TwitterHateSpeechDetection

- Project Increment 2

CSCE 5290: Natural Language Processing, Spring 2022

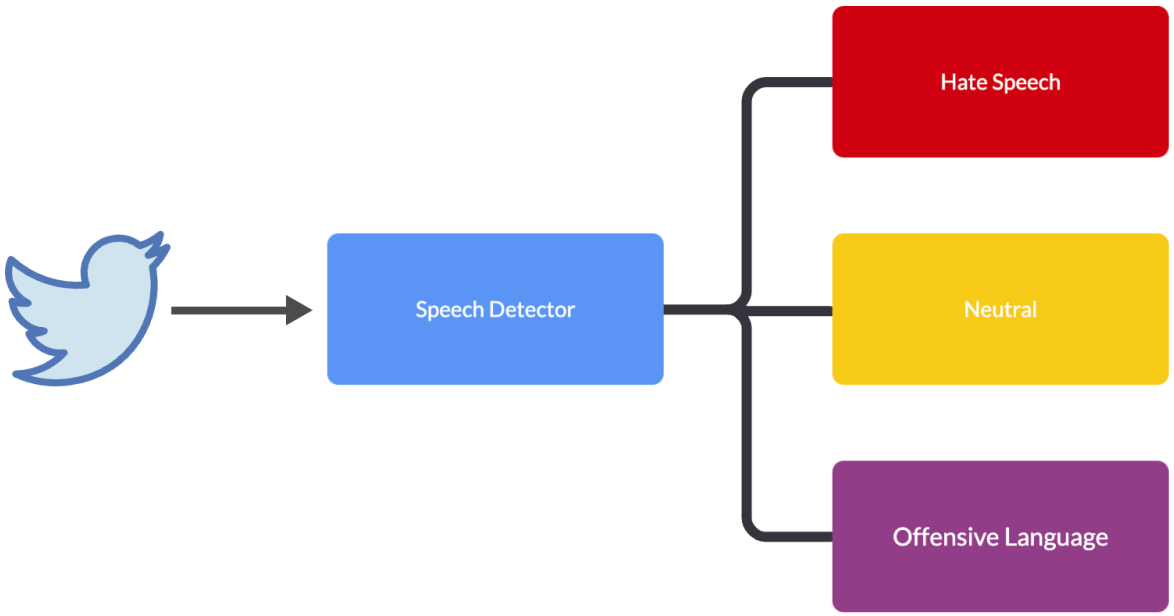
Instructor: *Dr Sayed Khushal Shah* (sayed.shah@unt.edu)

By: *Manoj Kolluri (manojkolluri@my.unt.edu)*

GitHub Link: https://github.com/ManojKolluri/NLP-Project-CSCE_5290

Introduction

Idea Description



The basic idea behind this proposal is to develop machine learning and deep learning models that will be able to read a tweet and classify it to be belonging to one of 3 categories which are Hate Speech, Offensive Language and Neutral Language. At the end of the project, we would also like to integrate a User Interface which would allow the end user to be able to interact with the models and give a custom tweet as an input to these models and check the results produced by the models. Also, before integrating the User interface with the models the models would first be evaluated using certain evaluation metrics such as accuracy, F1 score, precision, sensitivity, and a confusion matrix using a testing dataset which would be separated from the main dataset prior to the training and validation processes of the models. Prior to the training process we would also be applying various text cleaning and NLP preprocessing

techniques like stemming and lemmatization to remove unwanted data from the text used to train these models.

Motivation & Significance:

Over the last 2 decades there has been a tremendous amount of growth in social media both in terms of its scale and in terms of its importance to a huge population of individuals and communities all around the world as one the major means of communication. However, the nature of these social media platforms is built in such a way that it allows any person in the world with a proper access to that platform to post or share any kind of information or express their opinion to millions of people using that platform which might sometimes be inappropriate or even repugnant. Amongst these kinds of inappropriate Hate Speech which is an abusive writing that usually expresses a certain amount of prejudice or ill intent against a particular person, or a group based on race, sex, or religion is one of the most important and dangerous aspects. Uncontrolled spread of such kind of speech can be damaging to the society in many ways therefore there is a need to automate the process of detecting and filtering out these kinds of content from the social media platforms.

The application that we are proposing to build in this project is a text classifier that is able to read a tweet from twitter which is one of the world's most famous social media platforms and try to classify the text belonging to either the category of hate speech, offensive speech or simply just neutral. We would also create a friendly user interface into which the user may be able type or enter a tweet and the models integrated within the UI classifies the tweet and generates an instant result. This would allow the users to gain a practical understanding of how these techniques work.

Objectives & Features:

The major objective of this project is to develop a certain number of machine learning and deep learning models and train them to be able to identify hate speech in twitter tweets and categorize them instantly upon giving a tweet as an input to the user interface.

Technical Goals:

1. Using different types of NLP techniques to preprocess and clean the dataset.
2. Segregating the dataset for testing and training proposes.
3. Creating Machine Learning models and evaluating them.
4. Creating a Deep Learning Model that takes tweets as inputs and evaluate its performance against the machine learning models.

Dataset

Data Collection

The dataset used in this particular project is called Hate Speech and Offensive Language Dataset and was created using data from twitter. It is commonly used in research of hate speech detection and contains a csv file with 7 attributes and 24783 entries in total. The tweets in this dataset belong to 3 different categories which are hate speech, offensive language, and neutral. This dataset was gathered from Kaggle [4].

Related Work

1. **Deep Learning for Hate Speech Detection in Tweets** [1]

   This article was published in the year 2017 by Pinkesh Badjatiya. According to the authors hate speech detection on twitter is critical for many different range of applications like building ai chat bots, controversial event extraction, sentiment analysis, etc. the author define this task as being able to classify and filter out a tweet which might be racist, sexist or neither. In the paper the authors performed many extensive experiments using multiple deep learning architectures and attempt to learn semantic word embeddings to handle complexities. The dataset used by the authors is the 16K annotated tweets dataset.

2. **Hate speech detection: Challenges and solutions** [2]

   This article was published in the year 2019 by Sean MacAvaney. According to the authors the spread of hate speech continues to grow as online content and the spread of the internet keeps to grow. In this paper the authors examined various different challenges faced by most automated online approaches in the task of detecting such kind of inappropriate content. The authors also mention that one of the most common difficulties is the limited availability of data epically when it comes to other languages or subtitles.

3. **Towards generalisable hate speech detection: a review on obstacles and solutions** [3]

   This article was published in the year 2021 by Wenjie Yin. According to the author hate speeches are one of the most harmful contents online which promote hate towards an individual or a group based on their perceived aspects of identification like gender, sexual orientation, race,

and religion. The author says that it has recently been observed that many hate speech detection tools have been performing poorly if new data is shown. This paper is basically a survey that in short summarizes why hate speech models struggle to generalize and suggests how to overcome this obstacle.

Implementation

Data Pre-Processing

- Before we initialize the preprocessing phase we first checked if there were any missing data in the dataset.
- After verifying this we then moved on to apply different kinds of NLP techniques to the text.
- we first convert all the letters of the tweet from the data frame from upper case if there are any to lower case alphabets.
- we then went on and removed all special characters like ‘, . ? -’ from our texts which would serve no real purpose in the classification.
- Then we removed all '\n' and '\t', extra spaces, quoting text, and progressive pronouns from the text as well.
- After we were done with the per-processing part we encode the data using a bag of words approach and then split the dataset into training and testing sets.

Models

We built 3 Machine Learning models in order to classify the datasets and compared their results. The Machine Learning models we used are Random Forest classifier, Decision tree classifier, and AdaBoost classifier.

Random Forest Classifier

After applying the stratified split and creating training and testing datasets we created a Random Forest Model by using the scikit learn and trained the model using the training dataset as shown in the snippet below.

Using Random Forest Classifier as the Model and printing evaluating it using confusion matrix

```
clf = RandomForestClassifier(n_estimators=10)
clf = clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("accuracy is: ",accuracy)
CM = confusion_matrix(y_test, y_pred)
plot_confusion_matrix(CM, classes = range(3))
```

Decision tree Model

After applying the stratified split and creating training and testing datasets we created a Decision tree Model by using the scikit learn and trained the model using the training dataset as shown in the snippet below.

Using Decision tree as the Model and printing evaluating it using confusion matrix

```
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("accuracy is: ",accuracy)
CM = confusion_matrix(y_test, y_pred)
plot_confusion_matrix(CM, classes = range(3))
```

AdaBoost Model

After applying the stratified split and creating training and testing datasets we created a Adaboost Model by using the scikit learn and trained the model using the training dataset as shown in the snippet below.

Using AdaBoost Classifier as the Model and printing evaluating it using confusion matrix

```
clf = AdaBoostClassifier(n_estimators=100)
clf = clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("accuracy is: ",accuracy)
CM = confusion_matrix(y_test, y_pred)
plot_confusion_matrix(CM, classes = range(3))
```

LSTM Model:

We designed and trained our Long-Short-Term Memory Model or LSTM model and trained it on the training dataset for 25 epochs and printed the and used the testing data set for both validation of the model during the training process and testing it later on in the testing process and this can be seen in the code segment below.

Creating and Training an LSTM Model

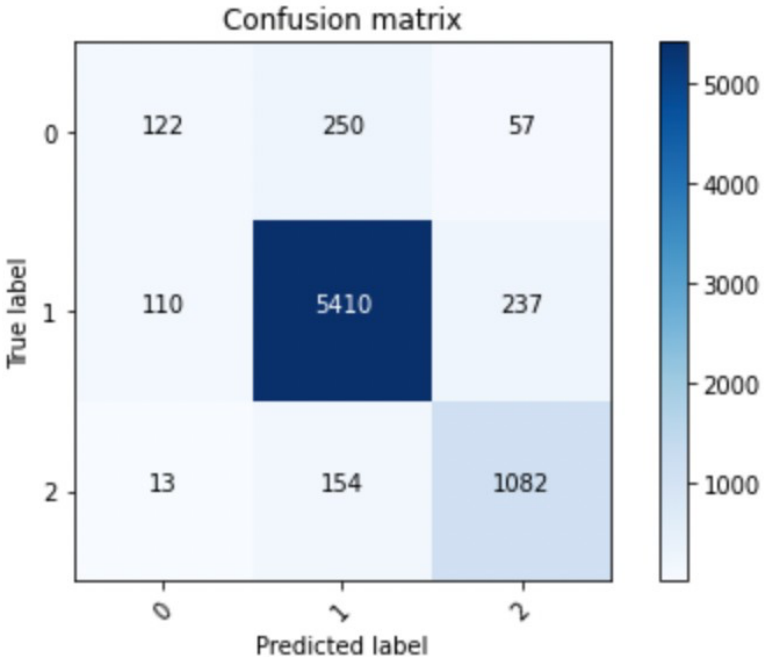
```
model = Sequential()
model.add(Embedding(232337, 100, input_length=X_train.shape[1]))
model.add(SpatialDropout1D(0.2))
model.add(LSTM(20, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(3, activation='softmax'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

epochs = 25
batch_size = 64

history = model.fit(X_train, y_train,validation_data = (X_test,y_test), epochs=epochs, batch_size=batch_size)
```

Preliminary Results

Random Forest Model:

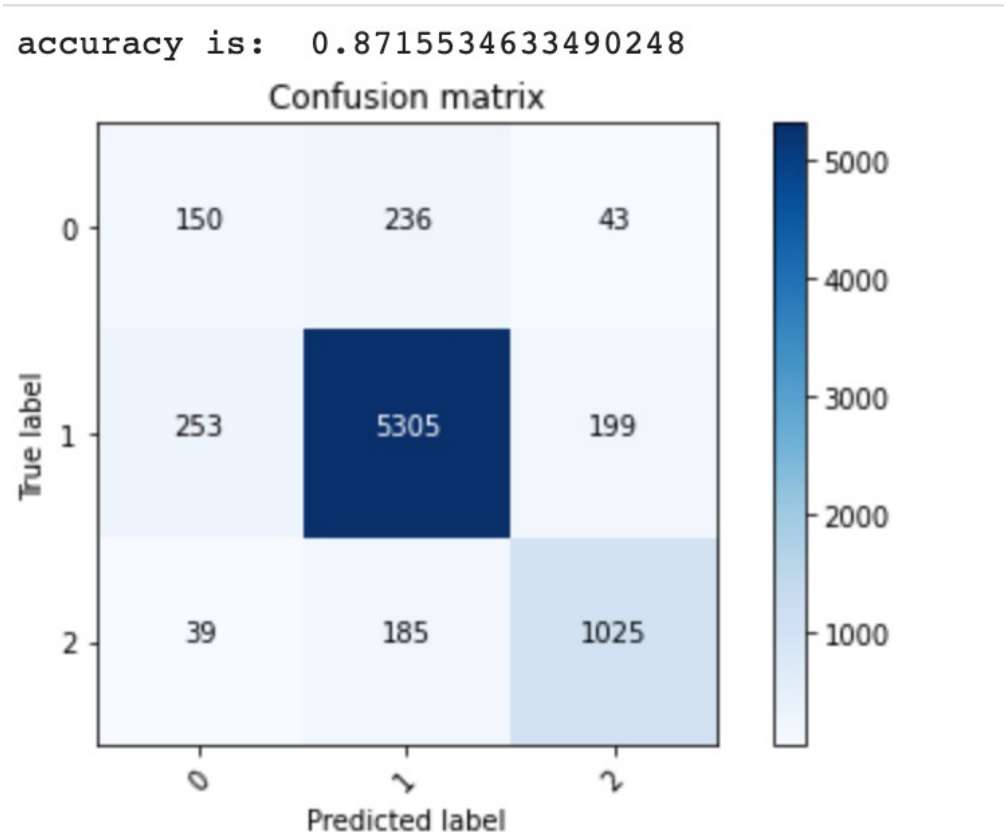
After testing the random forest model with the testing dataset we printed its accuracy and also plotted its confusion matrix shown in the figure below.

```
accuracy is:  0.8895763281775386
```



Decision tree Model:

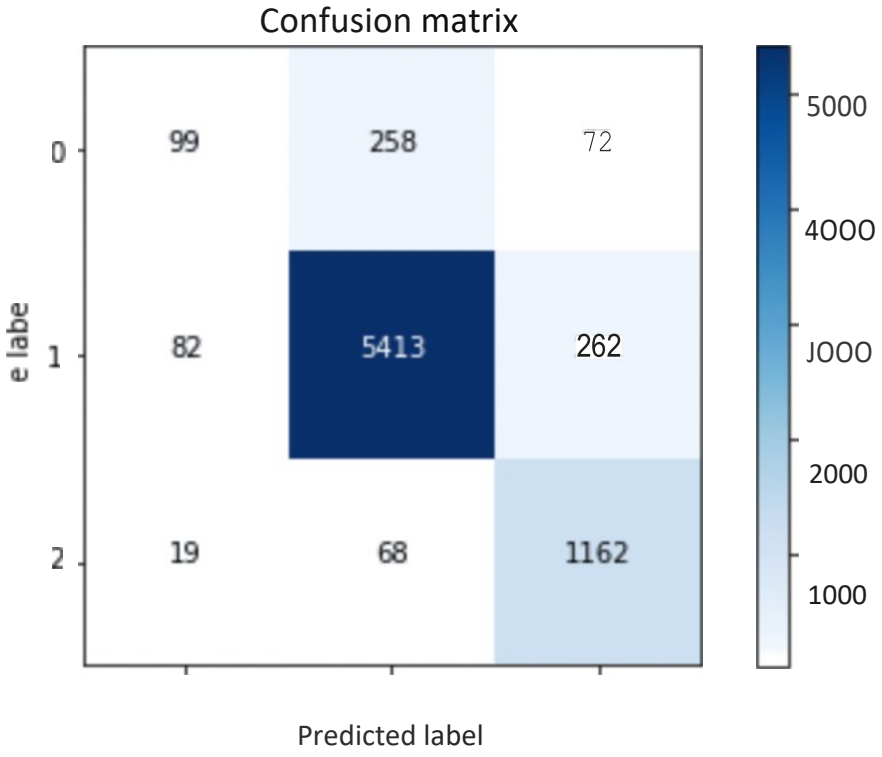
After testing the decision tree model with the testing dataset we printed its accuracy and also plotted its confusion matrix shown in the figure below.



Adaboost Model:

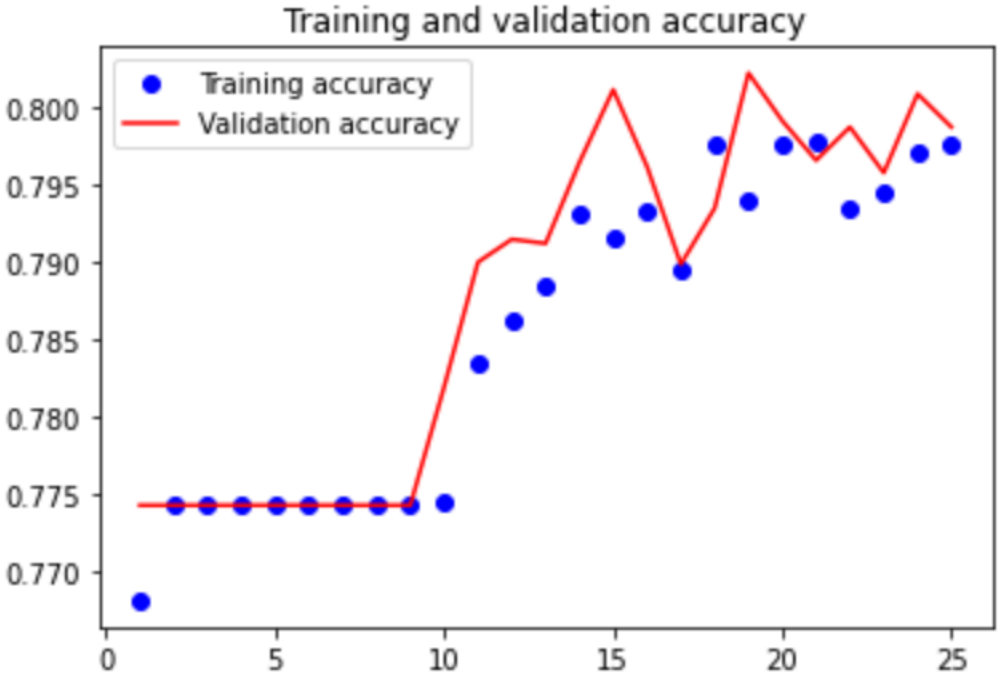
After testing the decision tree model with the testing dataset we printed its accuracy and also plotted its confusion matrix shown in the figure below.

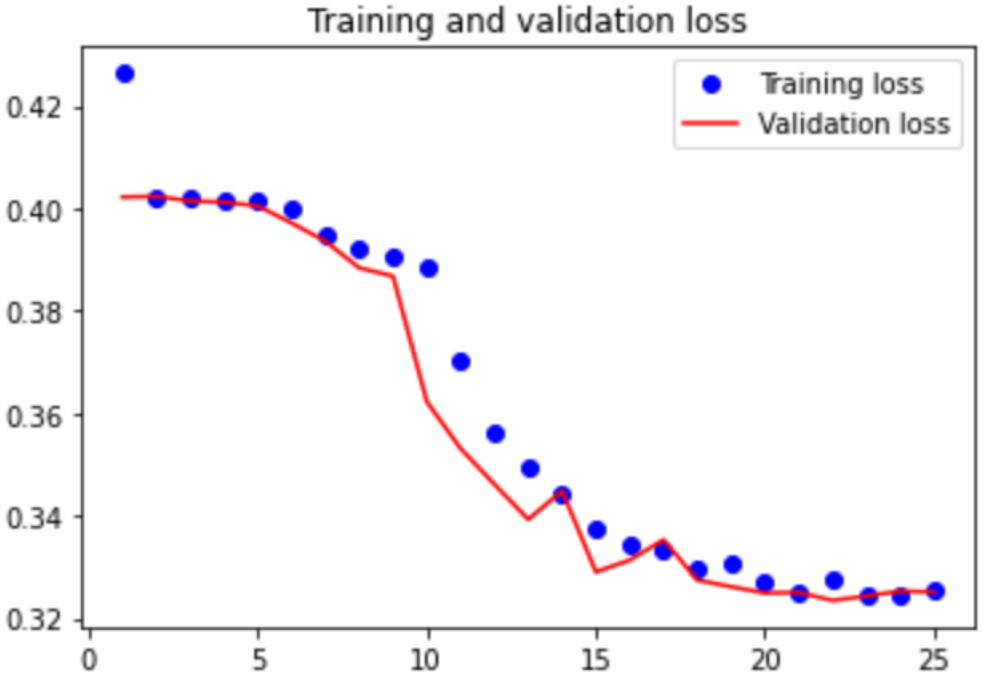
accuracy is: 0.8976462676529926



LSTM Model:

After the training process we observed the graph shown in the figure below and achieved an accuracy of 79.88% during validation.





The above figure shows the graph for the training and validation accuracies and losses during training.

References

1. Badjatiya,P.,Gupta,S.,Gupta,M.,&Varma,V.(2017).Deeplearningforhatespeechdetectionin tweets. doi:10.1145/3041021.3054223

2. MacAvaney,S.,Yao,H.,Yang,E.,Russell,K.,Goharian,N.,&Frieder,O.(2019).Hatespeech detection: Challenges and solutions. *PLoS ONE, 14*(8), e0221152. doi:10.1371/journal.pone.0221152

3. Yin,W.,&Zubiaga,A.(2021).Towardsgeneralisablehatespeechdetection:Areviewonobstacles and solutions. *PeerJ. Computer Science, 7*, e598. doi:10.7717/peerj-cs.598

4. https://www.kaggle.com/datasets/mrmorj/hate-speech-and-offensive-language-dataset