

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SINGAPORE

CZ4045 Natural Language Processing

Course Assignment Report

Group 16

Matric No.	Name
U1921577D	Alicia Chua Jieying
U1923151C	Anil Ankitha
U1922129L	Arora Srishti
U2022593D	Justyn Phoa Zairen
U2022059F	Liu Changsong
U1921903J	Tan Mei Xuan

Table of Contents

1 Introduction	3
2 Crawling	4
2.1 Methodology for Crawling of Data	4
2.2 Insights Derived	5
2.3 Text Corpus Information	6
3 Data Preparation	7
3.1 Pre-processing of Text Corpus	7
3.1.1 Merging Twitter and Reddit CSVs	7
3.1.2 Data Preprocessing	7
3.1.3 Stemming, Lemmatization and Removal of Stopwords	8
3.1.4 Word Count	9
3.2 Manual Labeling of Collected Data	9
4 Classification	10
4.1 Classification approach	10
4.1.1 K-Nearest Neighbors (KNN)	10
4.1.2 Random Forest (RF)	10
4.1.3 Naive Bayes	11
4.1.4 Doc2Vec	11
4.2 Evaluation Metrics	12
4.2.1 Overall Result	13
4.3 Performance Metrics	15
4.3.1 Time taken per Classification Model	15
5 Innovations for Enhancing Classification	16
5.1 Ensemble Learning via Model Stacking	16
5.2 LSTM and Bi-Directional LSTM	18
5.2.1 Motivation	18
5.2.2 Long Short Term Memory	19
5.2.3 Bi-Directional Long Short-Term Memory	20
5.2.4 Architecture	21
5.2.5 Results	21
5.3 Attention-based Bi-Directional LSTM	22
5.3.1 Architecture	24
5.3.2 Results	25
5.4 User Interface (UI)	25
6 Conclusion	25
7 Limitations and Future Improvements	27

8 References	29
9 Submission Links	29

1 Introduction

Gun control is one of the most controversial issues in the United States (US), with the debate on whether gun control is a restriction on an individual's right and whether there is a correlation between gun possession and crime rates. In recent years, gun violence has surged amid the COVID-19 pandemic. In 2021, gun-related deaths accounted for more than forty-five thousand people, the highest toll in decades; and this figure is projected to increase. This has raised several questions about the existing gun control policy in the US. Many debates and discussions between gun-control advocates and gun-rights proponents have been ongoing on social media platforms.

This paper aims to capture the public opinion on gun laws in the US in terms of pro- and anti-gun sentiments. The corpus that contains public opinion on gun laws is crawled from 2 social media platforms, Twitter, and Reddit. We will perform sentiment analysis in terms of polarity and subjectivity detection. The evaluation results are demonstrated using classification approaches.

2 Crawling

2.1 Methodology for Crawling of Data

Crawling Twitter

The data is crawled using snsrape library instead of the popular python library Tweepy. This is because Tweepy library has a scraping limit and no access to historical data. However, as we want our dataset to be more comprehensive, we crawled from the year 2020 to 2022.

As our tweets are only targeted at gun laws in the US, our specific keyword will consist of both related words about the country ("USA" OR "the US") and gun laws (#gunlaws OR "gun control" OR "gun laws" OR "gun violence"). To get different time periods we specify the date with 'since: until:'. Figure 1 shows an example of the keyword query.

```
# Using TwitterSearchScraper to scrape data and append tweets to list
for i,tweet in enumerate(sntwitter.TwitterSearchScraper('("USA" OR "the US" OR "United States of America" OR "America") \
(#gunlaws OR "gun control" OR "gun laws" OR "gun violence") \
-is:retweet lang:en since:%s until:%s' %(start,end)).get_items()):
```

Figure 1: Crawling Twitter with Keywords

The tweets are retrieved as a dataframe and exported as a CSV file. An example of a CSV file is shown in Figure 2.

1	Datetime	TweetId	Text	RetweetC	LikeCount	Username
2	2020-12-3	1.34E+18	â€œYou can Dixie Chick us anytime you want.â€™ â€œ Musical duo John and TJ Osborne have becom	38	217	nowthisnews
3	2020-12-3	1.34E+18	Fuck America. Fuck the police. Fuck the system. POLICE = GUN VIOLENCE	2	8	fayomar82
4	2020-12-3	1.34E+18	@NigelThorndaddy @AdmiralSpeedy @mariem201 @TorontoStar Illinois is among the states with	0	1	djjsjeisisi
5	2020-12-3	1.34E+18	America: why are we so bad at trains and gun control? Everyone else seems to do those two things	1	5	adamstemple4
6	2020-12-3	1.34E+18	@NBCNews get ur shyt together america . u all kill each other more than any international war eve	0	1	SaysSheryl
7	2020-12-3	1.34E+18	@ILikeGearsOWar1 @Breaking911 All you're doing is providing distracting facts from the real	0	0	deceptive_donny
8	2020-12-3	1.34E+18	HEREâ€™S WHAT OBAMA WAS SETTING UP FOR HILARY CLINTON, NOW JOE BIDEN WILL IMPLEMENT	0	0	LeoRed74084809
9	2020-12-3	1.34E+18	@tigerguy529 @shoe0nhead @PUNISHEDASH Yes, because privacy has direct mental implications.	0	0	_KeanuReaves
10	2020-12-3	1.34E+18	@a_paradisefound @KaMist72 @ossoff Not all of the US, but much. The NRA has had unlimited fu	0	1	HadleyTa
11	2020-12-3	1.34E+18	@RepMaryFranson @GovRonDeSantis @j_w_84 Serious message, Mary. I've looked at your positio	0	7	annbauerwriter
12	2020-12-3	1.34E+18	The number of mass shootings in America fell almost by half between 2019 and 2020 even as gun v	0	0	WKOW
13	2020-12-3	1.34E+18	@kingcowpoke @ColumbiaBugle nonsense, he managed to pardon convicted spy for the greatest i	0	1	MrTianfang

Figure 2: Crawled Tweets in CSV Format

Crawling Reddit

The data is crawled from an archiving database, Pushshift, that contains both real-time and historical Reddit posts. The API 'https://api.pushshift.io/reddit/search/comment' was used to directly access the database.

The keywords used for crawling Reddit are similar to the ones used for crawling Twitter. Figure 3 shows an example of the keyword query.

```
# Take 250 comments from certain ranges from the past 2-3 years
days = [i for i in range(10,810,10)]
SIZE = 250
keywords = "('gun USA law'|'gun America law'|'gun USA violence'|'gun America violence')"
```

```
URI_TEMPLATE = r'https://api.pushshift.io/reddit/search/comment/?after={}&d&limit={}&q={}&sort=asc'
```

```
res = []
for day in days:
    print(day, end=' ')
    posts = map_posts(requests.get(URI_TEMPLATE.format(day, SIZE, keywords)).json()['data'])
    res.extend(posts)
```

Figure 3: Crawling Reddit with Keywords

The Reddit posts are retrieved as a dataframe and exported as a CSV file. An example of a CSV file is shown in Figure 4.

	user	timestamp	text
0	serenity_I	1661790456	Yeah we have a gun problem but to insinuate that America is the only place where you will come across stupidity or violence is wrong. That's the point that I am making.
1	TaliesinW	1661791729	>What you said is simply something that gets made fun of constantly.I have dealt with people from multiple European countries on a personal and professional level
2	ART3MIS1	1661792591	America! Diabetes for all! The land of freedom? Really? More like the land gun violence, taxes and police violence.
3	phantomt	1661792712	From his [website](https://betoorourke.com/issue/promote-gun-safety/-) a Closing the private sale background check loophole may have prevented the 2019 Midland-
4	Shallow-T	1661792891	America is a unique blend of first world and third world countries. Economic problems are a good way to foster violence. And we have plenty of poor regions where tha
5	panicattac	1661794746	I find it bewildering that people thought this movie was going to inspire mass shooters.Mass shooters donâ€™t need inspiration, and this scene wouldnâ€™t exist if we c
6	ryhalswh	1661794832	Has anyone ever studied the impact on tourism from the gun violence problem in America?
7	heyzeus2	1661795134	When people from other countries ask why we have so much gun violence in America, they should be shown this, with the explanation that this person could buy weapo
8	avonarret	1661796744	Oh, you really can't be bothered by dead people getting on your nerves anymore, right? So annoying. And it's aaaaalways the same "yada yada america bad". How old are
9	Unusual-V	1661796751	""What the American left needs now is allegiance, not allyship. It must abandon any imagined fantasies about the sanctity of governmental institutions that long ago gav
10	mormagil	1661797955	Right, I'm not disagreeing with you that these kinds of laws are unpopular, but the point I'm making is that if that's the cause...then that's the cause. You can't say "what a
11	supamaric	1661798443	So long as America shows an indifference towards certain discriminated groups, an inability to address the issue of politically motivated gun violence, and an unwilling
12	megor	1661799352	I don't think giving everyone fully automatic weapons would be a great solution to gun violence in America.

Figure 4: Crawled Reddit posts in CSV Format

2.2 Insights Derived

The text corpus contains public opinion on gun laws. From this data, we can get insights into the reasons for someone being a pro-gun or an anti-gun advocate. For example, anti-gun advocates often cite mass shootings and gun violence statistics when debating against pro-gun proponents. Whereas, pro-gun proponents often argued about the rights of self-defence.

1. **Pro-gun:** favouring the freedom of gun ownership and against gun control. For example, "The only thing that stops a bad guy with a gun, is a good guy with a gun" or "Gun control isn't gonna help. God bless the 2nd amendment! #sorrynotsorry".
2. **Anti-gun:** favouring gun control and express outrage and sympathy to the event. For example, "I cannot fathom how one could be so vile that they open fire on children or anyone. We need STRICT gun controls. #Newtown" or "BAN GUNS!!! Let our children be safe".

3. **Neutral:** irrelevant in the context or neither convey a progun or anti-gun sentiment. For example, “Obama is possibly visiting Newtown” or “Not sure if gunshot or firework”.

2.3 Text Corpus Information

We will merge both the crawled data from Twitter and Reddit. Overall, our combined dataset has 27,860 rows. As we are focussing on the content of the tweets and Reddit posts, the dataset is stored as a csv file with only the column ‘Text’. Table 1 is an example of the ‘Text’ column.

Field	Description	Example Content
Text	Text content of the tweet or Reddit post.	Never going to understand the gun laws in the US, one shooting in Scotland and guns were banned, never happened again yet hundreds of people still die from gun violence and they do nothing??

Table 1: Example Content

Each row in the CSV file corresponds to a tweet or Reddit post. There are a total of 5016138 word tokens crawled for the “Text” column. Among those, there are 187630 unique words. After removing stop words and either using stemming or lemmatization, 2613056 word tokens are left.

Figure 5 shows the word count information and Figure 6 shows the unique word information.

```
print("Text Corpus Information for ALL the data:")
print()
wordcount = int(df['Original_Text'].str.split().str.len().sum())
print("There are {} words in the corpus.".format(wordcount))

wordcount = int(df['Cleaned_Text'].str.split().str.len().sum())
print("There are {} words in the corpus after cleaning.".format(wordcount))

wordcount = int(df['Stemmed_Text'].str.split().str.len().sum())
print("There are {} words in the corpus after stemming and removal of stopwords.".format(wordcount))

wordcount = int(df['Lemmatized_Text'].str.split().str.len().sum())
print("There are {} words in the corpus after lemmatization and removal of stopwords.".format(wordcount))
```

Text Corpus Information for ALL the data:

There are 5016138 words in the corpus.
There are 4965085 words in the corpus after cleaning.
There are 2613056 words in the corpus after stemming and removal of stopwords.
There are 2613056 words in the corpus after lemmatization and removal of stopwords.

Figure 5: Word Count Information for Combined Data

```

print("Unique Words Information for ALL the data:")
print()

unique = df.Original_Text.str.split(expand=True).stack().value_counts()
print("There are {} Unique words in the corpus.".format(len(unique)))

unique = df.Cleaned_Text.str.split(expand=True).stack().value_counts()
print("There are {} Unique words in the corpus after cleaning.".format(len(unique)))

unique = df.Stemmed_Text.str.split(expand=True).stack().value_counts()
print("There are {} Unique words in the corpus after stemming and removal of stopwords.".format(len(unique)))

unique = df.Lemmatized_Text.str.split(expand=True).stack().value_counts()
print("There are {} Unique words in the corpus after lemmatization and removal of stopwords.".format(len(unique)))

```

Unique Words Information for ALL the data:

There are 187630 Unique words in the corpus.
There are 65932 Unique words in the corpus after cleaning.
There are 46969 Unique words in the corpus after stemming and removal of stopwords.
There are 59721 Unique words in the corpus after lemmatization and removal of stopwords.

Figure 6: Unique Word for Combined Data

3 Data Preparation

3.1 Pre-processing of Text Corpus

Preprocessing had to be done on the “Text” column as both tweets and Reddit post are “noisy”. Tweets often include links, hashtags and mentions whereas Reddit posts contain HTML special characters that need to be decoded.

3.1.1 Merging Twitter and Reddit CSVs

Since we crawled from both Twitter and Reddit, we need to merge the 2 CSV files together to obtain a combined dataset. As we only require the text content from both the tweets and Reddit posts, we will just merge the common “Text” column.

3.1.2 Data Preprocessing

Data Cleaning

The steps used for cleaning the “Text” column are as follows:

1. Characters are converted to lowercase
2. URLs, hashtags and mentions are removed
3. All the punctuations and non-alphanumeric characters are removed
4. Decoded HTML special characters

Figure 7 shows the code snippet used for cleaning the “Text” column.


```

def clean_text(text):
    text = re.sub("@[A-Za-z0-9_]+", "", text)
    text = re.sub("#[A-Za-z0-9_]+", "", text)
    text = re.sub(r'[^a-zA-Z ]+', ' ', text)
    text = re.sub(r'http\S+', ' ', text)
    text = re.sub(r'https?:\/\/\/*[\r\n]*', '', text)
    return text

def remove_html_punc(text):
    text = html.unescape(text)
    text = text.translate(str.maketrans('', '', string.punctuation))
    text = ' '.join([word for word in text.split()])
    text = text.lower()
    return text

```

Figure 7: Code Snippet for Data Cleaning

3.1.3 Stemming, Lemmatization and Removal of Stopwords

The text normalization methods we used are stemming and lemmatization. We used PorterStemmer for stemming and Wordnet lemmatizer from NLTK to lemmatize the words. Stopwords are also removed using NLTK. Figure 8 shows an example of the 2 methods.

1	Original_Text	Cleaned_Text	Lemmatized_Text	Stemmed_Text	Label
	<p>@GunDeaths @KingJames @POTUS @VP @AttorneyCrump Gun Violence is out of control in America. Courts need to give automatic sentences of life without parole to anyone who commits a crime using a gun.</p> <p>It is obvious that BLM does nothing to help stop the gun violence on the streets every night</p>	<p>gun violence is out of control in america courts need to give automatic sentences of life without parole to anyone who commits a crime using a gun it is obvious that blm does nothing to help stop the gun violence on the streets every night</p>	<p>gun violence control america court need give automatic sentence life without parole anyone commits crime using gun obvious blm nothing help stop gun violence street every night</p>	<p>gun violenc control america court need give automat sentenc life without parol anyon commit crime use gun obviou blm noth help stop gun violenc street everi night</p>	0

Figure 8: Stemmed and Lemmatized Text

3.1.4 Word Count

The word count for both the labelled and unlabelled data is as follows:

```
Text Corpus Information for Labelled data:

There are 244375 words in the corpus.
There are 242243 words in the corpus after cleaning.
There are 126856 words in the corpus after stemming and removal of stopwords.
There are 126856 words in the corpus after lemmatization and removal of stopwords.
```

Figure 9: Text Corpus Information for Labelled Data

```
Text Corpus Information for the Non-labelled data:

There are 4771763 words in the corpus.
There are 4722842 words in the corpus after cleaning.
There are 2486200 words in the corpus after stemming and removal of stopwords.
There are 2486200 words in the corpus after lemmatization and removal of stopwords.
```

Figure 10: Text Corpus Information for Unlabelled Data

3.2 Manual Labeling of Collected Data

The team has manually labeled a total of 1,214 rows of data out of 27,860 rows of collected data. The inter-annotator agreement between the 2 raters is 0.949.

The labeled classes are as follows:

- 0 = Neutral
- 1 = Anti Gun (Guns should be banned)
- -1 = Pro Gun (In favor of keeping guns)

```
labelled2raters_df = pd.read_csv('/content/drive/MyDrive/Train/Rater/labelled_rater2.csv')

rater_1 = labelled2raters_df['Rater 1'].to_numpy()
rater_2 = labelled2raters_df['Rater 2'].to_numpy()

from sklearn.metrics import cohen_kappa_score
kappa_boi = cohen_kappa_score(rater_1, rater_2)
print("The inter-annotator score between 2 raters is {}".format(kappa_boi))
```

The inter-annotator score between 2 raters is 0.9493753299680935

Figure 11: Inner-annotator score of Labeled Data

4 Classification

4.1 Classification approach

The data has been divided between 80% training courses and 20% testing classes after being manually labeled and preprocessed. In addition, we have selected the subsequent classification models. To get the optimum evaluation score for processing, we took into account various preprocessing techniques for each model, specifically stemming and lemmatization.

4.1.1 K-Nearest Neighbors (KNN)

We first used Scikit-learn's kNN implementation to train and test the k nearest neighbor classifier on our labeled dataset for both subjectivity and polarity. kNN works by calculating the distance between the query and the database and within a threshold of distance, whichever class has more points that are close to the query will classify the query to its class. For the classifier, we feed a list of hyperparameters, namely metric, leaf size, neighbor size and weight into grid search cv which will give us the optimal combination of hyperparameters. Then we take advantage of the results and use it as the kNN model. For vectorizer, we tried both count vectorizer and TF-IDF vectorizer. It turned out that TF-IDF vectorizer overall performs better between the two in almost all of the training results. We also use 5-fold cross validation to avoid a biased result and the accuracy is around 70%. The confusion matrix is shown below.

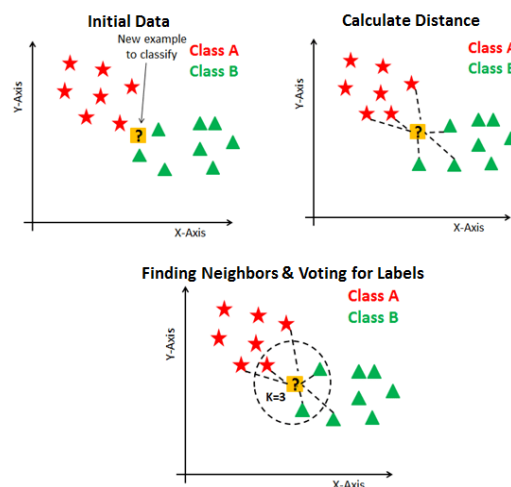


Figure 12: Demonstration of how KNN works for a binary classification in sentiment analysis

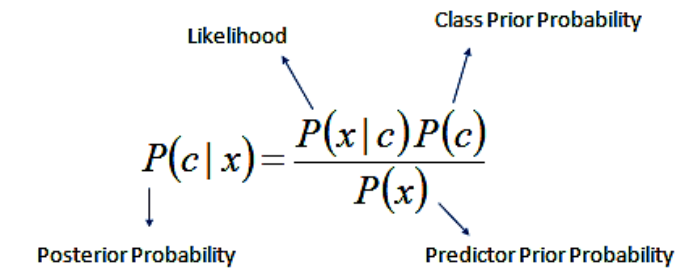
4.1.2 Random Forest (RF)

We also used Random Forest on the labeled dataset. Similar to kNN implementation, we listed down various hyperparameters for grid search to tune with so that we can achieve a better accuracy. The hyperparameters include number of trees in random forests (`n_estimators`), number of features to consider at every split (`max_features`), minimum number of samples

required to split a node (min_samples_split), and minimum number of samples required at each leaf node (min_samples_leaf). For Random Forest, we also tested count vectorizer as well as TF-IDF and TF-IDF still gave a higher result of accuracy for both subjectivity detection and polarity detection, with around 70% of accuracy.

4.1.3 Naive Bayes

The Naive Bayes classifier makes the assumption that a certain feature's presence in a class has no bearing on the presence of any other feature. Naive Bayes offers a method for estimating the posterior probability $P(y | x)$ of each class y , given an object x , utilizing the information in sample data. When we obtain these estimations, we can use them for categorization or other decision-supporting applications.



The diagram shows the equation for the Naive Bayes classifier: $P(c | x) = \frac{P(x | c)P(c)}{P(x)}$. Arrows point from labels to the terms in the equation: 'Likelihood' points to $P(x | c)$, 'Class Prior Probability' points to $P(c)$, 'Posterior Probability' points to $P(c | x)$, and 'Predictor Prior Probability' points to $P(x)$.

$$P(c | X) = P(x_1 | c) \times P(x_2 | c) \times \dots \times P(x_n | c) \times P(c)$$

Figure 13: Equation for Naive Bayes Classifier

Complement Naive Bayes is an adaptation of the standard Multinomial Naive Bayes algorithm that works well with imbalanced datasets. Instead of calculating the probability of an item belonging to a specific class, we calculate the probability of the item belonging to all classes in complement Naive Bayes.

In this case, we used Scikit-Naive learn's Bayes implementation, specifically MultinomialNB() and ComplementNB(), as well as the count vectorizer and TF-IDF vectorizer, and concluded that the results for both the vectorizers are almost equal, about 70% for both subjectivity and polarity detection.

4.1.4 Doc2Vec

In the previous methods, we used TF-IDF and Count Vectorizer as word embedding techniques to convert the text into input vectors for model consumption. In the section below, we explore using Doc2Vec as an alternative method to TF-IDF.

Doc2Vec is based on Word2Vec, which is based on the Continuous Bag-of-Words model and the Skip-Gram model. Word2Vec aims to generate representation vectors out of words such that it can encapsulate different relations between words. A classic example is shown below:

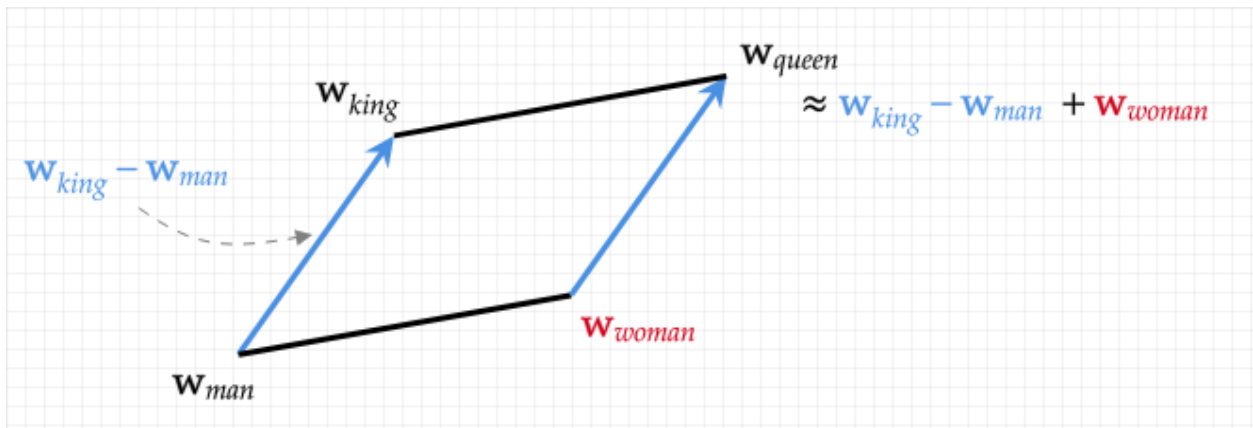


Figure 14: Diagram showcasing the relationships between word vectors

The example above illustrates the equation of “king - man + woman = queen”, and demonstrates how the mathematical closeness of the representative vector values are similar to the semantic closeness between the words. It can be understood that the words “king” and “queen” are very similar to each other, but vectorial differences arise only because of their gender.

The goal of Doc2Vec is to create a numeric representation of a document, regardless of its length. Building on the Word2Vec model, instead of using just words to predict the next word, another feature vector, which is document-unique. When training the word vectors, the document vector is trained as well, and at the end of training, it holds a numeric representation of the document.

In the context of this project, we used the **gensim** library to create Doc2Vec models that are trained on the different levels of preprocessing used (Cleaned_Text, Lemmatized_Text and Stemmed_Text). For each type of text, each sample was then converted into a vector of length 40 which represented that particular document. After which, these vectors are used as inputs to several supervised classification models (e.g. Logistic Regression, Decision Tree, Random Forest and Support Vector Machine Classifier) and trained to predict the subjectivity and polarity of each sample.

Overall, the performance of using Doc2Vec as a word representation technique gave similar results to previous sections and the average accuracy for most models hovered around 70%. It also faced the same issue of a very high False Positive Rate for both Subjectivity and Polarity detection, which might be due to some imbalance of classes in the training dataset.

4.2 Evaluation Metrics

The evaluation metrics used are the F1 Score, Precision score as well as Recall Score for each preprocessing data of Stemming and Lemmatization under the Count Vectorizer, TF-IDF Vectorizer and Doc2Vec.

4.2.1 Overall Result

Subjectivity Detection

Model	Text Normalization	Precision Score	Recall Score	F1 Score
KNN	Stemming	0.728	0.953	0.825
	Lemmatization	0.728	0.953	0.825
RF	Stemming	0.707	0.994	0.822
	Lemmatization	0.703	0.988	0.823
Naive Bayes	Stemming	0.704	1	0.826
	Lemmatization	0.704	1	0.826
Complement Naive Bayes	Stemming	0.704	1	0.826
	Lemmatization	0.707	1	0.826
Doc2Vec (Logistic Regression)	Stemming	0.704	1	0.826
	Lemmatization	0.704	1	0.826
Doc2Vec (Decision Tree)	Stemming	0.726	0.713	0.701
	Lemmatization	0.696	0.714	0.701
Doc2Vec (Random Forest)	Stemming	0.706	0.930	0.803
	Lemmatization	0.720	0.953	0.817

Doc2Vec (SVM)	Stemming	0.704	1	0.826
	Lemmatization	0.704	1	0.826

Table 2: Overall Subjectivity Detection Results

Polarity Detection

Model	Text Normalization	Precision Score	Recall Score	F1 Score
KNN	Stemming	0.712	0.945	0.812
	Lemmatization	0.715	0.955	0.817
RF	Stemming	0.682	0.936	0.802
	Lemmatization	0.687	0.955	0.796
Naive Bayes	Stemming	0.689	0.927	0.79
	Lemmatization	0.714	0.927	0.806
Complement Naive Bayes	Stemming	0.688	0.9	0.779
	Lemmatization	0.712	0.918	0.802
Doc2Vec (Logistic Regression)	Stemming	0.667	1	0.800
	Lemmatization	0.667	1	0.800
Doc2Vec (Decision Tree)	Stemming	0.715	0.627	0.672
	Lemmatization	0.671	0.718	0.696

Doc2Vec (Random Forest)	Stemming	0.721	0.864	0.794
	Lemmatization	0.665	0.773	0.735
Doc2Vec (SVM)	Stemming	0.685	0.964	0.800
	Lemmatization	0.667	1	0.800

Table 3: Overall Polarity Detection Results

4.3 Performance Metrics

4.3.1 Time taken per Classification Model

We obtain the following by calculating the timing of each classification model and vectorizer methods.

Model	Text Normalization	Time for Subjectivity (seconds)	Time for Polarity (seconds)
KNN	Stemming	17.3	6.44
	Lemmatization	20.05	6.31
RF	Stemming	42.45	14.80
	Lemmatization	96.43	16.78
Naive Bayes	Stemming	3.74	2.44
	Lemmatization	5.14	1.66
Complement Naive Bayes	Stemming	3.78	2.43
	Lemmatization	5.88	1.68

Doc2Vec (Logistic Regression)	Stemming	0.44	0.31
	Lemmatization	0.31	0.29
Doc2Vec (Decision Tree)	Stemming	0.31	0.26
	Lemmatization	0.29	0.26
Doc2Vec (Random Forest)	Stemming	3.10	2.69
	Lemmatization	3.11	2.72
Doc2Vec (SVM)	Stemming	0.33	0.25
	Lemmatization	0.33	0.27

Table 4: Timings for each Classification Model

5 Innovations for Enhancing Classification

5.1 Ensemble Learning via Model Stacking

Separation of models used during classification leads to limitations in the areas of performance and robustness. Ensemble learning is the most effective method to enhance accuracy. A better predictive performance is achieved as it reduces the variance component of the prediction error. 3 main issues which standard learning algorithms face are alleviated. These issues are:

Issue	Root cause
Statistical	Algorithms suffer from high variance
Computation	Algorithms suffer from computational variance
Representation	Algorithms exhibit high bias

Table 5: Issues of standard learning algorithms and their root causes

High variance is detrimental to prediction as it is too focused on training data, leading to overfitting and high levels of inaccuracies on test data. High bias, on the contrary, is the opposite. It focuses too little on the training data, leading to oversimplification of the algorithm

and high levels of inaccuracies on both the training and test data.

Ensemble learning is able to reduce both the bias and variance of learning algorithms, making it an optimal solution. In fact, it is widely used in real life across different industries as it ensures output is not determined by a single model. In other words, it prevents systems from placing all their eggs into one basket. For this reason, ensemble learning is applied in speech emotion recognition in multilingual environments. It has been found to improve accuracy of results without compromising language corpus' accuracy. Ensemble learning has also been applied in the medical industry for illness detection, such as cardiovascular disease detection from X-Ray and CT scans. Lastly, it has been used to classify scenes and monitor physical aspects of an area via data gathered from different sensors. It is particularly useful in remote sensing as the data from the sensors are of different degrees of resolution and leads to incoherence in data distribution. This list is non-exhaustive and there are certainly other applications of ensemble learning as well.

We have implemented ensemble learning via the use of model stacking.

Model stacking consists of 3 steps:

1. Run a dataset on multiple base machine learning algorithms on a single dataset. We have chosen gradient boosting, decision tree, random forest and linear SVC as the base learning algorithms.
2. These outputs are then used in the second layer machine learning algorithm, also known as a meta-level classifier, as inputs. The inputs are in the form of pairs of feature vectors and their respective base-level classifiers. We have chosen logistic regression as our meta-level classifier.
3. The meta-level classifier learns the best way to combine the input predictions to output an improved prediction.

A pictorial representation of model stacking is shown below:

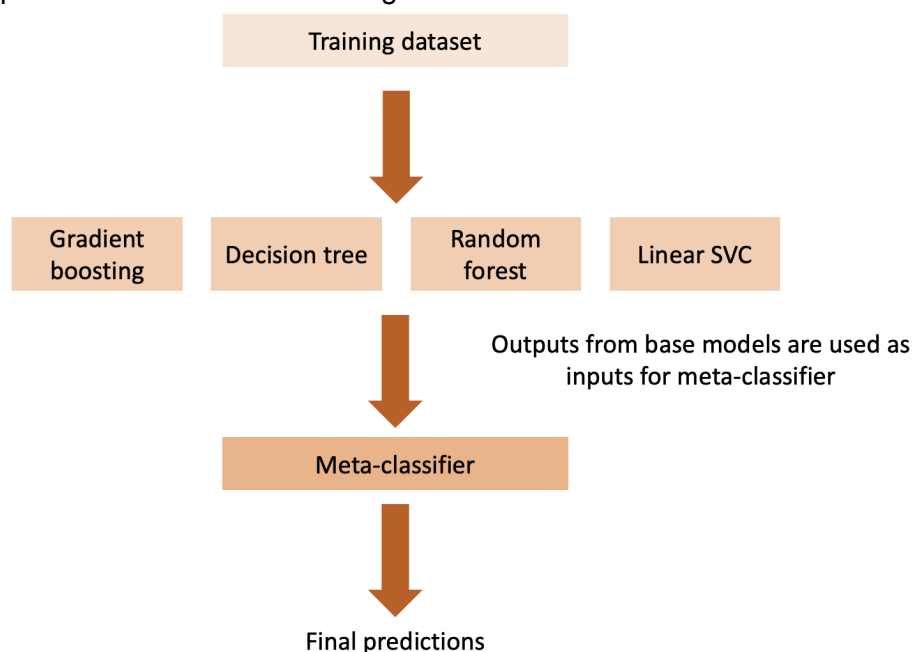


Figure 15: Model Stacking

Model stacking takes into account heterogeneous weak learners, learns them simultaneously and subsequently combines several of these learners by training a meta-learner to output predictions based on the weak learners' predictions.

The results are as follows:

Subjectivity	Precision Score	Recall Score	F1 Score	Accuracy
0	0.63	0.25	0.36	0.72
1	0.73	0.93	0.82	
Weighted average	0.70	0.72	0.68	

Table 6: Stacking Subjectivity Detection Results

Polarity	Precision Score	Recall Score	F1 Score	Accuracy
0	0.69	0.60	0.64	0.75
1	0.78	0.84	0.81	
Weighted average	0.75	0.75	0.75	

Table 7: Stacking Polarity Detection Results

Combined, both models have an average accuracy of 0.735, outperforming the rest of the learning algorithms, proving that it enhances the accuracy of our classification models.

5.2 LSTM and Bi-Directional LSTM

5.2.1 Motivation

The previous models fail when we need to remember context in a long sentence.

For the given dataset, it is important to understand the context before concluding whether the given data is in favor of keeping guns or not. Let us look at an example:

E.g. "If the people who love trying to make Chicago the poster child for gun violence only zoomed out and saw how the US is viewed that way by the rest of the world. Perhaps there would be a better understanding of the larger issue"

In the above example, there is a shift of subjects between the sentences and you need to remember what was being discussed in the earlier sentence to understand what the “larger issue” refers to. It is thus essential that we train our model to remember the historical context. This is achieved by using a LSTM which has a memory unit to remember long contexts.

5.2.2 Long Short Term Memory

RNNs are a type of feedforward neural network with a hidden recurrent state that is periodically activated by the active states. RNNs are able to handle variable-length sequences and dynamically model contextual information. However, RNNs face the long term dependency issue, which means by the time they reach the end of a given sentence their gradient become so small that no results can be derived from it. RNNs thus cannot help us to remember long contexts.

LSTM is a type of RNN architecture that is currently the standard structure for RNNs. By using memory blocks in place of the self-connected hidden units, LSTM solves the vanishing gradient issue. The memory block is better at identifying and utilizing long range context because it uses purpose-built memory cells to store information. Using the memory units, the network can be aware of when to learn new knowledge and when to forget outdated information. A LSTM unit is made up of these four parts:

1. The i is an input gate and it controls the size of the new memory content added to the memory.
2. The f is a forget gate and it determines the amount of the memory that needs to be forgotten.
3. The o is an output gate and it modulates the amount of the output memory content.
4. The c is the cell activation vector and it consists of two components, namely partially forgotten previous memory c_{t-1} and modulated new memory c_t . t nominates the t^{th} moment.

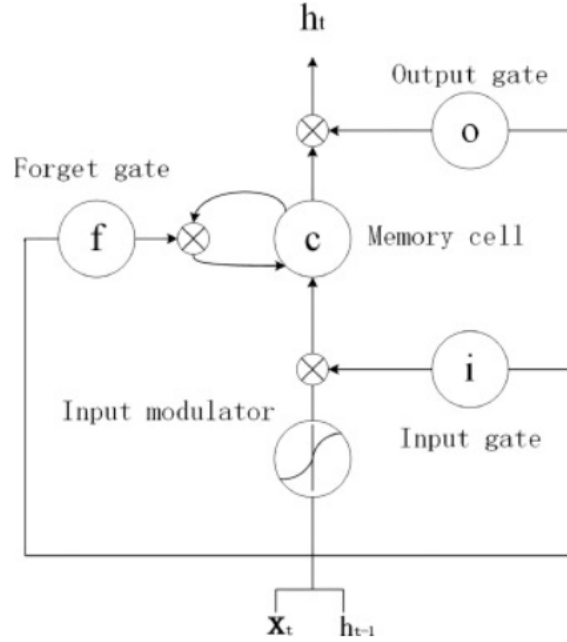


Figure 16: Illustration of LSTM unit. The weight matrices are represented by lines with arrows.

5.2.3 Bi-Directional Long Short-Term Memory

Conventional LSTM networks can only utilize historical context. However, a lack of future context could result in a misunderstanding of the problem's significance. As a result, it is Bi-LSTMs are introduced to access both the preceding and following contexts by combining a forward hidden layer with a backward hidden layer, as shown in Fig. 2's (b) illustration. Similar to conventional network forward and backward passes, the forward and backward passes across the unfolded network over time are carried out, with the exception that BiLSTM must unfold the forward hidden states and the backward hidden states for each time step. In order to train the BiLSTM networks, backpropagation over time is used (BPTT).

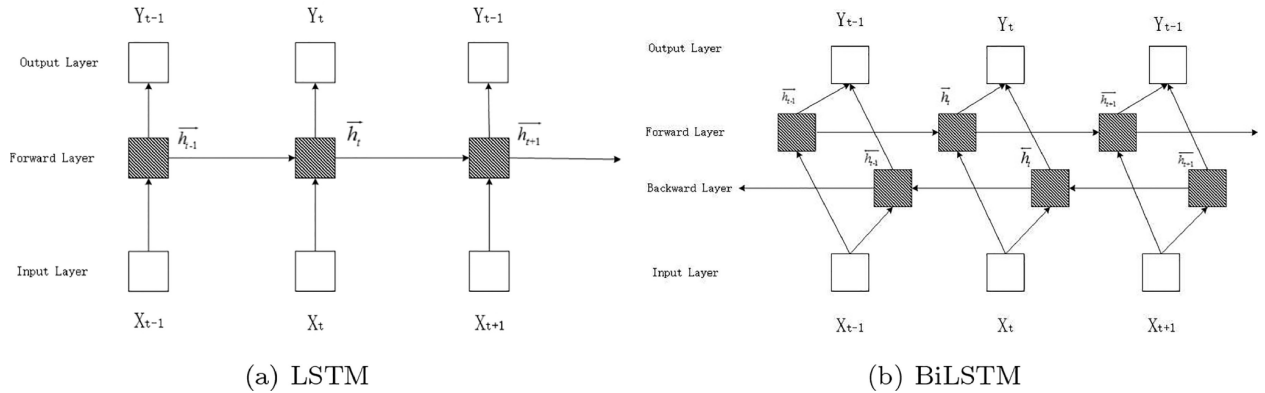


Figure 17: Illustration of a LSTM model (a) and a BiLSTM model (b).

5.2.4 Architecture

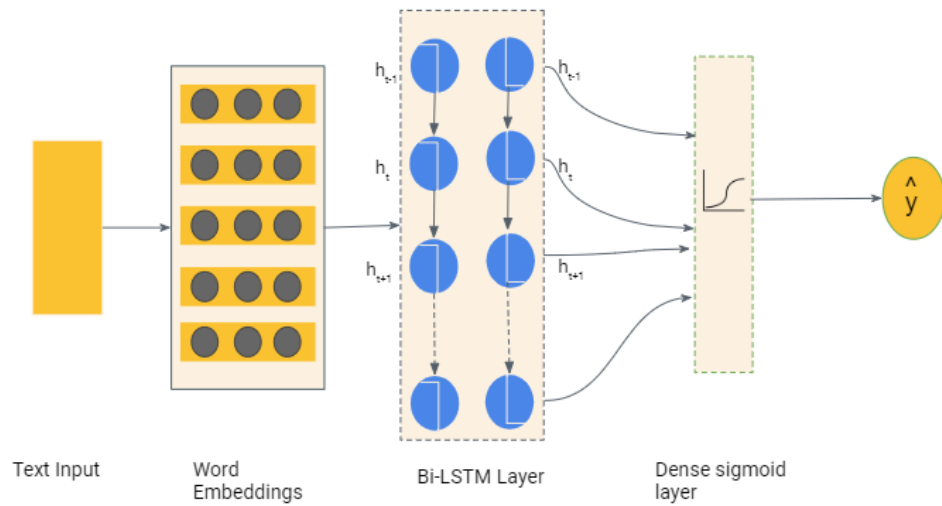


Figure 18: Architecture of Bi-LSTM Neural Network

5.2.5 Results

	F1	Precision	Recall
Polarity	0.7605	0.6747	0.8756
Subjectivity	0.8210	0.7065	0.9848

Table 8: Bi-LSTM Results

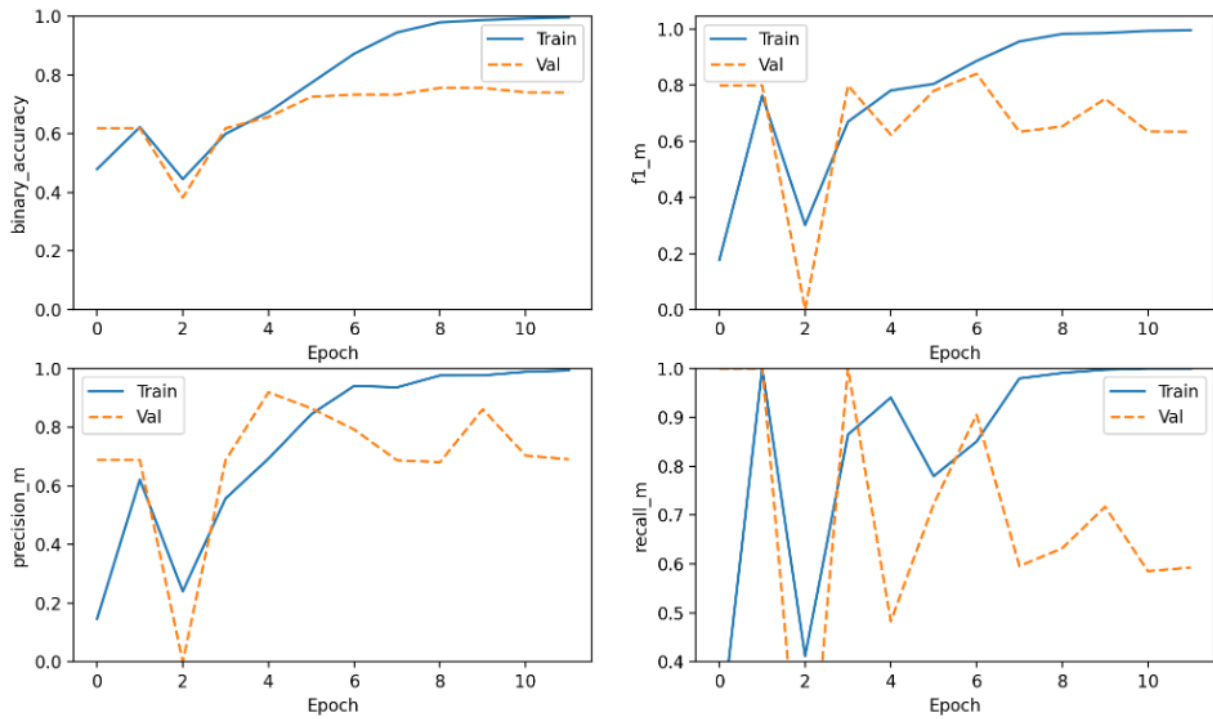


Figure 19: Overfitting is observed after very few epochs

5.3 Attention-based Bi-Directional LSTM

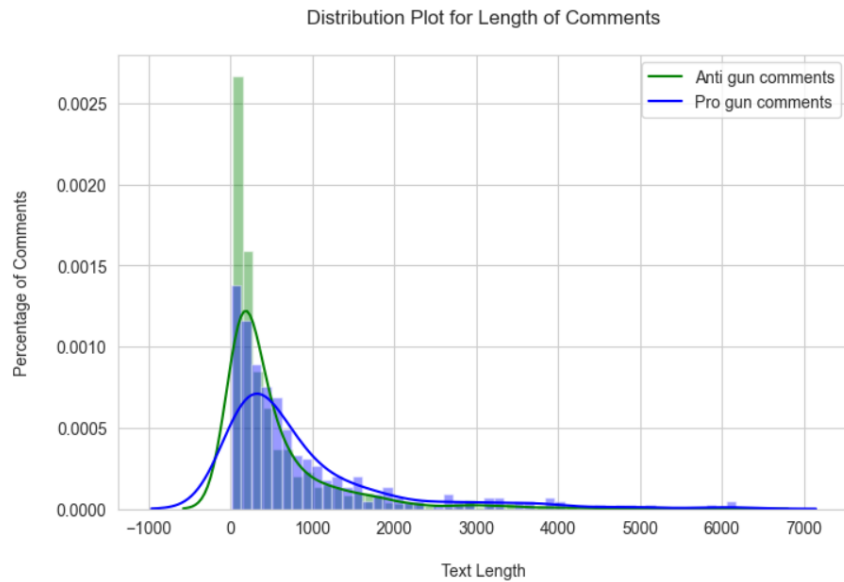


Figure 20: Distribution of Text Length

It was observed that the length of tweets in our dataset is very long. The average length of one tweet is 638.8 words, while maximum length is 4000 words. There was thus a need for introducing an attention layer which can help the model learn the few important parts which might be ignored previously.

Now the encoder which we are using in the network is a bidirectional LSTM network where it has a forward hidden state and a backward hidden state. Representation of the encoder state can be done by concatenation of these forward and backward states.

$$\mathbf{h}_i = [\vec{\mathbf{h}}_i; \overleftarrow{\mathbf{h}}_i]^\top, i = 1, \dots, n$$

Where in the decoder network, the hidden state is

$$\mathbf{s}_t = f(\mathbf{s}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}_t)$$

For the output word at position t, the context vector \mathbf{c}_t can be the sum of the hidden states of the input sequence.

$$\mathbf{c}_t = \sum_{i=1}^n \alpha_{t,i} \mathbf{h}_i \quad ; \text{ Context vector for output } y_t$$

$$\alpha_{t,i} = \text{align}(\mathbf{y}_t, \mathbf{x}_i) \quad ; \text{ How well two words } y_t \text{ and } x_i \text{ are aligned.}$$

$$= \frac{\exp(\text{score}(\mathbf{s}_{t-1}, \mathbf{h}_i))}{\sum_{i'=1}^n \exp(\text{score}(\mathbf{s}_{t-1}, \mathbf{h}_{i'}))} \quad ; \text{ Softmax of some predefined alignment score..}$$

There are multiple predefined alignment functions available, with additive and scaled dot products being the most commonly used ones. Since the dimensions of our data are small, both Scaled Dot-Product Attention and additive attention would perform similarly. We chose Scaled Dot-Product Attention. The alignment score function of Scaled Dot-Product Attention is given by,

$$\text{score}(\mathbf{s}_t, \mathbf{h}_i) = \frac{\mathbf{s}_t^\top \mathbf{h}_i}{\sqrt{n}}$$

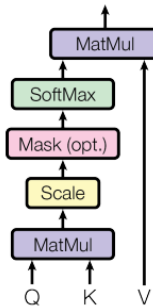


Figure 21: Architecture of Scaled Dot-Product Attention

5.3.1 Architecture

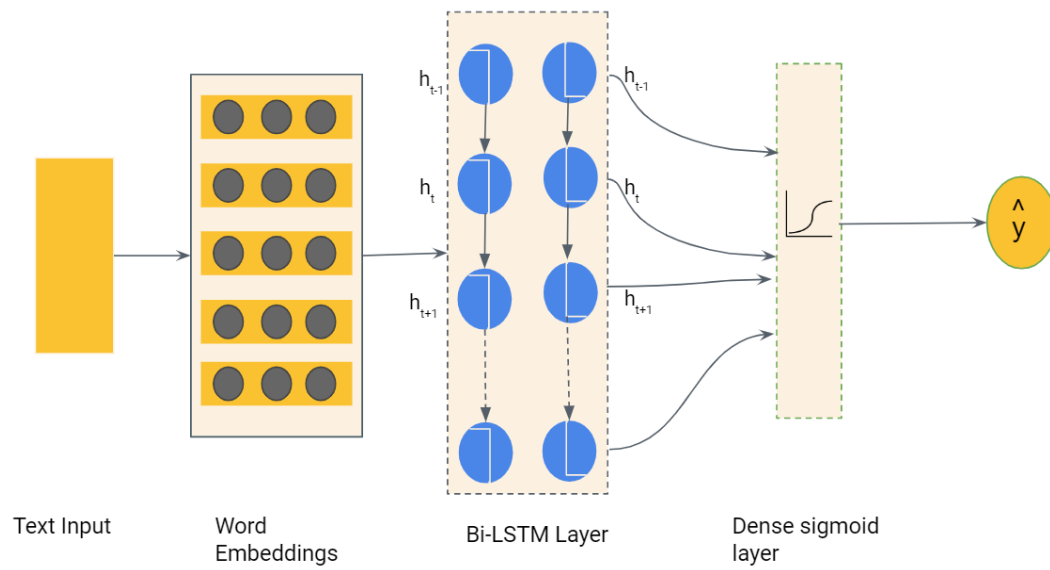


Figure 22: Architecture of the attention-based LSTM model

5.3.2 Results

	F1	Precision	Recall	Binary Accuracy
Polarity	0.8255	0.7131	0.9936	0.7030
Subjectivity	0.7299	0.6431	0.8507	0.6689

Table 9: attention-based LSTM Results

Since, our data includes some sentences with > 2000 words, using multi head attention with a large number of heads might improve the performance as compared to scaled dot product attention. Multi-head attention will allow the model to jointly attend to information from different representation subspaces at different positions, thus improving the performance of the model on long sentences.

5.4 User Interface (UI)

Our team has deployed a UI to display our classification results using Gradio, a python library. It is integrated locally in our python notebook, “UI.ipynb” (located at: Submission_2-Group16.zip → UI Folder)

As shown in Figure 23, this sentiment analysis UI takes in the input text and returns its classification for either pro-gun (0) or anti-gun (1) sentiment as output.

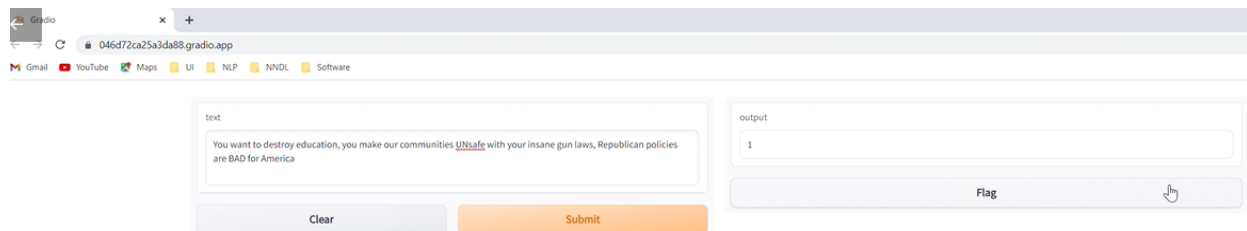


Figure 23: Classification UI

6 Conclusion

The best model was used to predict the label of the remaining 90% of the data crawled in Section 2. The data covers the period from January 2020 - August 2022. The observations from the data are as follows:

1. Overall, the number of anti gun tweets have remained significantly larger than the pro gun tweets between 2020 and 2022.

2. A few dips in the number of anti gun tweets and spikes in pro gun tweets are observed in the beginning of 2020. This possible as a result of the gun control passed by the USA Supreme Court in early 2020
3. The total number of tweets increased in 2022. Further, the number of anti gun tweets achieved a global maxima around mid 2022, shortly after the shooting incident at an elementary school in Texas which killed 19 children. This is also followed by a spike in pro gun tweets, indicating the retaliation to the gun control bill passed in June 2022.
4. Fig. 25 shows that the increase in the number of anti gun expressions have increased linearly from 2020 to 2022. Using this information, we can expect the trend to continue for the next three years or till the next government takes charge (whichever earlier).

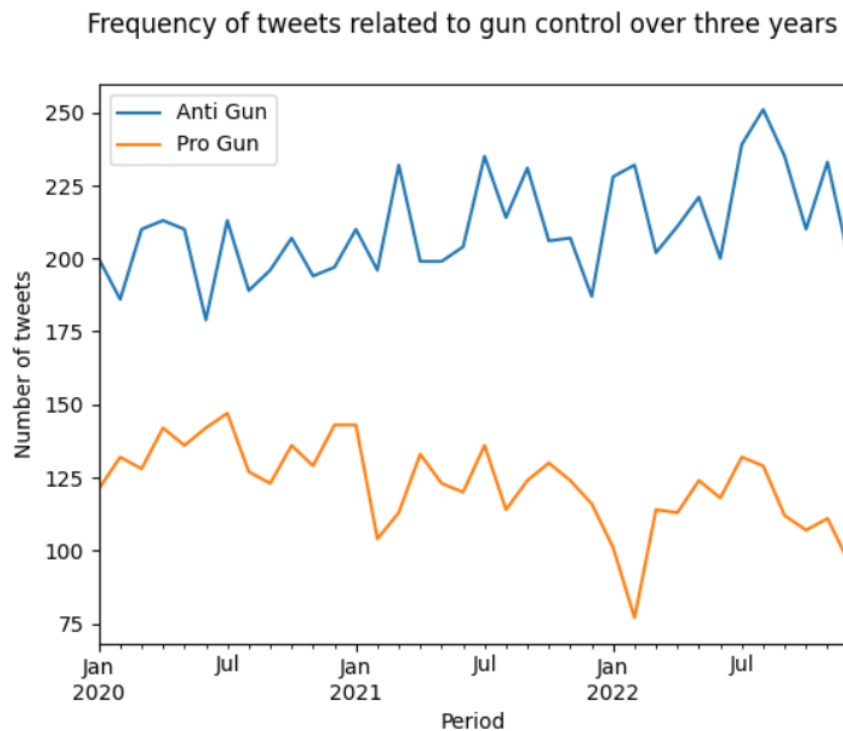


Figure 24: Time series showing the number of pro and anti gun tweets/reddits from 2020 - 2022

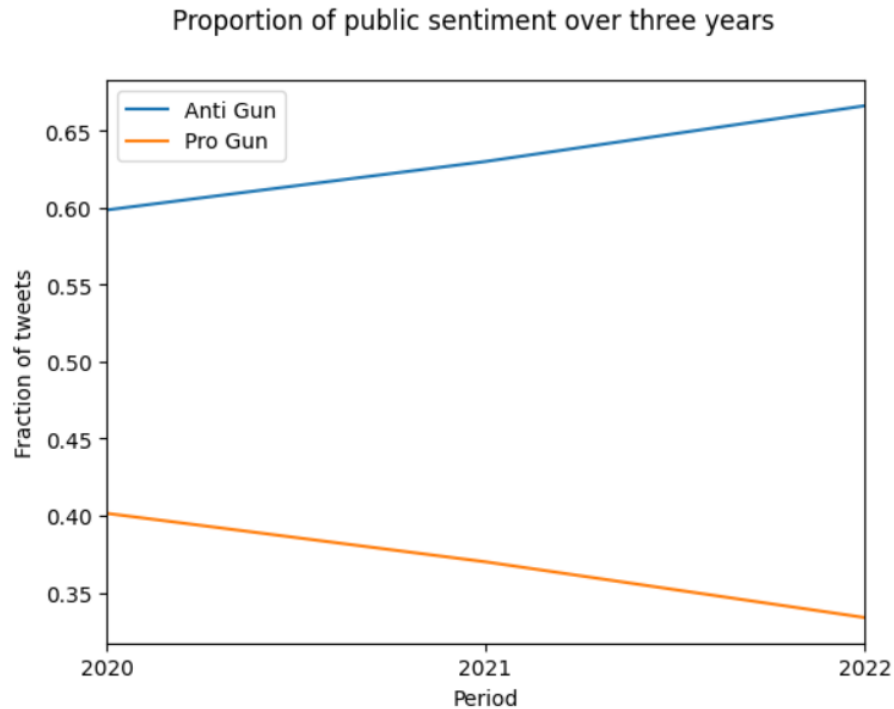


Figure 25: The proportion of pro and anti gun tweets changed linearly from 2020 - 2022

7 Limitations and Future Improvements

1. **Ambiguous data:** The collected data is observed to be quite **ambiguous** and is difficult to be labeled even by human annotators. This leads to misclassification and deteriorates performance. To efficiently obtain labeled data, domain knowledge has been used in many application areas (Ren et al., 2017; Cruciani et al., 2018; Konishi et al., 2019; Bejnordi et al., 2017). However, as some studies have pointed out (Wagner et al., 2005; Li et al., 2016; Shahriyar et al., 2018), there are often ambiguous samples that are substantially difficult to label even by domain experts. As proposed in the literature (Naoya Otani et al., 2020), to effectively solve the problem of classification with ambiguous data, we can extend classification with a reject option that trains a **classifier and a rejector** simultaneously using Positive and Negative samples based on the 0-1-c loss with rejection cost c (Cortes et al., 2016).

E.g. 1: “Remember and understand, that **this** is happening in our country, **they** have been teaching socialism to our kids for years, the same kids who join ANTIFA and BLM, just remember criminals don’t follow gun laws, it’s time to push back to keep America free.” The ambiguous questions in this example are: what is “this” referring to, who are “they”?

E.g. 2: “Mayor Lightfoot asks federal government for help fighting city’s crime”. In this example the “crime” is ambiguous. Is carrying guns and shooting people a crime? Or is reducing the freedom to carry guns a crime?

In the above examples, there is a need to establish additional context about current affairs. Such tweets should either be removed, or paired with other tweets which are replies to the given tweet to establish further context of the situation.

2. **Limited annotated data.** The team annotated 1000 data samples manually and used it to train the model. The size of this dataset was quite small and hence the models would start overfitting very quickly. It is not possible to annotate the data using pre-trained libraries like NLTK, VADER or BERT because their sentiment analysis classifies the text into positive and negative sentiments, however, we are interested in classifying the text as pro or anti gun. Additionally, the accuracy of NLTK is only around 80%, which means if we use NLTK to annotate, our results can only be as good as NLTK. Here a few suggestions to overcome this issue:
 - a. **Few Shot Learning** can be explored further to overcome this limitation since this technique does not require a large sample of training data.
 - b. Manually annotate more data

3. **Class Imbalance.** A heavy class imbalance is observed. The following attempts have been made to resolve this issue:
 - a. **Using class weights while training:** This technique does not resolve the class imbalance issue completely and the trained model tends to be biased towards the Anti-gun class.

```
history = model.fit(x_train, y_train , validation_data=[x_val, y_val],
                    epochs=12,
                    class_weight=compute_class_weights(y_train),
                    callbacks=[cp_callback])
```

- b. **Truncating data to eliminate class imbalance:** This option would have been feasible if we had a lot of training data. But since we have only 1000 data points, truncating data left us with 300 samples of each class which was insufficient for training a good classifier.

To completely resolve this issue we can manually annotate more data corresponding to the less represented class, and include it in the training data.

4. **Context Drift:** is the situation when the functional relationship between the model inputs and outputs changes. The context has changed, but the model doesn't know about the change. Its learned patterns do not hold anymore. Since our given data is heavily influenced by politics, the relationship between its features will change with changing governments. E.g. "I do not like the government's new gun control policies", would mean the person is pro gun if it is a Democratic government and anti gun if the government is Republican. We would thus need to keep **re-training the model** whenever there are changes in the ruling party.

5. **Excessive usage of satire and sarcasm**

E.g. "Just the way Putin likes it... fill USA full of guns, install a puppet; divide the country; let it blow itself up... no nukes required"

Since the studied topic is controversial, the majority of the tweets in the given dataset are sarcastic, one example shown above. This makes it extremely difficult to classify text. Sarcasm detection or detection in the shift of sentiment polarity from positive to negative over one sentence would enable the classifier to improve results.

6. **Named Entity Recognition:** The names of USA authorities, people who are involved in mass shootings, law making authorities, etc. often appear when people discuss their opinion on gun control. Adding NER to our project would thus be vital for improving the results further.

E.g. “Watched ‘Red State’ last night. It mirrors Merica. GOP and Trump family has conveniently latched onto red states because blind religion and lack of ED leaves ppl manipulable. Canada has quality education, free health care and gun control. USA is not great and likely never will be”

8 References

1. Otani, N., Otsubo, Y., Koike, T. et al. Binary classification with ambiguous training data. Mach Learn 109, 2369–2388 (2020). <https://doi.org/10.1007/s10994-020-05915-2>
2. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). Curran Associates Inc., Red Hook, NY, USA, 6000–6010.

9 Submission Links

A YouTube link to a video presentation of up to 5 minutes

<https://youtu.be/fXUhtCry97I>

A Dropbox (or Google Drive) link to a compressed (e.g., zip) file with crawled text data, queries and their results, manual classifications, automatic classification results, and any other data for Questions 3 and 5

Readme document is provided inside the folder

https://drive.google.com/file/d/1GomeWkpi25ySw54L9035KB-XyGXVyM8r/view?usp=share_link

A Dropbox (or Google Drive) link to a compressed (e.g., zip) file with all your source codes and libraries, with a readme file that explains how to compile and run the source codes

Readme document is provided inside the folder

https://drive.google.com/file/d/1bFPnDAZcKER2sD9PhF7f7v0YSpYhUtu0/view?usp=share_link