Sentiment analysis on Fluoridated water in the USA using Twitter data

Contents

1 Introduction 3

2 Technology Explanation 3

3 About the data 4

3.1 Pre-processing 5

3.2 Final data 5

3.3 I/O EXAMPLES 6

4 Results and Discussion 6

5 Assessment 8

6 Conclusion 9

1 Introduction

According to the CDC, many research studies have proven the safety and benefits of fluoridated water. Drinking fluoridated water helps prevent cavities by about 25 % in children and adults. There are mixed reviews on water fluoridation on the internet, and many people are against it. In this project, we will set up an automated framework to investigate the general feedback on water fluoridation.

Sentiment analysis is not an infamous technique in natural language processing that aims to find emotions/attitude towards a topic/sentence. It focuses on one of the three components – Polarity (positive, neutral, or negative), emotions (happy, sad, angry, etc.), and intentions (Interested or not interested). In this project, we will be analyzing public response (sentiment) for water fluoridation in the United States by performing sentiment analysis on tweets related to the topic. We will focus on the problem of identifying polarity of the tweets, classifying them as positive, negative, or neutral. Goal is to investigate if there is a correlation between the user's tweets in general and response to fluoridation in water. In other words, examine if the user has a pattern of tweeting negative tweets (e.g., negative posts related to the covid vaccine), which leads to them tweeting negatively about fluoridation too. Therefore, the model is built on tweets containing the following keywords – water fluoridation, fluoride and covid-19 vaccination.

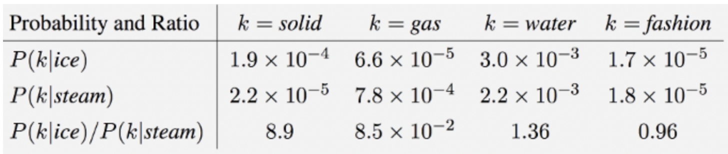
Manually assigning sentiment to tweets is a tedious task, hence, an automated way is important. In deep learning, word embedding is one of the most useful techniques to convert words/documents into vector form. The method captured a lot of attention because it successfully captures the syntactic and semantic meaning within sentence/corpora. Further, we will be experimenting random embedding and GloVe embedding with 100d and 300d. Training GloVe requires a large corpus to train. However, since we have a small size of data set, we must use pre-trained word vectors.

In this project we will experiment with different deep learning architectures with random and pre-trained embedding for the sentiment analysis. We will use validation and test data set to evaluate the performance of the different models from different architectures.

2 Technology Explanation

We will be experimenting with simple RNN, LSTM, BI-LSTM, and CNNs (with and without GloVE) methods.

Baseline method: GloVe: is an unsupervised learning algorithm for obtaining vector representations for words. Pre-trained model Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.



Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

Table 1: Probabilities from 6 billion word corpus

Reference for Table 1 and GloVe pre-trained model

Proposed method: (1) CNN: CNN can extract information but fails to persist the information; LSTM can address this limitation. **(2) LSTM:** Long Short-Term Memory Network is an advanced RNN, a sequential network. It allows information to persist the limitation of CNN and can handle the vanishing gradient problem faced by RNN.

Explored method: (1) RNN: RNNs have a unique architecture that helps them model memory units (hidden state) that enable them to persist data, thus modeling short-term dependencies. **BI-LSTM:** In BI-LSTM (bidirectional), the input flows in two directions (both backward and forward), making a BI-LSTM different from the regular LSTM. Since the data flows backward or forward it preserves the future and the past information.

3 About the data

The project requires Twitter data – tweets contain specific keywords. Since the goal is to investigate if there is a correlation between users' tweets in general Vis-à-vis water fluoridation tweets, we have considered Keywords - fluoride, fluorides, or fluoridation and Covid-19 vaccination. For Twitter data extraction, the open-source library snscrape has been used. The total number of tweets containing the keyword (from Jan to June 2020) – fluoride, fluorides, and fluoridation: 44.3K. The total number of tweets containing the keyword (from Jun to Dec 2020) - Covid-19 vaccination: 6K and the label has been annotated manually.

3.1 Pre-processing

Since we are working with the text data, there are several steps involved to clean the data:

- **URL/Emails/single quotes:** Tweets contains URL, Emails, single quotes, they are not important for the model hence, they have been removed.
- **Null value:** Checked null tweets and replaced them with no content.
- **Label encoding:** Converted the label (-1,0,1) into a machine-readable numeric form, used label encoding for the same. There are NAs in the label, which means the tweet is not very clear, since the future data will also consist NAs those NAs have been considered as neutral.

3.2 Final data

The final input to the model data contains 820 tweets with the keyword Covid-19 Vaccine, and tweets containing fluoride OR fluorides OR fluoridation keywords are 804. Further polarity distribution is as per the below table.

Keywords	Sentiment	Count
Covid-19 Vaccine	Negative	229
	Neutral	372
	Positive	219
fluoride OR fluorides OR water fluoridation	Negative	315
	Neutral	398
	Positive	81

Table 2: Count of sentiments

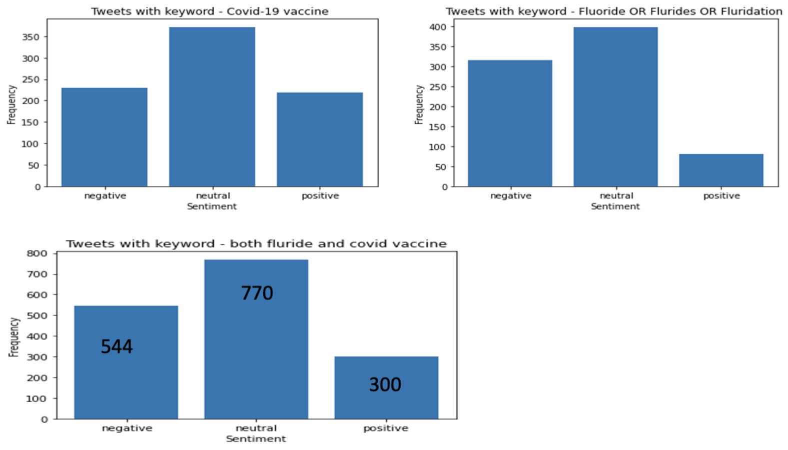
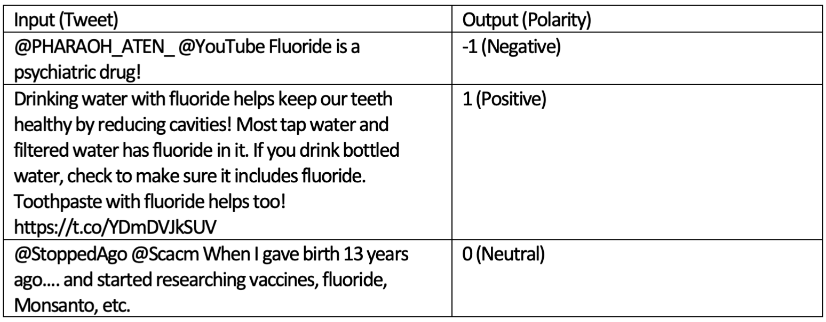


Figure 1: Polarity distribution in tweets containing Covid-19 vaccination, Fluoride, and Total

3.3 I/O EXAMPLES

Below table is an I/O example. Input is the text data (which before feeding to model will be converted into sequences) and output is polarity.

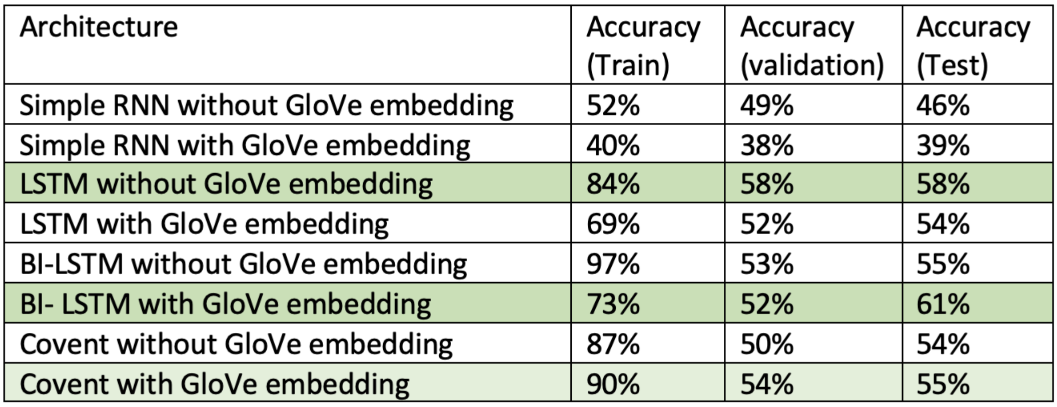


Input (Tweet)	Output (Polarity)
@PHARAOH_ATEN_ @YouTube Fluoride is a psychiatric drug!	-1 (Negative)
Drinking water with fluoride helps keep our teeth healthy by reducing cavities! Most tap water and filtered water has fluoride in it. If you drink bottled water, check to make sure it includes fluoride. Toothpaste with fluoride helps too! https://t.co/YDmDVJkSUV	1 (Positive)
@StoppedAgo @Scacm When I gave birth 13 years ago.... and started researching vaccines, fluoride, Monsanto, etc.	0 (Neutral)

Table 3: I/O Example

4 Results and Discussion

Below is a summary of the results from the various architectures explored. We have also tested the ResNet Unit network with and without GloVE, which did not give a reasonable accuracy on validation and test.

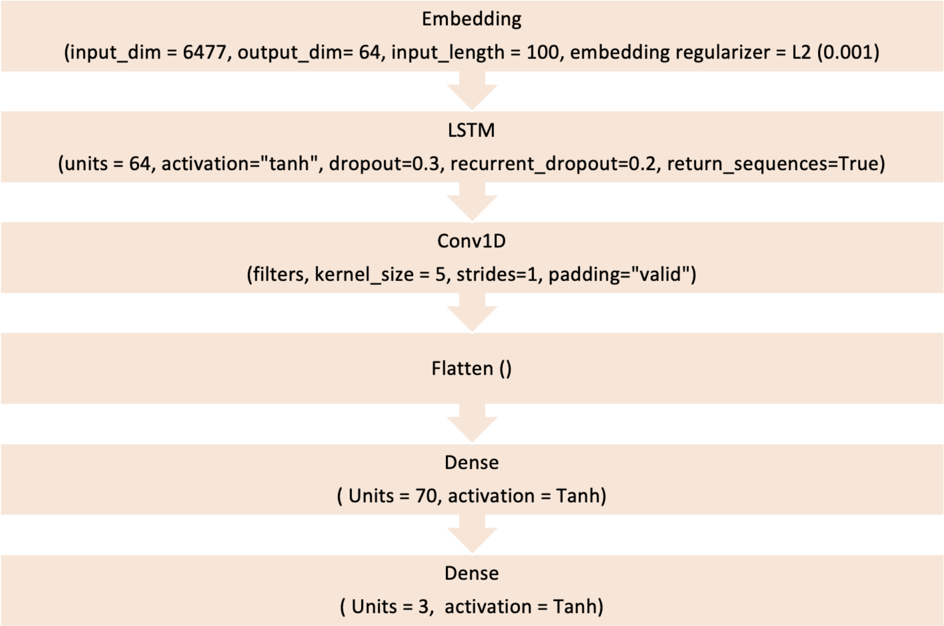


Architecture	Accuracy (Train)	Accuracy (validation)	Accuracy (Test)
Simple RNN without GloVe embedding	52%	49%	46%
Simple RNN with GloVe embedding	40%	38%	39%
LSTM without GloVe embedding	84%	58%	58%
LSTM with GloVe embedding	69%	52%	54%
BI-LSTM without GloVe embedding	97%	53%	55%
BI- LSTM with GloVe embedding	73%	52%	61%
Covent without GloVe embedding	87%	50%	54%
Covent with GloVe embedding	90%	54%	55%

Table 4: Combination of architecture explored

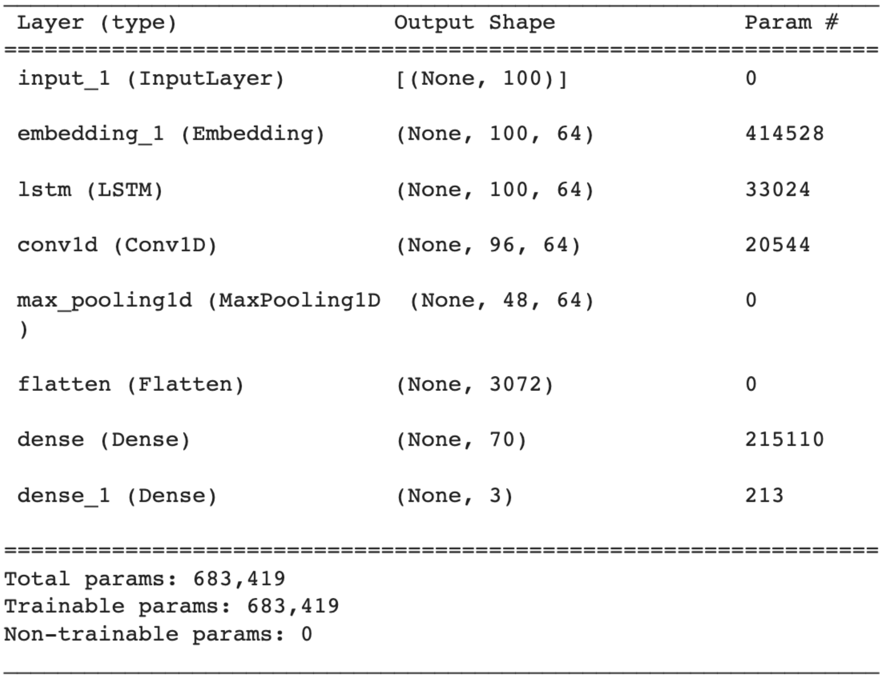
Note: The Epoch for all the architecture is 5, and the activation function is Relu.

As we can see in the above table that LSTM without GloVe performed better in both validation and test data sets, whereas the second best was BI-LSTM with GloVE. Since there was a significant difference in the accuracy of validation and test in the second-best model, i.e., BI-LSTM with GloVE final model was built using LSTM and Conv1D layers without Glove embedding. Below is the final model architecture with summery.


Embedding
(input_dim = 6477, output_dim= 64, input_length = 100, embedding regularizer = L2 (0.001)
LSTM
(units = 64, activation="tanh", dropout=0.3, recurrent_dropout=0.2, return_sequences=True)
Conv1D
(filters, kernel_size = 5, strides=1, padding="valid")
Flatten ()
Dense
(Units = 70, activation = Tanh)
Dense
(Units = 3, activation = Tanh)

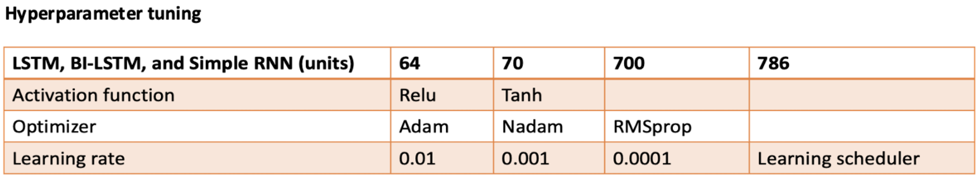
Note: Optimizer used: rmsprop , 6477: word index

Figure 2: Final Model Architecture



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 100)]             0

 embedding_1 (Embedding)     (None, 100, 64)           414528

 lstm (LSTM)                 (None, 100, 64)           33024

 conv1d (Conv1D)             (None, 96, 64)            20544

 max_pooling1d (MaxPooling1D  (None, 48, 64)           0
 )

 flatten (Flatten)           (None, 3072)              0

 dense (Dense)               (None, 70)                215110

 dense_1 (Dense)             (None, 3)                 213

=================================================================
Total params: 683,419
Trainable params: 683,419
Non-trainable params: 0
_________________________________________________________________
```

Figure 3: Model Summary



Hyperparameter tuning

LSTM, BI-LSTM, and Simple RNN (units)	64	70	700	786
Activation function	Relu	Tanh		
Optimizer	Adam	Nadam	RMSprop	
Learning rate	0.01	0.001	0.0001	Learning scheduler

Table 5: Combinations of hyperparameter explored

5 Assessment

Success is measured by the accuracy of classification of Train, validation, and Test data set along with Recall, specificity, and F-Score. Below is the Confusion matrix generated from the model.

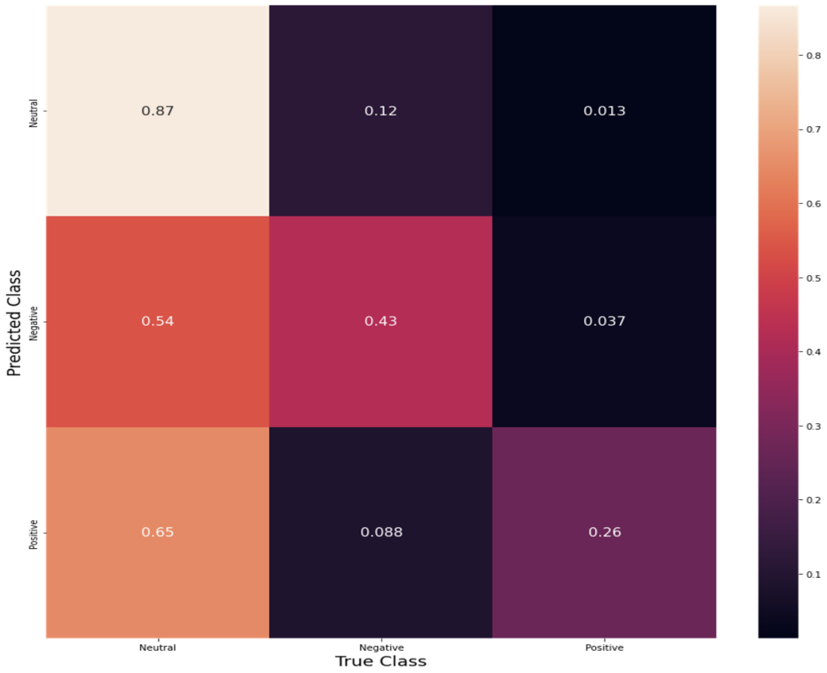


Figure 4: Confusion matrix

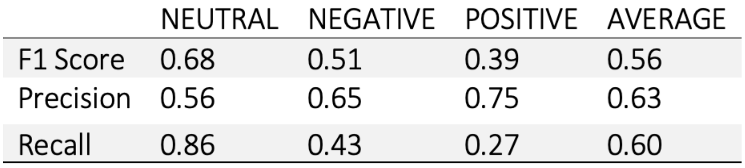
We have computed Precision, recall, and F1-score using the sklearn library. Precision, recall, and F1-score computation as per below formula.

$$\text{Precision}(class = a) = \frac{TP(class = a)}{TP(class = a) + FP(class = a)}$$

$$\text{Recall}(class = a) = \frac{TP(class = a)}{TP(class = a) + FN(class = a)}$$

$$\text{F-1 Score}(class = a) = \frac{2 \times \text{Precision}(class = a) \times \text{Recall}(class = a)}{\text{Precision}(class = a) + \text{Recall}(class = a)}$$

F1 Score, Precision, Recall ranges from 0 to 1; an F1 score is considered perfect when it's 1. Precision helps when the costs of false positives are high, whereas Recall helps when the cost of false negatives is high, and F1 is an overall measure of a model which combines precision and Recall. Below is the table for Precision, Recall, and F1-score computed basis of the Neutral, Negative, Positive, and average of all counts in all classes as weights.



	NEUTRAL	NEGATIVE	POSITIVE	AVERAGE
F1 Score	0.68	0.51	0.39	0.56
Precision	0.56	0.65	0.75	0.63
Recall	0.86	0.43	0.27	0.60

Table 6: Precision, Recall, and F1 Score of the model

As we can see, for class neutral, F1 Score, Precision, and Recall are around/more than 0.60. Whereas for class Negative, F1 Score, Precision, and Recall about/more than 0.50, the average was close to 0.60 across the scores.

6 Conclusion

In this project, we attempted to use sentiment analysis on Twitter data to gauge user polarity (positive, negative, neutral) towards the topic of fluoridation in drinking water in the US. We experimented with simple RNN, LSTM, BI-LSTM, and Conv1D architectures to build the model. We found that LSTM and Conv1D, along with the Dense layer, performed better than the single layer of LSTM/ Conv1D. Activation function Tanh and optimizer RMSprop gave the highest accuracy.

In the future, we can use the model to see the polarity distribution of users on the topics of fluoridated water and Covid-19. Also, we can analyze the user's tendency to Twitter posting. We can also test the model on a general population containing any keywords.

[1] [2] [3] [4]

References

[1] A. Géron, in *Hands-On Machine Learning with Scikit-Learn and TensorFlow*, vol. 2. O'Reilly.

[2] G. C. B. G. K. Usha Devi Gandhi, Priyan Malarvizhi Kumar, "Sentiment analysis on twitter data by using convolutional neural network (cnn) and long short term memory (lstm)," 2021. [Online]. Available: https://link.springer.com/article/10.1007/s11277-021-08580-3

[3] K. R. L. Jin Wang, Liang-Chih Yu and X. Zhang, "Dimensional sentiment analysis using a regional cnn-lstm model," 2016. [Online]. Available: https://aclanthology.org/P16-2037.pdf

[4] M. G. V. V. Pinkesh Badjatiya, Shashank Gupta, "Deep learning for hate speech detection in tweets," 2017. [Online]. Available: https://arxiv.org/abs/1706.00188

http://nlp.stanford.edu/projects/glove/