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, which overturned 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 to have 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 text 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 length is less than 500 and get the first 500 tokens if the length 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 words, not in the vocab. For the dataset from news API, we do similar preprocessing like tweets, but we focus on the title and headline.

# Data Visual

Word cloud

# 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

For ANN, we define first layer as embedding layer that will learn embeddings for different words. We define input with size of the vocabulary and length of input sequences. We define output with dimension of the dense embedding. It is the size of the vector space in which words will be embedded and it defines the size of the output vectors from this layer for each word.

In an embedding, words are represented by dense vectors where a vector represents the projection of the word into a continuous vector space. The position of a word within the vector space is learned from text and is based on the words that surround the word when it is used. The position of a word in the learned vector space is referred to as its embedding. The dense layers require inputs as batch size and input size.

Configure model and train/validation/test model. We implement four models (ANN, RNN, LSTM, CNN), we use three dataset(train, validation, test) to evaluate the model. They are all based on tensorflow and keras. And confusion matrix will be generated by sklearn library.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | ANN | RNN | LSTM | CNN |
| Train Loss |  |  |  |  |
| Train Accuracy |  |  |  |  |
| Validation Loss |  |  |  |  |
| Validation Accuracy |  |  |  |  |
| Test Loss |  |  |  |  |
| Test Accuracy |  |  |  |  |
| Confusion matrix | 1616 481  879 2024 |  |  |  |

It seems that LSTM and CNN perform better than ANN and RNN for this dataset, LSTM perform generally better but with more training time and space.

NewsAPI (200 samples for two different topic):

Two topics: antiabortion and antiabortion

Get 100 samples for abortion and 100 samples for election.

Top words for abortion: rights, voters, states, midterm

Top words for election: polls, state, president, close, results

ANN RNN LSTM CNN

Train Loss 0.7347 0.5301 0.6923 0.6798

Train Accuracy 0.5050 0.7300 0.5400 0.5150

RNN perform better for the dataset gather from NewsAPI

# Conclusion

This dataset is not intended to be used to take a position on the discussion on the right to abortion. I 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.

I use neural networks intended to answer following questions:

1. How is people reaction changes between dates?

2. Can we use the neural net to predict tweets opinion?

3. What is the frequency of tweets during the whole timeline?

4. What are the words that contribute to pro-life/pro-choice

For now, the epoch is 1000 and learning rate 0.01 with sample size 37,546, we get 78% accuracy. Parameter tuning will be used during next tasks.

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

All projects must be submitted with a 5 paragraph CONCLUSIONS that clearly describes how the exploration, prediction(s), modeling, and/or other project methods support, clarify, or assist with the topic, Conclusions are 100% non-technical and should contain images that help the reader to understand the "take-home" and key messages, discoveries, and/or predictions. Each project will have a different conclusion based on the topic and goals.

##### 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/>

‌

‌

‌