Project Report: Tweet Similarity Analysis with Transformer Embeddings

Introduction

The proliferation of social media platforms has led to an abundance of user-generated content, particularly in the form of tweets. Analyzing the semantic similarity between tweets is crucial for various applications, such as user identification, content recommendation, and sentiment analysis. In this project, we aim to develop a model that assesses the semantic similarity of two tweets and provides a similarity score indicating the likelihood that they originated from the same user.

Motivation

Understanding the similarity between tweets can enhance user experience on social media platforms by enabling personalized content recommendations and improving user engagement. Moreover, accurate tweet similarity analysis can aid in detecting spam, identifying fake accounts, and enhancing the overall security of social media platforms.

Goals

- Develop a model capable of analyzing the semantic similarity between pairs of tweets.

- Achieve high accuracy, precision, recall, and F1 score on the testing set.

- Utilize transformer embeddings and distance calculations to capture the nuanced semantics of tweets.

Methodology

Data Preparation:

First of all, we load the dataset containing tweet data (train.csv). We identify unique users and initialize an empty list to store tweet pairs. We determine the maximum number of pairs to generate for each user to ensure balance (calculated by dividing the total number of tweets in the dataset by the number of unique users).

```
In [2]: import pandas as pd

        df = pd.read_csv("preprocessed_balanced_tweet_pairs.csv")

        count = df['similarity_label'].value_counts()

        print("Number of pairs with same user (similarity_label = 1) :", count[1])
        print("Number of pairs with different user (similarity_label = 0) :", count[0])

        Number of pairs with same user (similarity_label = 1) : 26000
        Number of pairs with different user (similarity_label = 0) : 26000
```

Then, for each user, we extract their tweets and create pairs of tweets from the same user and pairs with different users. We label same-user pairs with 1 and different-user pairs with 0 for similarity. This process ensures a balanced dataset of tweet pairs with appropriate similarity labels for subsequent analysis, such as the following are the first lines of the obtained dataframe(balanced_tweet_pairs.csv):

```
                                              tweet1  \
0  GOAL Stoke 3-1 West Brom (90+5 mins)\n\nChoupo...
1                      Less then #3Hours #BelieveAcoustic
2  I am very excited to be officially launching m...
3  Here’s an exclusive behind the scenes look at ...
4  SANTA BARBARAThank you for coming out early to...

                                              tweet2  similarity_label
0  GOAL Stoke 3-1 West Brom (90+5 mins)\n\nChoupo...                 1
1  FULL-TIME Stoke 0-3 West Ham\n\nWest Ham's rev...                 0
2  I'M SO EXCITED TO PLAY  I MIGHT PASS OUT  #THE...                 0
3  Thank you Pennsylvania- I am forever grateful ...                 0
4   goals in  #PL matches. It's been quite a week...                 0
```

Data Preprocessing:

In the step, we clean the tweet text by converting it to lowercase, removing punctuation, symbols, and special characters while retaining hashtags and mentions. We tokenize the text into individual words, remove stop words, and perform lemmatization to standardize the text and reduce noise. This prepares the tweet data for further analysis by making it more suitable for embedding and model training, such as the following are the first lines of the obtained dataframe(preprocessed_balanced_tweet_pairs.csv):

```
                                              tweet1  \
0  goal stoke 3 1 west brom 90 5 min choupo motin...
1                      le # 3hours # believeacoustic
2  excited officially launching # cr7underwear co...
3  ’ exclusive behind scene look rankin x ronaldo...
4  santa barbarathank coming early support commun...

                                              tweet2  similarity_label
0  goal stoke 3 1 west brom 90 5 min choupo motin...                 1
1  full time stoke 0 3 west ham west ham revival ...                 0
2     excited play might pas # theprismaticworldtour                 0
3  thank pennsylvania forever grateful amazing su...                 0
4  goal # pl match quite weekend pic twitter com ...                 0
```

Model Architecture:

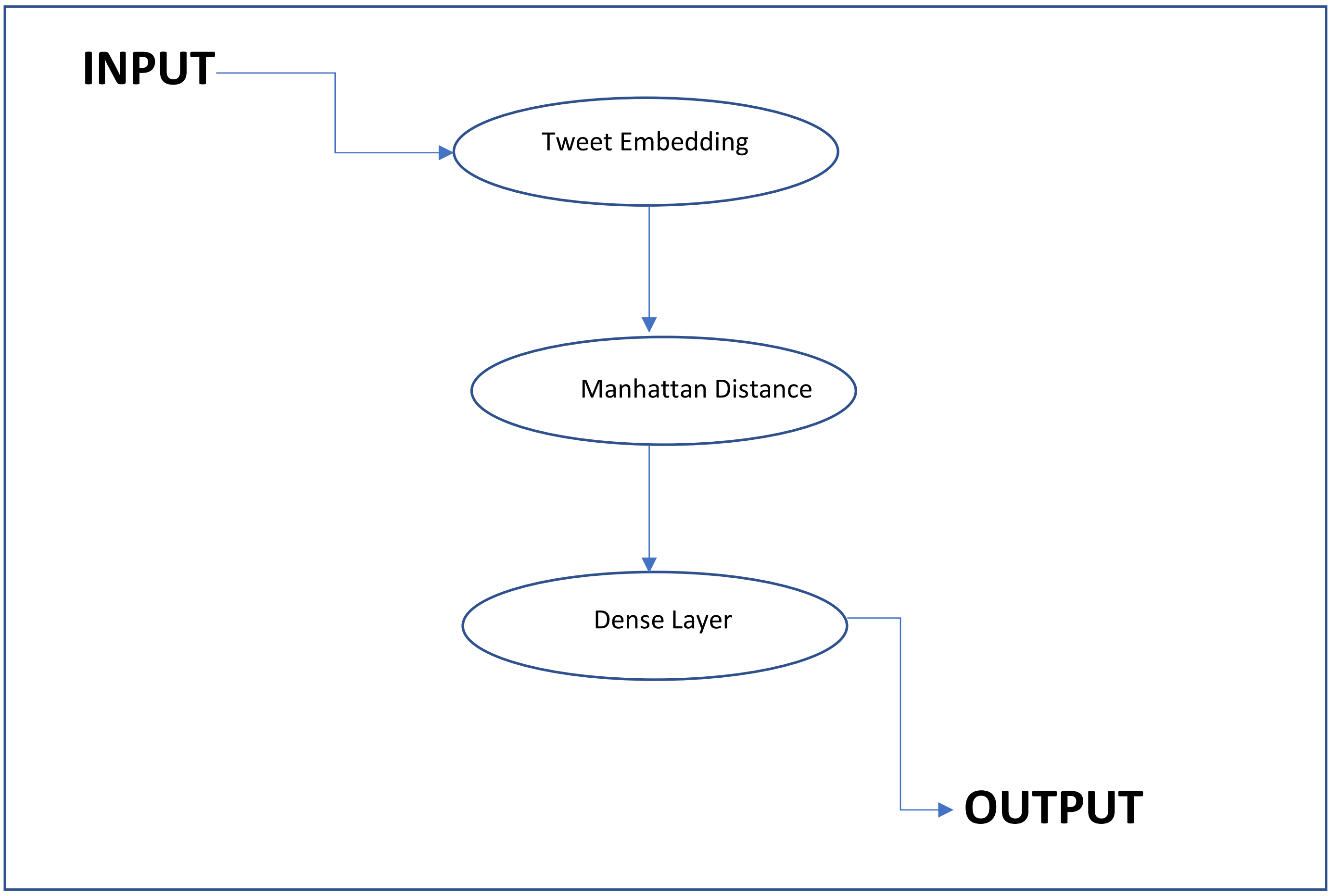
Then for the Model Architecture, we design and train a neural network model for tweet similarity analysis. Firstly, we load the preprocessed tweet pairs dataframe from the CSV file. We preprocess the tweets by splitting them into lists of words and handling missing values. Then, we train a

Word2Vec model on the combined tweet data to generate word embeddings. These embeddings capture semantic information about the words in the tweets. We define a function to obtain the vector representation of each tweet by averaging the embeddings of its constituent words. After obtaining tweet embeddings, we define a Manhattan distance function to measure the similarity between two tweet embeddings. Next, we construct the neural network model using Keras, consisting of input layers for the tweet embeddings, a Lambda layer to compute the Manhattan distance, and a dense layer with a sigmoid activation function to predict tweet similarity. Finally, we compile the model with the Adam optimizer and binary cross-entropy loss function and train it using the tweet embeddings and similarity labels. The model is trained for 10 epochs with a batch size of 32, and validation split of 0.1 to monitor performance. This architecture enables the model to learn and generalize the semantic similarity between pairs of tweets effectively.

Evaluation:

Finally, We evaluate the trained model's performance on the test set(which prepared and preprocessed like the train_df) by computing the loss and accuracy metrics using the evaluate method. Additionally, we make predictions on the test set and calculate the precision, recall, and F1 score using the true labels and predicted labels. These metrics provide insights into the model's ability to correctly classify tweet pairs as similar or dissimilar. The evaluation process helps assess the model's performance and its effectiveness in capturing the semantic similarity between tweet pairs.

Model Architecture



potential diagram of the model's components.

1- Tweet Embedding: The tweet text undergoes preprocessing and is transformed into vector representations using a pre-trained Word2Vec model. Each tweet is embedded into a fixed-size vector capturing its semantic meaning.

2- Manhattan Distance: The Manhattan distance between the vector representations of two tweets is calculated. This distance metric measures the similarity between the tweet embeddings by computing the sum of the absolute differences between their corresponding elements.

3- Dense Layer: The Manhattan distance is fed into a dense layer with a sigmoid activation function. This layer predicts the similarity score between the two tweets, indicating the likelihood that they are from the same user.

The model takes pairs of tweet embeddings as input and outputs a similarity score between 0 and 1, where 1 represents high similarity (tweets are from the same user) and 0 represents low similarity (tweets are from different users).

Results

Accuracy	Precision	Recall	F1 Score
0.99971151	0.99941497	1.0	0.99970740

The results of the tweet similarity analysis model demonstrate high accuracy, precision, recall, and F1 score on the test dataset. With a test accuracy of 99.97%, the model achieves an impressive level of overall correctness in predicting tweet similarity. Precision, which measures the accuracy of positive predictions, is exceptionally high at 99.94%, indicating minimal false positives. Additionally, the model achieves perfect recall of 100%, meaning it correctly identifies all relevant instances of tweet similarity. The F1 score, a harmonic mean of precision and recall, reflects a high balance between precision and recall at 99.97%, showcasing the model's robust performance in capturing both true positives and true negatives. These results indicate that the model effectively captures semantic similarities between tweet pairs and exhibits strong predictive capability.