High Level TensorFlow for Sentiment Analysis

Introduction

TensorFlow is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. It is a symbolic math library based on dataflow and differentiable programming. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache License 2.0 in 2015.

Sentiment Analysis is an application of Natural Language Processing (NLP) in which we try to define the sentiment used by author in the given text. The output is a classification task, and it could be anything like positive, negative, neutral, happy, etc.

Sentiment Analysis & TensorFlow

Sentiment analysis (also known as opinion mining or emotion AI) is the use of natural language processing, text analysis, computational linguistics, and biometrics to systematically identify, extract, quantify, and study affective states and subjective information.

Sentiment analysis is widely applied to voice of the customer materials such as reviews and survey responses, online and social media, and healthcare materials for applications that range from marketing to customer service to clinical medicine.

Sentiment analysis is one of the very common natural language processing tasks. Businesses use sentiment analysis to understand social media comments, product reviews, and other text data efficiently. TensorFlow and Keras are amazing tools for that.

Keras to be precise, is a high level TensorFlow (TF) API which can be utilized to create Deep Learning model in easy way by utilizing the pre-made functions in the library.

Sentiment analysis is often driven by an algorithm, scoring the words used along with voice inflections that can indicate a person’s underlying feelings about the topic of a discussion. Sentiment analysis allows for a more objective interpretation of factors that are otherwise difficult to measure or typically measured subjectively.

There can be various use cases for sentiment analysis, some of them are given below-

1. Segmenting user based on opinions
2. Planning a product or service improvement
3. Tracking sentiment continuously
4. Planning a process improvement

Let us now dive into the practical implementation of determining sentiment analysis using TensorFlow Keras APIs.

Data Preparation

Data preparation is the first step in any ML model development process. To encode the labels in the data we utilize the Pandas library which can easily convert “Positive” & “Negative” labels to numerical categories.

```
df['sentiments'] = df.sentiments.apply(lambda x: 1 if x==’Positive’ else 0)
```

The next step is tokenization, where we convert the sentences to numerical representations. Here we select the vocabulary which has a select minimum number of words and rest we declare as Unknown words, (OOV).

Along with the unknown words, we select the maximum length of sentence for training the text data. If any sentence is lower than the given length, we need to add padding (blank words until the max length). In case of length greater than the set value, we cut the sentence till the length.

```
tokenizer = Tokenizer(num_words=vocab_size, oov_token=oov_tok)


tokenizer.fit_on_texts(training_sentences)


word_index = tokenizer.word_index
```

We also created a word index dictionary to map the words after inference.

```
sequences = tokenizer.texts_to_sequences(training_sentences)

padded = pad_sequences(sequences, maxlen=max_length, truncating=trunc_type)

testing_sentences = tokenizer.texts_to_sequences(testing_sentences)

testing_padded = pad_sequences(testing_sentences, maxlen=max_length)
```

Model Development

Model development is the most crucial step, where we make the Deep Learning model consisting of Artificial Neural Networks (ANNs) and the structure for Natural Language Processing (NLP).

There could be several layers consisting of various neurons created using TensorFlow (TF) library.

```
model = tf.keras.Sequential([

tf.keras.layers.Embedding(vocab_size, embedding_dim,

                                              input_length=max_length),

tf.keras.layers.GlobalAveragePooling1D(),

tf.keras.layers.Dense(6, activation='relu'),

tf.keras.layers.Dense(1, activation='sigmoid')])
```

After model definition, we create the compile method to give parameters like optimizer and accuracy metrics.

```
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
```

Finally, we train the model using the fit method provided by Keras API.

```
training_labels_final = np.array(training_labels)

testing_labels_final = np.array(testing_labels)
```

```
num_epochs = 20

history = model.fit(padded, training_labels_final,

epochs=num_epochs,

validation_data=(testing_padded, testing_labels_final))
```

Conclusion

To conclude, TensorFlow provides excellent high-level model development and data preparation APIs through Keras.
For practical implementation purpose, this serves to be the best choice for beginners for implementing Deep Learning models.

References

1. https://en.wikipedia.org/wiki/TensorFlow
2. https://www.tensorflow.org/api_docs/python/tf
3. https://www.tensorflow.org/hub/tutorials/tf2_text_classification
4. https://towardsdatascience.com/a-complete-step-by-step-tutorial-on-sentiment-analysis-in-keras-and-tensorflow-ea420cc8913f
5. https://callminer.com/blog/sentiment-analysis-examples-best-practices