CE807 – Assignment 2 - Final Practical Text Analytics and Report

Student id: 2201110

Abstract

Text analytics commonly involves performing text classification, which is a standard task. The specific task at hand is to classify offensive speech using a subset of the OLID dataset. The provided dataset has already been split into train, validation, and test sets.The objective is to determine whether a given data point contains offensive language or not, focusing on Level A: Offensive language Detection.

1 Materials

- Code
- Google Drive Folder containing models and saved outputs
- Presentation

2 Model Selection (Task 1)

Different models were implemented and their performance was examined to determine the most appropriate method for classifying a tweet as offensive (OFF) or not (NOT). Logistic Regression was implemented as Model 1 since it is used for binary classification problems and for Model 2 CNN was implemented with global pooling as they can handle large noisy datasets.

2.1 Summary of 2 selected Models

Logistic Regression as model 1 was applied to dataset after preprocessing the data. It was trained on different ditribution of dataset and was tested on test data where the performance was measured through F1-score,Recall and Accuracy. Accuracy of 80

For classfiying offensive tweets, the Model 2 was deployed using the neural networks as they are currently the most effective approach in natural language processing. Specifically, Convolutional Neural Networks (CNNs) was employed with an embedding layer, a 1D Convolutional layer, a GlobalMaxPooling1D layer, and three Dense layers, with a Dropout layer added between each Dense layer.Model was trained and tested on different distribution of dataset and the accuray was good when data was equally distributed.

2.2 Critical discussion and justification of model selection

The training and the testing data set used for both the models which was provided from subset of OLID dataset. it containted different tweets from users and each entry was annotated using two labels OFF and NOT representing Offensive language and Non offensive language respectively. The data set had a total of 3078 instances with 2076 instances under the category OFF and 1002 instances under the non offensive category which shows the imbalanced nature of the data in the data set.A separate validation set was provided was predicting the accuracy of model before its implemented on test dataset.Different classification problem with Deep learning models was implemented comparing the performance with Machine language model.

2.2.1 Model 1: Logistic Regression

Logistic regression was the one of the machine learning model for classification, for binary classification tasks which classified offensive language (represented as 1) versus non-offensive language (represented as 0). Prior to training, preprocessing steps included removal of duplicate entries in the dataset, removal of unnecessary labels and special characters.Next, the function performs lemmatization, which is the process of reducing a word to its base or root form. It downloads the WordNet corpus for lemmatization and initializes a stemmer for English..It then applies the stemmer to each word in the tweet using the SnowballStemmer function from the nltk library. After lemmatization, the function removes stop words from the tweet

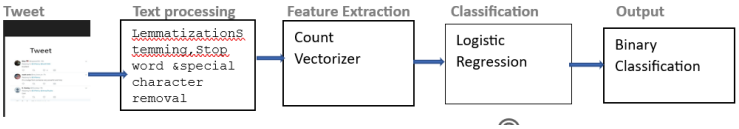

Figure 1: Pipeline of Model 1 - Logistic Regression

Model: "sequential_4"

Layer (type)	Output Shape	Param #
embedding_4 (Embedding)	(None, 30, 64)	64000
conv1d_4 (Conv1D)	(None, 26, 128)	41088
global_max_pooling1d_4 (GlobalMaxPooling1D)	(None, 128)	0
dense_12 (Dense)	(None, 512)	66048
dropout_8 (Dropout)	(None, 512)	0
dense_13 (Dense)	(None, 8)	4104
dropout_9 (Dropout)	(None, 8)	0
dense_14 (Dense)	(None, 1)	9

Total params: 175,249
Trainable params: 175,249
Non-trainable params: 0

Figure 2: CNN pipeline

using the nltk.corpus.stopwords function and then transformed the data into a matrix of vectors using CountVectorizer function. If split is set to 'train', the function fits the CountVectorizer object on the training data and returns the fitted object and the transformed data. If split is set to 'test', the function only transforms the test data using the fitted CountVectorizer object and returns the transformed data. Once trained, we used the model to predict the labels for the given test dataset. Our training data results for this model were an accuracy of 78.37

2.2.2 Model 2: CNN

The experiments included different approaches to handling uppercase and lowercase words, swear words, non-alphanumeric tokens, and out-of-vocabulary words. The results show that lowercasing all words improves the system's performance, while including non-alphanumeric tokens decreases it.The labels are converted to binary format for training purposes. Then, the function performs lemmatization and stop-word removal using NLTK and spaCy libraries.Further using Keras preprocessing library to convert the text data into a numerical format that can be used as input to a neural network model. First, it creates a tokenizer object with a vocabulary size of 1000 words (i.e., only the 1000 most frequent words will be used for the model input) and an out-of-vocabulary (oov) token to represent words that are not in the vocabulary. The tokenizer is then fitted on the training data tweets, which generates a word index for each word in the vocabulary.Next, the texts to sequences() method is used to convert each tweet in the training and validation datasets into a sequence of integers based on the word index generated by the tokenizer. This allows each word to be represented as a unique integer in the sequence.Finally, the pad sequences() method is used to ensure that all sequences have the same length (30 in this case) by padding the sequences with zeros or truncating the sequences if they are longer than the specified maximum length. This is necessary because neural networks require input data to be of a fixed size.

3 Design and implementation of Classifiers (Task 2)

The text reports the results of experiments conducted on a system that classifies tweets as offensive or not. The system was optimized for F1-measure, and the experiments focused on improving the system's performance. (Ref)

3.0.1 Model 1: Logistic Regression

The system was tuned for various hyperparameter tuning using grid search.First, the hyperparameters being tuned are 'C' and 'solver'. The 'C' parameter controls the inverse of regularization strength, and 'solver' is the algorithm to use in the optimization

problem. Next, a GridSearchCV object is created with the LogisticRegression() model, the parameter grid, and a 5-fold cross-validation. GridSearchCV performs an exhaustive search over all the parameter combinations specified in the parameter grid, and evaluates the performance of each model using cross-validation.After fitting the model using grid search, the best cross-validation score and best parameters are determined.(Pedersen)

3.0.2 Model 2: Convolutional Neural Network

This code is building a Convolutional Neural Network (CNN) model using Keras for sentiment analysis on tweets.The text data is tokenized and then converted into sequences of integers.Further pad sequences() method is used to ensure that all the sequences have the same length (30), by padding with zeros or truncating the sequences.An embedding layer is added to convert word indices into dense vectors of fixed size to extract features from the text data, with 128 filters and kernel size of 5.A global max pooling layer is added to reduce the output of the convolution layer to a fixed size vector.Two fully connected (Dense) layers with ReLU activation are added to the model. Each layer has 512 hidden units, and a dropout of 0.4 is added after each dense layer to reduce overfitting. A final dense layer with sigmoid activation is added to predict the sentiment of the input tweet.Finally, the model is compiled by specifying the loss function, optimizer, and evaluation metric to be used during training with EarlyStopping callback to stop the training if the validation loss does not improve for 5 epochs.

Dataset	Total	% OFF	% NOT
Train	12313	33.2	66.8
Valid	927	33.2	66.8
Test	860	27.9	72.1

Table 1: Dataset Details

As seen from above table our test dataset was more biased towards the not offensive tweets whereas train and validated dataset was biased but with 66.4 and 66.8. After training the models on different datasets and then testing them on validation set we finally apply it ton test dataset to achieve the final F1 score as noted from below.Concluding that our Model 1 implementation with Logistic regression performed better than our neural network with CNN.

Model	F1 Score
Model 1	0.72
Model 2	0.64

Table 2: Model Performance

4 Data Size Effect (Task 3)

The train test split() function from the scikit-learn library is used to split the dataset into a train subset and a test subset.the total number of tweet changes with the stratify parameter is set to train data[target], which is the column in the dataset that contains the target variable labels.The loop iterates over a list of split ratios [0.25, 0.5, 0.75]. For each split ratio, the train subset is saved to a CSV file named according to the split ratio.

In logistic regression , Various hyperparameters were given to the function.Various values of C was tried to avoid overfitting. Solver was used in the optimization process to find the optimal weights for the logistic regression model. There are different solver algorithms available in scikit-learn, such as newton-cg, lbfgs, liblinear, etc.cv: cv is set to 5, which means that the data will be split into 5 folds and the model will be trained and evaluated 5 times, with each fold used as the test set once. For CNN, we will tune the number of words based on word frequency this was adjusted based on complexity of dataset. filters: The number of filters in the Conv1D layer was tuned to controls the number of features that the model can extract from the text. kernel size: Various kernel sizes were tried to fit the model activation: The activation function used in conv1d and dense layer were used from relu and sigmoid kernel regularizer and bias regularizer: The regularization parameters for the Conv1D and Dense layers. Regularization can help prevent overfitting by adding a penalty term to the loss function.

Data %	Total	% OFF	% NOT
25%	3078	32.62	67.4
50%	6156	33.1	66.9
75%	9234	33.5	66.5
100%	12313	33.2	66.8

Table 3: Train Dataset Statistics of Different Size

Both models are compared using accuracy score as the dataset increased the accuracy score increased and decreased for dataset distribution of

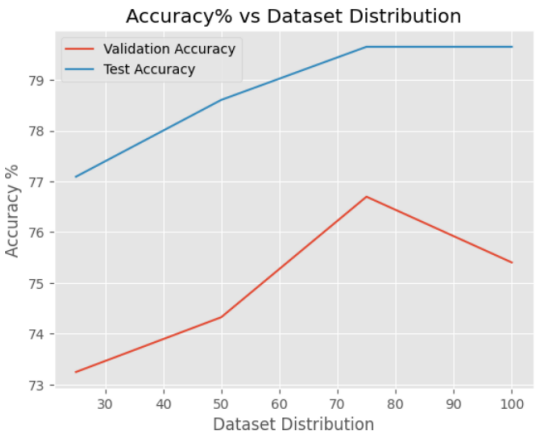

Figure 3: Comparision of Models based on Different data sizes.

100 for Logistic regression model. The highest accuracy achieved was 77 percentage on training dataset and same model when applied on test dataset accuracy of 80 percentage was achieved.

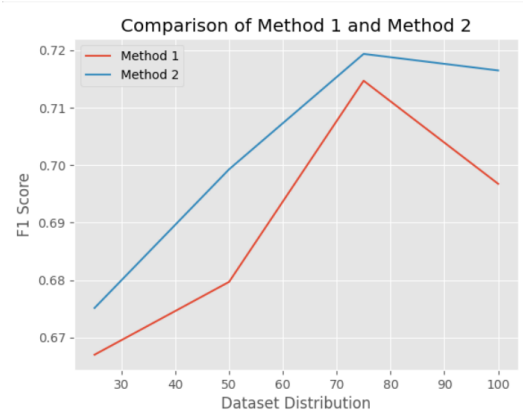

Figure 4: Comparison of F1 score for dataset distribution

For evaluating performance even F1 score was calculated which is to evaluate the distribution of the classes in the data is imbalanced, as it takes into account both false positives and false negatives. A high F1 score indicates that the model has good precision and recall, meaning that it is able to accurately identify both positive and negative instances in the data. F1 score 0f 0.72 was achieved on the test dataset of 75 percentage distribution.

5 Summary (Task 4)

Based on the obtained results, it can be concluded that for the given dataset the logistic regression machine learning models has performed better than the deep learning models CNN for classifying offensive language. Though with more dataset the deep learning model transformer based models has done the more accurate predictions compared to machine learning models.(C Jerin Mahibha, 2021)

5.1 Discussion of work carried out

The different tweets were misclassified as they were missing URL links or were in different language other than English or they were out of context. Logistic regression and CNN were finally selected as model1 and model 2 but since the data was bias was towards the non offensive tweets the F1 score for NOT is higher.

5.2 Lessons Learned

Identifying offensive language is a complex problem without a clear definition, as a statement's offensiveness depends on its context and the source and target of the statement. Offensive statements may also be true, making it difficult to establish boundaries. To address this issue, specific boundaries should be defined for a particular corpus or task, taking into account the target, source, and context of the language. The preprocessing has to be done mulitple times and various methods of stemming lemetization has to be followed to increase the accuracy.Also we should make sure there is no bias in the dataset.

6 Conlusion

Development on more fine grained approached can be focussed upon which also use make use of data such as images videos for for offensive language detection.Also investigating the context of tweets can help increase the accuracy. Exploring more fair and reliable models that are not bias towards any subject. Focus should be on developing more effective evaluation metrics for offensive language detection that take into account the complexity of the task and the various factors that influence offensiveness.

References

Durairaj Thenmozhi Sundar Arunima C Jerin Mahibha, Sampath Kayalvizhi. 2021. *Offensive Language Identification using MachineLearning and Deep Learning Techniques*.

Example %	GT	M1(100%)	M2(100%)	Remark%
smack url	NOT	NOT	OFF	Model couldnt read url
And yet everyone seems to lack it URL	NOT	NOT	NOT	
@USER Antifa has TS level influence. It's scary.	OFF	NOT	NOT	Lack of context
6ix9ine aus den speakern, fick deine political correctness	NOT	NOT	NOT	Spell errors
MonbebeSelcaDay she is beautiful URL	NOT	NOT	NOT	

Table 4: Comparing two Model's using 100% data: Sample Examples and model output using Model 1 & 2. GT (Ground Truth) is provided in the test.csv file.

Example %	GT	M1(25%)	M1(50%)	M1(75%)	M1(100%)
Season2 scary Er while we are on the subject do you condemn Antifa and those who take Soros money?	NOT	NOT	NOT	NOT	NOT
Emmys sooo lovely I love the dress colour and design looks like fairy tales and she is very tender	NOT	NOT	NOT	NOT	NOT
ChristineBlaseyFord is your Kavanaugh accuser					
Liberals try this EVERY timeConfirmJudgeKavanaugh	OFF	NOT	NOT	NOT	NOT
@USER Antifa has TS level influence. It's scary.	OFF	NOT	NOT	NOT	NOT
Feinstein you cant say anything truthful.					
You are a loers and liar	NOT	NOT	NOT	NOT	NOT

Table 5: Comparing Model Size: Sample Examples and model output using Model 1 with different Data Size

Example %	GT	M2(25%)	M2(50%)	M2(75%)	M2(100%)
crazyredhead angry orange cheeto claim president acceptdenying life lost puerto rico monste	OFF	OFF	NOT	NOT	OFF
bitch m fucking coming url	OFF	OFF	NOT	NOT	NOT
christineblaseyford kavanaugh accuser liberal try time confirmjudgekavanaugh	OFF	OFF	OFF	OFF	OFF
account quasianonymous completely anon twitter ban quasianon ill anon account tweet insane shit drown twitter bullshit banned ill	OFF	OFF	OFF	OFF	OFF
celebration germany german yeah kind expected happen western world securityfears maga mega mbga mcga conservative patriot	NOT	NOT	NOT	NOT	NOT

Table 6: Comparing Model Size: Sample Examples and model output using Model 2 with different Data Size

Ted Pedersen. Duluth at SemEval-2020 Task 12: Offensive Tweet Identification in English with Logistic Regression.