Kaggle Competition

Data preparation

In this section, several tasks are performed to prepare the data.

- Load data: The data is first loaded from various CSV files.
- Extract important feature: After examining the raw data in tweet_DM.json, I found that important features such as hashtags, text, and id are nested within the _source field. These features need to be extracted.
- Saperate data: Based on the id and identification columns in the data_identification.csv file, the data is separated accordingly.
- Assign emotion: Based on the id and emotion columns in the emotion.csv file, each tweet is assigned with its corresponding sentiment.
- Drop irrelevent columns: We assume that _score, _index, _source, _crawldate,and _type are irrelevant features for determining a tweet's sentiment. The hastage is also dropped since too many rows have empty values for this field
- Tokenize: The text is tokenized for latter tasks.
- Save data: The preprocessed data is then saved in a pickle file to avoid repeating the preprocessing steps each time.

Exploratory data analysis (EDA)

- In this section, exploratory data analysis is performed. I discovered that the data is imbalanced. Therefore, a resampling technique is applied to address this issue. However, after the resampling the data decome extremly large.

BoW and TFIDF

- In this section, Bag of Words (BoW) and TF-IDF are utilized to extract features. Given the large vocabulary size in this dataset, I chose 1000 as the dimension of the embedding vector for each tweet.

- A simple Multinomial Naive Bayes classifier is then used for predictions, serving as a baseline model. I found that the performance was similar when using either BoW or TF-IDF.
- Further examined the features chosen by BoW and TF-IDF and found that these features are hardly relevant to the sentiment of a tweet.
- Additionally, a 2-layer neural network is trained using BoW features for 10 epochs, and the performance is slightly better but still poor.

Conculsion:

Simple feature extraction techniques like BoW or TF-IDF may seem too naive for this task.

Word2Vec

To address the limitation that Bag of Words (BoW) and TF-IDF features cannot effectively capture the sentiment in a tweet, Word2Vec features are utilized. In this section, I utilized a pre-trained Word2Vec model from Gensim to obtain word embeddings. To represent each tweet, the embeddings of all its words were summed to create a single vector representation. The predictions are then generated by a neural network. The following outlines the details of my approach and experiments:

- glove-twitter-50
    1. hidden layer: 64-64-8, data: balanced, epoch: 10
    2. hidden layer: 64-128-256-256-124-64-8, data: balanced, epoch: 19
    3. hidden layer: 64-128-256-256-128-64-8, data: unbalanced, epoch: 50
- glove-twitter-100
    1. hidden layer: 128-256-512-512-256-128-8, data: unbalanced, epoch: 30
    2. hidden layer: 128-256-512-512-512-512-256-128-8, data: unbalanced, epoch: 30

- glove-twiter-200

    1. hidden layer: 128-256-512-1024-1024-512-256-128-8, data: unbalanced, epoch: 30

Seq2seq model with attention

- Since attention mechanisms have proven to be highly effective in NLP tasks, in this section a LuongAttention is utilized in a seq2seq model. The validation accuracy show very high accuracy compare to the previous approuch even for just to epoch.