Santosh Kore-CS410 Fall 2020

Technology Review-TensorFlow and Keras API for NLP

TensorFlow is open source end-end to platform for implementing Machine Learning Algorithms, it is used for training and evaluating neural network. It provides a comprehensive ecosystem of tools for developers, enterprises and researchers who want push the state of the art of the machine learning and build scalable machine learning powered applications. It takes input as multi-dimensional array, Multi-dimensional arrays are also known as Tensors, hence the name TensorFlow.

The core TensorFlow library is implemented in C++ for portability and performance: it runs on several operating systems including Linux, Mac OS X, Windows, Android, and iOS; the x86 and various ARM-based CPU architectures; and NVIDIA's Kepler, Maxwell, and Pascal GPU microarchitectures. The implementation is open-source, and there are several external contributions that enable TensorFlow to run on other architectures.

TensorFlow provides multiple official models for Natural language processing such BERT (Bidirectional Encoder Representations from Transformers), ALBERT (A Lite BERT), NHNet (News Headline generation model), Transformer and XLNet. Official models are well optimized for speed/performance, but still they are easy to read and understand. These models are maintained, tested and kept to up to date with latest version of TensorFlow. Also, there are official computer vison models for Image Classification (MNIST, ResNet, EfficientNet) and Object Detection and Segmentation (RetinaNet, Mask R-CNN, ShapeMask and SpineNet).

TensorFlow runs and graphs can be inspected and understood using the **TensorBoards**, suite of web applications provided by TensorFlow. Some of the common TensorBoards are Scalar Dashboard used to visualize the statistics that vary over the time; Histogram Dashboard displays the statistical distribution varied over the time; Distribution Dashboard, Image Dashboard, Audio Dashboard, Graph Explorer, Embedding Projector, Text Dashboard.

TensorFlow Lite is TensorFlow's lightweight solution for mobile and embed devices. TensorFlow Lite runs the machine learned model on mobile device with low latency so one can do classification, regression etc. without making roundtrip to the server. It is supported on IOS devices using the C++ API and on android devices through Java Wrapper APIs. To run the model on mobile, model will be built separately and exported to mobile devices.
TensorFlow with Keras has numerous use cases like Text Classification, Sentiment Analysis, Image Recognition, Speech Recognition, Object Tagging.

Keras is a high-level deep learning API written in Python, running on top of the machine learning platform TensorFlow 2.0. It is developed as an API without backend and it supports

multiple backend APIs. Even though it's a high-level API most of the time it is sufficient for many use cases. Main data structures in Keras are layers and models. Layers are the basic building blocks of neural networks in Keras. A layer consists of a tensor-in tensor-out computation function (the layer's call method) and some state, held in TensorFlow variables (the layer's *weights*). While Keras offers a wide range of built-in layers, they don't cover every possible use case. Creating custom layers is very common, and very easy. Core layers in Keras are Input object, Dense layer, Activation layer, Embedding layer, Masking layer, Lambda layer.

There are numerous companies and research/educational institutions worldwide are leveraging the TensorFlow to build state-of-the-art machine learning solution to solve the problems that were not easy or impossible to solve previously. To mention few, Airbnb improved the guest experience by using TensorFlow to classify the images and detect objects; Engineering students in India along with their mentors have used TensorFlow to build an app that measure the air quality; General Electrics trained neural network using TensorFlow to identify the anatomy on MRI's of the brain; PayPal Inc uses TensorFlow to detect Fraud; Coca-Cola has leveraged the TensorFlow platform to enable mobile proof-of-purchase.

Basic Text Analysis/Text Classification using TensorFlow-Keras API involves below steps at a high-level,

Understand the problem statement: Basically, we would like to understand the end goal of the model here for example classify Twits as Sarcastic/Non-Sarcastic, classify reviews as negative/positive etc.

Understand the data-set: Next step is to understand the data set available for us to train and test the data set. This involves length and dimension of the data, number of records available for training etc. Many of the commonly used data sets are already exposed through Keras API, for example IMDB reviews can be accessed using *from tensorflow. python.keras.dataset import imdb*. We can leverage Keras API to expose any new datasets.

Pre-process the examples and labels: Next step is to preprocess the data in order to feed the data into model/neural network. This step involves tokenizing the data, padding the data using pad_sequences (from *tensorflow.python.keras.preprocessing.sequence* import pad_sequences) and converting the text representation into numerical representation. This data preparation step can be performed using the Tokenizer API provided with Keras

Understand word embedding: A word embedding involves representing words and documents using a dense vector representation. In an embedding, position of the word in embedding is learned by words surrounding it in vector space rather than individual words. Keras offers extensive API for creating embedding layer that can be used for neural network model on text data.

Create a neural network model: Next step we create the model that will be trained using the training data and evaluated using the test data. Keras model API provides 3 ways to create models,

The Sequential model, consists of a simple list of layers. This layer limited to single-input, single-output stacks of layers as the name suggests.

The Functional API, which is an easy-to-use, fully-featured API that supports arbitrary model architectures. This is the model that should fit to most use cases in the real-world scenario.

Model subclassing, this is used when model requirement does not fit into the existing API and need to build custom implementation from scratch. We can obtain the summary of the model created using the Model. Summary method that prints a string summary of the network.
Train the model to fit the data set: Keras network model training involves compiling and fitting the model to training data. *Model. Compile* method takes multiple input to compile the model. Main inputs to compile method are optimizers, losses, and metrics. Most of the cases you won't have to create optimizers, losses, and metrics, Keras API provides multiple optimizers such as SGD, RMSprop, Adam etc.; Losses such as MeanSquaredError, KLDivergence, CosineSimilarity etc. and Metrics such as AUC, Precision, Recall, etc.
Evaluate the model: Evaluate method provided by Keras API does the computation in batches and returns the loss value & metrics values for the model. Evaluate method takes multiple parameters such as batch size (default to 32 if not specified), set multiprocessing parameter to use process-based threading and also API provides the flexibility to use call back methods.

In this technical review on TensorFlow and Keras API I have only touched surface of the things that TensorFlow and Keras API can do. With Tenorflow2.0, TensorFlow Lite and Keras API researchers and engineers can build state of the art Machine Learning models that can be used across multiple problems worldwide and across different areas such as healthcare, manufacturing, aviation etc.

References:
https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/45166.pdf
https://www.tensorflow.org/
https://keras.io/