**Shiyu Wang**

**BUID: U43910814**

# Term Project Report

**Table of Contents**

**[Table of Contents.............................................................................................................................1](#_Toc1860463515)**

**[Project Description..........................................................................................................................2](#_Toc1860463515)**

**[Dataset...............................................................................................................................................2](#_Toc1860463515)**

**[What is Bert?....................................................................................................................................2](#_Toc1860463515)**

**Spark NLP........................................................................................................................................3**

Text Classification Steps............................................................................................................3

Related Components from Spark NLP.......................................................................................4

**Training Components Of This Project..........................................................................................5**

**Running on Google Cloud...............................................................................................................9**

Setting Up Cluster Environment................................................................................................9

Model Results............................................................................................................................9

**References.........................................................................................................................................9**

**Project Description**

This term project is to implement Bert in PySpark for sentiment analysising.

Many materials have proofed that Transformer owns an advantage in machine learning and deep learning tasks, thus it is necissary to learn and implement Bert/Transformer.

At the same time, NLP is a big branch in the field of artificial intelligence. The recent chatGPT has brought a huge breakthrough in the field. We seem to have taken a big step forward on the way to general artificial intelligence. Although this term project is just a simple start, thinking about what I will continue to learn in the future, it cannot be more exciting.

**Dataset**

At this term project, I use Twitter Sentiment Analysis Dataset. This is an entity-level sentiment analysis dataset of twitter. Given a message and an entity, the task is to judge the sentiment of the message about the entity. There are two classes in this dataset: Positive(label 4) and Negative(label 0).

The original dataset can be download from Kaggle. I took out some of them as a small dataset. For convenience, I put both the original dataset and small dataset in the Google cloud.

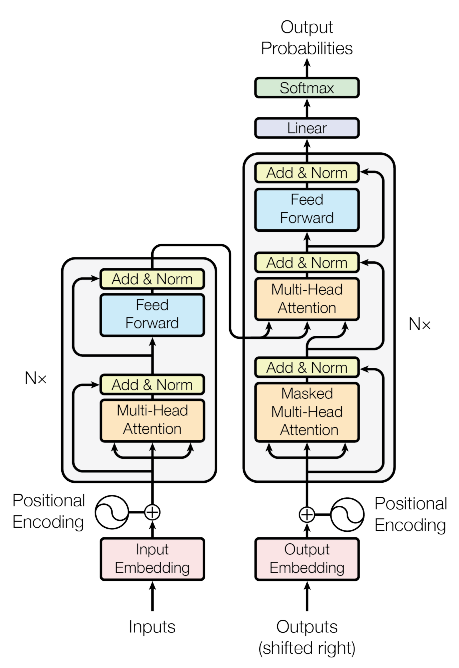
The links are in the below table. And the table also shows that the number of positive and negative data in each dataset is basically the same, which ensures the category balance of the training data.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Pos\_num | Neg\_num | Link |
| Large Dataset | 800,000 | 800,000 | <https://storage.cloud.google.com/777-term_project/largeDataset_1.6M.csv>  gs://777-term\_project/largeDataset\_1.6M.csv |
| Small Dataset | 8,037 | 7,905 | <https://storage.cloud.google.com/777-term_project/smallDataset.csv>  gs://777-term\_project/smallDataset.csv |

**What is Bert?**

Let’s start with Transformer Model. A transformer model is a neural network that learns context and thus meaning by tracking relationships in sequential data like the words in a sentence.

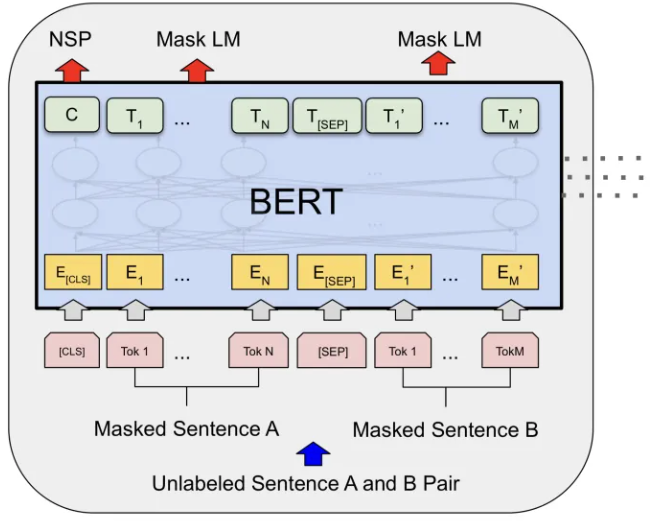
Transformer models apply an evolving set of mathematical techniques, called attention or self-attention, to detect subtle ways even distant data elements in a series influence and depend on each other.[6]



As we can see on the ahead picture, the left part of the model is named encoder, and the right part of the model is named decoder. The final output of the Encoder serves as the representation and encoding information of the input sentence. Decoder inputs the encoded information of the sentence and outputs the predicted content.

BERT stands for Bidirectional Encoder Representations from Transformers, is a pre-trained language representation model. Bert uses the masked language model (MLM), which means can use the words on the left and right of a word at the same time, thus it can generate a deep two-way language representation.

The training structure is as followed. In the figure, the blue part indicates that multiple encoder structures are stacked together. We input a pair of sentences A and B, and train the model to distinguish which words belong to A and which words belong to B. The final output is C in the figure. We do sentence-level tasks, and other layers can be connected after C.

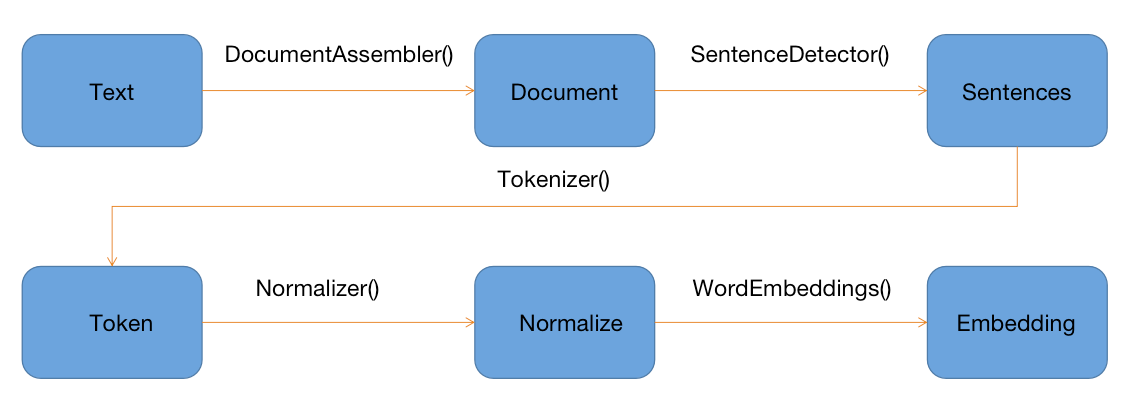


**Spark NLP**

***Text Classification Steps***

Basically, we can divide the text into two streams based on text embedding: word embedding based stream and sentence embedding based stream.

The process of Word Embedding is as follows:



The brief process of Sentence Embedding is as follows:



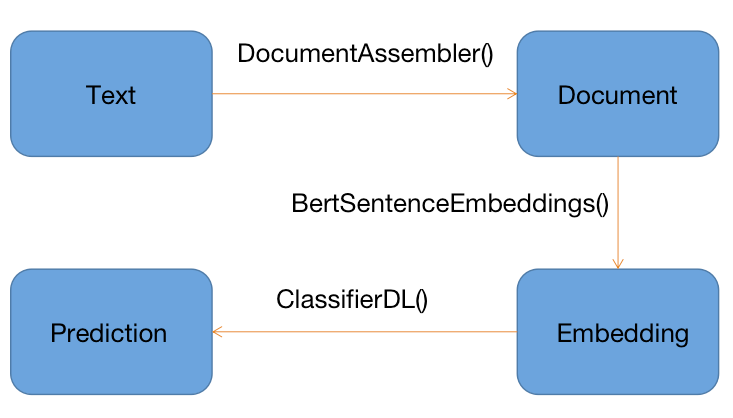
***Related Components from Spark NLP***

Some annotators related to this term project are selected below for illustration:

|  |  |
| --- | --- |
| **Annotators** | **Description** |
| [ClassifierDL](https://nlp.johnsnowlabs.com/docs/en/annotators" \l "classifierdl) | ClassifierDL for generic Multi-class Text Classification. |
| [DocumentAssembler](https://nlp.johnsnowlabs.com/docs/en/annotators" \l "documentassembler) | Prepares data into a format that is processable by Spark NLP. This is the entry point for every Spark NLP pipeline. |
| [DocumentNormalizer](https://nlp.johnsnowlabs.com/docs/en/annotators" \l "documentnormalizer) | Annotator which normalizes raw text from tagged text, e.g. scraped web pages or xml documents, from document type columns into Sentence. |
| [SentenceDetector](https://nlp.johnsnowlabs.com/docs/en/annotators" \l "sentencedetector) | Annotator that detects sentence boundaries using regular expressions. |
| [StopWordsCleaner](https://nlp.johnsnowlabs.com/docs/en/annotators" \l "stopwordscleaner) | This annotator takes a sequence of strings (e.g. the output of a Tokenizer, Normalizer, Lemmatizer, and Stemmer) and drops all the stop words from the input sequences. |
| [Tokenizer](https://nlp.johnsnowlabs.com/docs/en/annotators" \l "tokenizer) | Tokenizes raw text into word pieces, tokens. Identifies tokens with tokenization open standards. A few rules will help customizing it if defaults do not fit user needs. |
| [WordEmbeddings](https://nlp.johnsnowlabs.com/docs/en/annotators" \l "wordembeddings) | Word Embeddings lookup annotator that maps tokens to vectors. |
| **Available Transformers** | **Description** |
| [BertEmbeddings](https://nlp.johnsnowlabs.com/docs/en/transformers" \l "bertembeddings) | Token-level embeddings using BERT. BERT (Bidirectional Encoder Representations from Transformers) provides dense vector representations for natural language by using a deep, pre-trained neural network with the Transformer architecture. |
| [BertSentenceEmbeddings](https://nlp.johnsnowlabs.com/docs/en/transformers" \l "bertsentenceembeddings) | Sentence-level embeddings using BERT. BERT (Bidirectional Encoder Representations from Transformers) provides dense vector representations for natural language by using a deep, pre-trained neural network with the Transformer architecture. |
| [UniversalSentenceEncoder](https://nlp.johnsnowlabs.com/docs/en/transformers" \l "universalsentenceencoder) | The Universal Sentence Encoder encodes text into high dimensional vectors that can be used for text classification, semantic similarity, clustering and other natural language tasks. |

**Training Components Of This Project**

The following is a schematic diagram of the pipeline of this term project:



DocumentAssembler(): Prepares data into a format that is processable by Spark NLP. This is the entry point for every Spark NLP pipeline.

BertSentenceEmbeddings(): Sentence-level embeddings using BERT。The default model is "sent\_small\_bert\_L2\_768".

ClassifierDL(): The ClassifierDL annotator uses a deep learning model (DNNs) that have been built inside TensorFlow and supports up to 100 classes.

**Running on Google Cloud**

***Setting Up Cluster Environment***

Tutorial Source: <https://codelabs.developers.google.com/codelabs/spark-nlp#3>

|  |
| --- |
| Dataproc-->Activate Cloud Shell-->type following shell sentences  CLUSTER\_NAME=my-cluster  REGION=us-east1  BUCKET\_NAME=bm\_reddit   gcloud beta dataproc clusters create ${CLUSTER\_NAME} \      --region ${REGION} \      --metadata 'PIP\_PACKAGES=google-cloud-storage spark-nlp' \      --worker-machine-type n1-standard-8 \      --num-workers 2 \      --image-version 1.4-debian10 \      --initialization-actions gs://dataproc-initialization-actions/python/pip-install.sh \      --optional-components=JUPYTER,ANACONDA \      --enable-component-gateway  \*SparkNLP and Spark version match  Spark NLP version 4.3.1, Apache Spark version: 3.3.2  Or  Spark NLP version 2.4.5, Apache Spark version: 2.4.4 |

***Model Result***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-score | Support |
| Positive | 0.84 | 0.86 | 0.85 | 160, 745 |
| Negative | 0.89 | 0.84 | 0.86 | 159, 811 |
|  |  |  |  |  |
| accuracy |  |  | 0.85 | 320, 556 |
| Macro avg | 0.86 | 0.85 | 0.85 | 320, 566 |
| Weighted avg | 0.86 | 0.85 | 0.85 | 320, 566 |

**References**

1. <https://towardsdatascience.com/text-classification-in-spark-nlp-with-bert-and-universal-sentence-encoders-e644d618ca32>
2. <https://spark.apache.org/docs/latest/ml-guide.html>
3. <https://nlp.johnsnowlabs.com/docs/en/annotators>
4. <https://nlp.johnsnowlabs.com/docs/en/annotators#classifierdl>
5. <https://nlp.johnsnowlabs.com/docs/en/transformers#bertsentenceembeddings>
6. <https://blogs.nvidia.com/blog/2022/03/25/what-is-a-transformer-model/>
7. <https://arxiv.org/abs/1706.03762>