Pro-Choice vs. Pro-Life: Tweets Exploratory Data Analysis

Xingyu Chen   
 Department of Data Science  
University of Colorado at Boulder  
Boulder, CO 80309 USA   
xich9921@colorado.edu

***Keywords—Seaborn, Data Visualization, Data Model, Neural Nets, ANN, CNN, RNN, LSTM, NLP***

# Introduction

The 1973 Roe vs. Wade decision, legalizing abortion in all fifty states, seems to solve one of the most controversial issues in terms of abortion. However, on June 24, 2022, the Supreme Court issued a bill prohibiting women's access to out-of-state abortion services. Additionally, the Court banned abortions nationwide after 15 weeks of pregnancy, overturning the Roe v. Wade case.

The Congress's decision has drawn much attention on social media, especially from females. The argument divides into two opinions: Pro-Choice vs. Pro-Life. People who support pro-choice believe everyone has the fundamental human right to decide when and whether to have children. They think it is OK for them to have the ability to choose abortion as an option for an unplanned pregnancy – even if they would not choose abortion for themselves. The view is that a woman should have the legal right to an elective abortion, meaning the right to terminate her pregnancy.

People who support pro-life believe that the life of the fertilized egg, embryo, or fetus is much more critical. They despise children's welfare after birth and oppose child welfare legislation. The controversial issues pit people against each other like they are on two teams. Most Americans believe abortion should be legal because it is the human right to access abortion.

This paper presents an exploratory data analysis on tweets about pro-choice vs. pro-life. It is helpful for people who want a general idea about how people react to the bill that bans abortion in certain states, especially for females. The neural network model helps them to grasp a pragmatic understanding of the whole event timeline.

More specifically, the model should provide a decent result so people can learn primary online users' opinions behind the case. In addition, this model should facilitate decision-makers in Congress to pass bills involving controversial issues because it generates local and global impacts at a certain level.



Fig 1. Abortion-rights movements

# Data Gather

The dataset of 56,040 tweets was collected in the wake of the Roe vs. Wade cancellation sentence and analyzed the influence operations. The dataset is available to download from the Kaggle website, which lists in the reference.

The tweets are collected containing either the #prochoice or the #prolife hashtag, reflecting the two opposite poles of the discussion. The tweets with #prochoice have a target variable of 0, and the tweet with the #prolife has a target variable of 1.Table

Description automatically generated

Fig 2. Original dataset

Other columns, such as ‘created\_at,’ ‘retweet\_count,’ and ‘like\_count,’ can be valuable features for the data model. On the other hand, the ‘author\_name,’ ‘author\_username,’ and other columns are irrelevant to this paper and will be omitted during the data preprocessing.

In addition to the original dataset, newsapi.org provide API to gather unlabeled news titles related to Pro-Choice vs. Pro-Life. Registered a free account on newsapi.org to get the API key and set up an endpoint for the servers and the location on the server where data will be retrieved. The newspaper servers will query for all the topic names in the list: 'abortion' and 'antiabortion.'

The server will respond in JSON format with the date, title, headline, and source. Then the JSON format transforms into a large CSV file where each article is in a row. Adding one column that either abortion or antiabortion to convert this data into a labeled data frame so model train and test with the data.A picture containing text

Description automatically generated

Fig3. Dataset retrieved from API

# Data Prep

Preprocessing is a crucial step in processing text, especially for tweets. Use standard text preprocessing techniques and tweets-specific preprocessing techniques to preprocess tweets. The standard preprocessing technique uses NLTK (Natural Language Tool Kit) library. The tweets-specific preprocessing technique uses Regular Expression (re in python library).

The DateTime module transforms the ‘date’ column into ‘hour’ and ‘month.’ Moreover, use feature extracting to count words and sentence length to add more features to the dataset. Here is the list of tweets-specific preprocessing tasks using a regular expression:

1. Lowercasing all the letters
2. Remove mentions ‘@.’
3. Remove hashtags ‘#.’
4. Remove links. Start with ‘HTTP’ or ‘www.’
5. Remove punctuations
6. Remove non-alphanumeric characters
7. Remove stop words

Graphical user interface, text

Description automatically generated

Fig 4. Preprocessed dataset

Then, we tokenize and vectorize text with stemming. The goal is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form. Here, we remove all columns containing numbers and any column with a name of 3 or smaller, like ‘it,’ ‘of,’ and ‘pre.’

After we tokenize the text, we encode it into a 500 numeric length array to represent the original text so the model can recognize them. We use the embedding (the matrix we made in the previous steps) to encode the reader into an index inside the embedding. The padding method modifies that every encoded sentence will be the same length as 500. We append the ‘pad’ symbol if the distance is less than 500 and get the first 500 tokens if the size is more significant than 500.

We also split the dataset into train, valid, and test data with a ratio of 60, 20, and 20. The shape will be (33624,12), (11208,12), and (11208,12) corresponding. Using vocab to convert reviews (text) into numerical form, Replacing each word with its corresponding integer index value from the vocabulary. Assign the max length of the vocab + 1 to terms, not in the vocab. For the dataset from news API, we do similar preprocessing like tweets, but we focus on the title and headline.

# Baseline: Neural Net w/ Backpropagation

Use preprocessed data to train a simple, one-layer NN with backpropagation, random weights, and biases. This model aims to set up a baseline that the model describes later, ideally providing better performance. This network will not use TensorFlow/Keras, Sklearn, or other packages. Here is the network structure:

Diagram, engineering drawing

Description automatically generated

Fig 5. Neural Net structure

Backpropagation is a process that takes the error rate of forward propagation and feeds this loss backward through the neural network layers to fine-tune the weights. After iteration 1000, the total loss is 4840.21, and the average loss is 0.12. The accuracy score is 0.78, and the confusion matrix list below, along with the loss plots. There is an overfitting issue with this NN due to the simple net structure.

[[10232 8087]

[ 16 19211]]

Chart, histogram

Description automatically generated

Fig 6. Total loss

Chart, histogram

Description automatically generated

Fig 7. Average Loss

# Neural nets

Build four neural nets with preprocessed train/validation/test datasets for predicting pro-choice and pro-life based on TensorFlow and Keras (ANN, RNN, LSTM, CNN). Evaluate the model with a confusion matrix, loss, and accuracy with the sklearn library.

The network intended to predict the tweets support pro-life or pro-choice, and the input vector is retweet\_count, like\_count, words\_count, sentence\_length, and hour.

For Convolutional Neural Networks, we need to define the convolutional layer, a generalization of the Dense layer. The convolution means sliding a flipped kernel, so most of the library use cross-correlation to implement the convolution layer because it is sliding a kernel without a flip.

Recurrent Neural Networks maintain an internal state and will update at every step. The set of weights is the same across all time steps through the sequence. The first layer is the embedding layer, which will learn embeddings for different words.

We define input with the vocabulary size and length of input sequences. We define output with a dimension of dense embedding. It is the size of the vector space in embedded words and establishes the size of the output vectors from this layer for each word.

Embedding layer, the word with similar meanings or that often occurs together in similar contexts will have a similar vector representation based on how close or far apart those words are in their meanings. The most common feature vectors are Word2Vec from Google and GloVe from Stanford.

In an embedding, each dense vector represents the projection of the word into a continuous vector space. A word's position in the learned vector space is called its embedding. The dense layers require inputs such as batch size and input size.

For Long Short Term Memory Networks, it contains computational blocks that control information flow. The gates control information is added or removed through structures. It can forget irrelevant information, store relevant information from current input, update the cell state selectively and output a filtered version of the cell state.

The encoder LSTM processes the entire input sentence and encodes it into a context vector, which is the last hidden state of the LSTM/RNN. The final state is the initial hidden state of the decoder. The decoder LSTM/RNN produces the words in a sentence one after another.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | ANN | RNN | LSTM | CNN |
| Train Loss | 0.4563 | 0.4098 | 0.4134 | **0.2343** |
| Train Accuracy | 0.7713 | 0.7729 | 0.7627 | **0.8568** |
| Validation Loss | 0.8828 | **0.8117** | 1.0254 | 1.6699 |
| Validation Accuracy | 0.6130 | 0.6143 | **0.6291** | 0.5989 |
| Test Loss | 0.8978 | **0.8269** | 1.0349 | 1.7021 |
| Test Accuracy | 0.6077 | 0.6205 | **0.6246** | 0.6042 |

Table 1. Neural Nets Loss and Accuracy on Tweets

|  |  |  |
| --- | --- | --- |
|  | ANN | RNN |
| Confusion Matrix | 4116 3010  1386 2696 | 3619 2370  1883 3336 |
|  | LSTM | CNN |
| Confusion Matrix | 3625 2330  1877 3376 | 3490 2424  2012 3282 |

Table 2. Confusion Matrix for detailed evaluation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | ANN | RNN | LSTM | CNN |
| Train Loss | 0.7497 | **0.5714** | 0.6873 | 0.6715 |
| Train Accuracy | 0.5131 | **0.7487** | 0.5654 | 0.6073 |

Table 3. Loss and Accuracy on dataset gathered from API

RNN and LSTM perform better compared to other two neural nets; RNN and LSTM perform similar result due to similar structure. ANN is unstable neural nets because it may predict all labels as 0 or 1. CNN is overfitting on train set and perform bad on test/validation set. On the other hand, RNNs perform better among all four neural nets in relatively more minor datasets with smaller batch sizes and learning rates.

# Conclusion

Tweets exploratory data analysis is a challenging but also necessary task. In this paper, we have 56,040 tweets labeled pro-choice or pro-life, and 200 sample news titles and headlines labeled abortion or antiabortion. Five neural nets predict output from these data. They potentially benefit readers or decision-makers in the supreme court towards a better understanding of how the public sees and sees abortion in (digital) life.

This paper is not intended that take a position on the discussion on the right to abortion. Focus on the ethical arguments and underlying issues rather than on political considerations that might also be involved. This dataset takes its cue from this discussion to create a corpus of tweets that can be tagged a priori. Text

Description automatically generated with low confidence

Fig 8. Pro-life Word Cloud

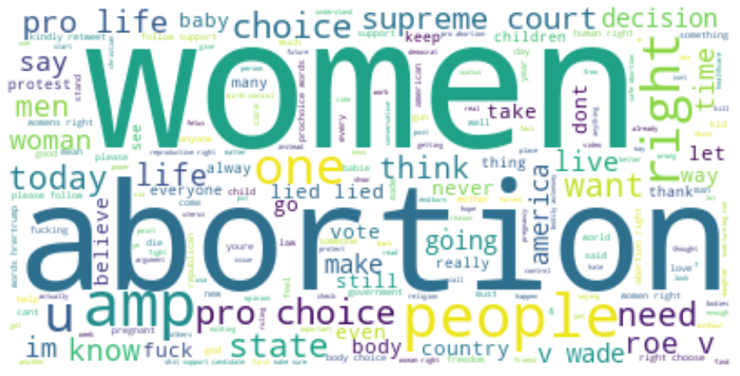


Fig 9. Pro-Choice Word Cloud

The top words for pro-life are people, right, children, and mother; meanwhile, the top words for pro-choice are women, decision, one, live, and believe. These words contribute to identifying pro-life/pro-choice labels.

Identifying pro-choice or pro-life in texts could be used in various applications like tweets sentiment analysis, argument faceted search, and value-based opinion profiling. It is important to note that these samples do not represent the user’s opinion but provide a benchmark for measuring classification robustness across sources.

For now, this paper introduced five neural nets, and the best neural nets provide around 80% accuracy with reasonable training time and memory space. A more significant community effort is needed to collect more solid datasets from a wider variety of sources besides Twitter.

1. Data Visual

This paper does not intend to provide sentimental analysis for these tweets but give a data insight behind the texts. Using NLTK library to categorize each sample into positive, negative, and neutral sentiment. Then, using seaborn and Plotly library to visual the dataset.

Chart, pie chart

Description automatically generated

Fig 10. Count plot of sentiments of the data

Chart

Description automatically generated

Fig 11. Sentence Length of based on date

Chart, histogram

Description automatically generated

Fig 12. Sentence Length Distribution

Chart, bar chart, histogram

Description automatically generated

Fig 13. Activity all over the day

We can see that most tweets are negative sentiment toward to abortion topic and people continually tweets about it with length 20 ~ 50 sentence length. Most of tweets occurs afternoon or after 2 pm.

##### References

[1] Twitter supervised dataset - prochoice VS prolife. (n.d.). Retrieved December 3, 2022, from

<https://www.kaggle.com/datasets/mcantoni81/twitter-supervised-dataset-prochoice-vs-prolife>

[2] Saxena. (2021, February 6). *Understanding Embedding Layer in Keras*. Analytics Vidhya. <https://medium.com/analytics-vidhya/understanding-embedding-layer-in-keras-bbe3ff1327ce>

[3] https://www.facebook.com/jason.brownlee.39. (2017, October 3). How to Use Word Embedding Layers for Deep Learning with Keras. Machine Learning Mastery. <https://machinelearningmastery.com/use-word-embedding-layers-deep-learning-keras/>

[4] tf.Keras.Input | TensorFlow Core v2.8.0. (n.d.). TensorFlow. <https://www.tensorflow.org/api_docs/python/tf/keras/Input>

[5] tf. Keras.layers.Embedding | TensorFlow Core v2.9.1. (n.d.). TensorFlow. <https://www.tensorflow.org/api_docs/python/tf/keras/layers/Embedding>

[6] tf.Keras.Layers.Bidirectional | TensorFlow Core v2.3.0. (n.d.). TensorFlow. <https://www.tensorflow.org/api_docs/python/tf/keras/layers/Bidirectional>

[7] News API - A JSON API for live news and blog articles. (2019). News API - A JSON API for live news and blog articles. Newsapi.org. <https://newsapi.org/>

[8] Team, K. (n.d.). Keras documentation: Embedding layer. Keras.io. <https://keras.io/api/layers/core_layers/embedding/>