library for AI and machine learning applications developed by Google in 2015. As a search company, Google required the use of large scale deep neural networks across many of their products, including Google Search, GMail, Youtube, and more. Tensorflow is used for training and inference of deep neural networks on Python primarily, with implementations in many other languages.

Tensorflow is named after the library's input component, the tensor. A tensor is an array, generally with three or more dimensions. Tensors are subject to a variety of linear transformations (e.g. multiplication, rotation) between them, which can be mapped onto a graph for easier management."Tensor flow" refers to the flow of data and transformations through the tensor graph or the workflow in Python. Tensorflow models use layers to distinguish between states, and each state has one or more input tensors, and one or more output tensors. Layers can be stacked to build a sequential model for a specific outcome, and there are pre-defined layers as well as mechanisms to build custom layers.

Many large companies today use Tensorflow for a variety of applications, including Google, SAP, Nvidia, Intel, Paypal, and Twitter. Tensorflow is also used by many Bay Area startups including Grammarly, Uber, Spotify, Snapchat, and Airbnb. These applications vary from image recognition (Airbnb) to fraud detection (PayPay) to NLP for grammar correction (Grammarly).

Using TensorFlow

Tensorflow provides many modules for NLP, which are used primarily through its programming interface for Python, called Keras. Keras supports both CPU and GPU machine learning and

allows abstraction of the details of Tensorflow, so that quick iterations and improvements can be performed without low-level changes that require deep TensorFlow knowledge.

Features of TensorFlow include:

Eager execution: When running quick iterations, developers need a quick response to inspect code and read results with print statements or debuggers. Eager execution, enabled by default, instructs Tensorflow to return values immediately to Python. Eager execution can be combined with NumPy for easy debugging or for use with other libraries.

Text Vectorization and Tokenization: During data preprocessing, NLP first requires separation of text or documents into smaller units, referred to as tokens. These are often individual words, but can also be characters or sub-words (n-character terms). Keras provides Tokenizer APIs which can process text documents and create an index of unique words to be used for fast search. The Tokenizer can also create numerical sequences based on input sentences or documents, which when combined with the index can reconstruct these sentences. These operations can be combined using TensorFlow's TextVectorization APIs, which combine efficient text tokenization and vocabulary indexing.

Normalization: Tensorflow's Normalization Keras layer performs feature-wise normalization of input features in a data set. Normalization is used to bring terms to a usable state after their tokenization. For example, the terms "move", "moves", "moving", and "moved" all refer to the same root (move), called a lemma. We can normalize our vocabulary by stemming or lemmatization of our index. Terms may also have to be normalized against other factors such as an index of stop words which will not assist in text analysis, or against the size of the input document or corpus.

Similarity Metrics with TensorFlow Text: A module called TensorFlow Text provides text metrics classes and operations which can be used to automatically evaluate text generation models, such as Rouge-L. Rouge-L can be used with text generation models to return f-measures, as well as p-measures and r-measures for precision and recall, respectively.

The Sequential Model: In TensorFlow, a sequential model is a straightforward method of modeling where layers are added to the model one by one linearly, from the input to the output. This is defined by using the Sequential API from Keras to define a model, then using a model.add() command to addlayers. It allows a level of flexibility for those that wish to define custom models, without

being overwhelming. One example where the sequential model can be applied with text is a recurrent neural network (RNN).

The Functional Model: For more flexibility, TensorFlow's Functional Model API provides customization by allowing the definition of explicit flow between layers. In the functional model, each connection is specified, from the output of one layer into the input of the next. This allows for models where there might be shared layers, non-linear topology, multiple input paths or vectors, or have multiple outputs, perhaps of differing types.

Training, evaluation, inference, and saving models work the same way for the Sequential and Functional APIs. Additionally, entire models can be used as layers, for creating complex and modular workflows in TensorFlow models. Models can be plotted to be viewed visually as a map, showing each layer or sub-model and its input and output.

There are some alternatives to using TensorFlow, one popular example being PyTorch. PyTorch was developed by Facebook, a re-written version of the Lua Torch framework for Python. PyTorch is newer and less commonly used, but its adoption is growing. PyTorch also has many differences from TensorFlow, including the way that graphs are defined. In PyTorch, graphs are dynamic, whereas they are statically defined in TensorFlow. TensorFlow also provides more visualization options for graphs. Which tool to use is going to depend on the type of model as well as the subfield of machine learning.

Sources:

https://www.tensorflow.org/guide
https://www.tensorflow.org/guide/keras/sequential_model
https://www.tensorflow.org/guide/keras/functional
https://www.tensorflow.org/about/case-studies
https://www.grammarly.com/blog/how-grammarly-uses-ai/
https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf
https://www.youtube.com/watch?v=fNxaJsNG3-s
https://pytorch.org/features/
https://pytorch.org/tutorials/beginner/nlp/pytorch_tutorial.html