Recurrent Neural Network Variants for Generating Opinion Distributions

Kerria Pang-Naylor
Department of Computer Science
Harvey Mudd College
kpangnaylor@g.hmc.edu

Arjun Taneja
Department of Computer Science
Harvey Mudd College
ataneja@g.hmc.edu

Abstract

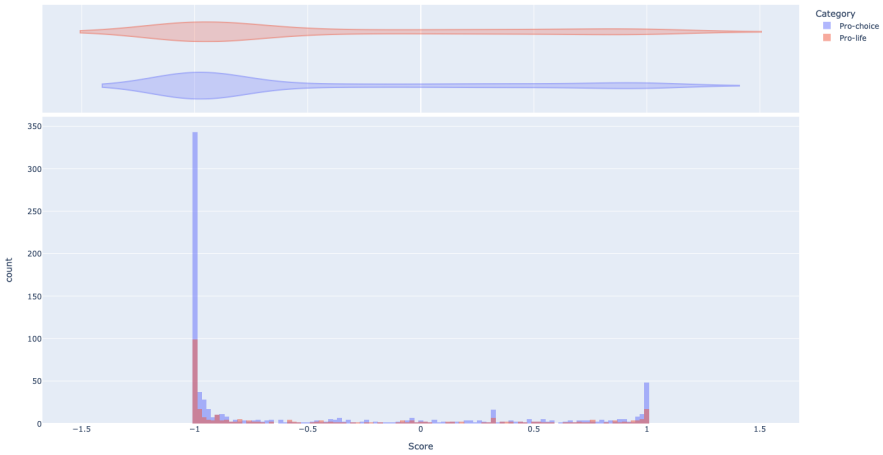
Bounded-confidence information dissemination models assume a non-polarized continuously valued underlying opinion distribution. Past attempts of using probabilistic logistic regression assignments have been overfit and produce heavily polarized distributions lacking the overlap in opinion on which the bounded-confidence mechanism relies. In this project, we train and regularize a new nonlinear algorithm – a single layer long short-term memory (LSTM) neural network – for probabilistic opinion assignment. While less accurate, this model produces non-polarized opinion distributions compatible with the bounded confidence mechanism.

1 Introduction and related work

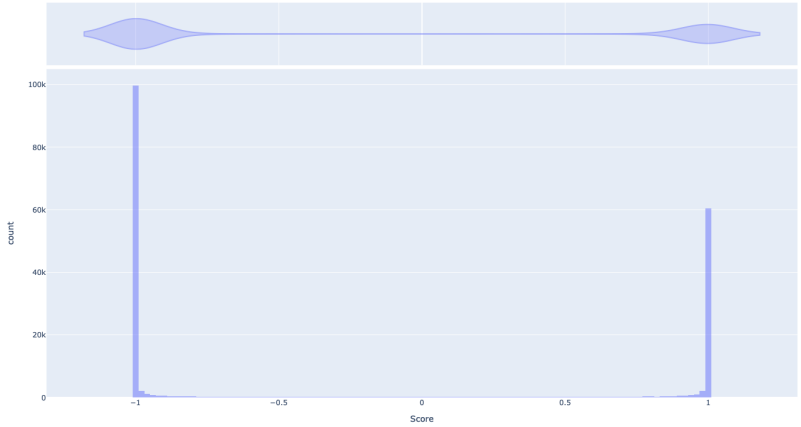
Opinion dynamics models aim to simulate observed social behaviors and opinion spreadBrooks and Porter (2020). Bounded-confidence models, a type of opinion dynamics model, focus on our tendency towards consensus and the preference for similar opinions. These models, most notably the Deffuant-Weisbuch or Hegselmann-Krause models, are characterized by a fixed cutoff or 'confidence bound' parameter c that restricts agents' interaction; if the difference between two nodes' opinions exceeds c, they may no longer influence each other's opinions Chen et al. (2020). This bounded-confidence mechanism acts on continuous-valued opinions within a network, and they assume a continuously-valued underlying opinion distribution.

However, the theoretical nature of these models prevent them from providing practically meaningful results. By fitting a bounded-confidence mechanism with real data, namely social media datasets, we can create more accurate simulations of opinion spread, and with it, meaningful insights into the mechanisms behind real-world online social dynamics like polarization, echo chambers, and the spread of misinformation Brooks and Porter (2020). A first step towards fitting bounded confidence with real data is to generate the underlying opinion distributions of social networks, which call for algorithms that assign scores of political leaning to text data. We have already produced logistic regression classification models that discretely label tweets and Twitter bios into one of two opinion categories. Due to limitations in gathering and labeling training data, we found that building multi-label classification models or a non-classification regression models is likely impractical. A potential solution to this would be to assign datapoints' opinions according to the determined probabilistic score. Unfortunately, we found that our original models created heavily polarized opinion distributions and were likely overconfident and overfit. These models originally had been optimized to increase classification accuracy rather than to produce a balanced opinion distribution. They produced heavily polarized opinion distributions even when tested on datasets irrelevant to training data (see figure 1), which, if correctly trained, the model should recognize as unrelated and therefore be "less confident" in its probabilistic assignments. Given that bounded confidence models rely on proximity of opinions

37th Conference on Neural Information Processing Systems (NeurIPS 2023).



(a) Relevant dataset SurgeAI Roe v. Wade reaction tweets.



(b) Unrelated COVID19 dataset.

Figure 1: Original Roe v. Wade tweet classification model (logistic regression) distribution on irrelevant COVID19 dataset and relevant Roe v. Wade tweet dataset.

for agents' interaction, classification models like these are incompatible with bounded confidence mechanisms.

In this project, we train and experiment with the parameters of a new nonlinear algorithm – a single layer Long short-term memory (LSTM) neural network – for probabilistic opinion assignment. We believe that this approach may be more successful than our previous use of a simple, linear logistic regression models with TF-IDF vectorization for tweet classification. LSTMs, a type of recurrent neural network (RNN), excel in capturing temporal dependencies and patterns in sequential data such as text data and can make predictions based on contextual understanding (e.g., informal language, abbreviations) Dupond (2019). LSTMs also handle the vanishing gradient problem of traditional RNNs, making them more applicable for messy and automatically labeled data like our custom tweet dataset Hu et al. (2018).

Through experimentation with altering vocabulary/feature size, dropout rate, and L1/L2 regularization, we have produced a $\sim 70\%$ accurate LSTM classifier that successfully creates non-polarized opinion distributions on both related and unrelated datasets. While its accuracy can be improved in future projects, point-testing our model with custom words and phrases also confirm that this algorithm is sufficiently "unsure" enough to be applicable for generating opinion distributions.

2 Datasets

At the time this research project began, we wanted to study the Twitter reaction to the overturn of Roe v. Wade. Unfortunately, few labeled datasets existed at the time, and those that did were far too small for model training. To workaround this, tweets were extracted directly through the Twitter API, and labeled based on who each tweet's author was following. We first extracted and labeled about 260 thousand users depending on whether they were followers of @NARAL, the National Association for the Repeal of Abortion Laws, a well-known pro-choice movement, or @March_for_Life, a well-known pro-life movement. Then, we queried tweets from these users along with the keywords "roe", "wade", and "unborn" within 5 days of the overturn, and labeled the tweets accordingly. This resulted in a dataset with 10616 labeled pro-choice tweets and 8816 labeled pro-life tweets on which our algorithms are trained.

Through exploratory data analysis and random sampling, this automated labeling method appears to almost always categorize Tweets correctly. The histograms of Figure 2 display the top 20 non-stopwords from each side, and is largely consistent with what most would expect.

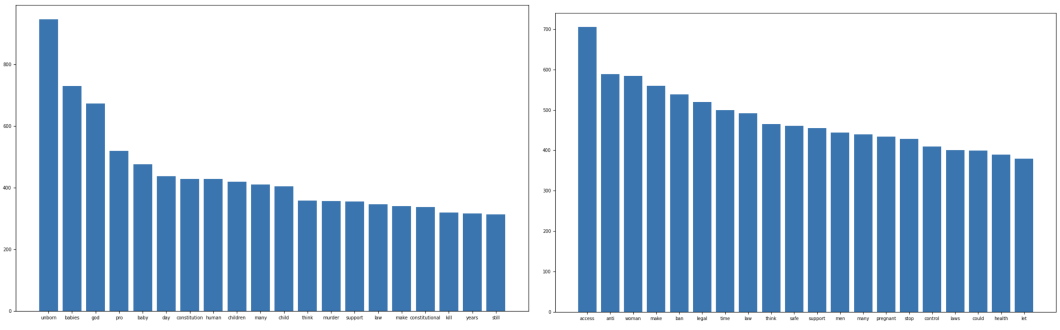


Figure 2: Top 20 non-stopwords (left) for Pro-Life and Pro-choice (right) tweets.

We also use two external datasets for testing our trained models: a small (5000 entry) freely available Roe v. Wade overturn labeled tweet dataset from SurgeAI SurgeAI (2022) (with none of the same tweets as our dataset), and an unrelated, unlabeled COVID19 tweet dataset from Kaggle P. (2020). We use the SurgeAI dataset to determine how well our models work with a large non-training sample of tweets related to the topics it was trained on. We use the unrelated COVID19 tweet dataset to gauge overfitting/model overconfidence by analyzing how confidently our models assign probabilistic scores to unrelated data.

3 Methods

3.1 Long short-term memory (LSTM) networks

The model that we used for this project is a single-layer long short-term memory neural network (LSTM) with a cross-entropy loss function. An LSTM is variant of a recurrent neural network (RNN) which is amenable to the opinion dynamics problem because, like RNNs, they are able to handle arbitrary-length input sequences (tokens from tweets), and captures positional dependencies between inputs. LSTMs are also designed to mitigate the vanishing gradient problems inherent to "vanilla" RNNs Hu et al. (2018). We will demonstrate why this is the case.

Notice there are two pathways that influence the output of the LSTM. There is the cell state (long term memories) notated above with the vector c_{t-1}, and the hidden state (short-term memories) notated with h_{t-1}. Along with the input x_t, a single pass through the LSTM updates the hidden state h_t and cell state c_t. Then, as with RNNs, the hidden state can be interpreted as the output of the layer, or can be fed into the next iteration of the LSTM along with the next input x_{t+1} Greff et al. (2017).

Recall that RNNs suffer from vanishing gradients because during backpropagation, the gradient is repeatedly multiplied by the same set of weights. For example, let $g_1, g_2, g_3, \ldots g_n$ the the gradients corresponding to the kth token in an input sequence. Since the weights in an RNN are identical, by the time we reach the first application of the RNN layer our gradient looks like

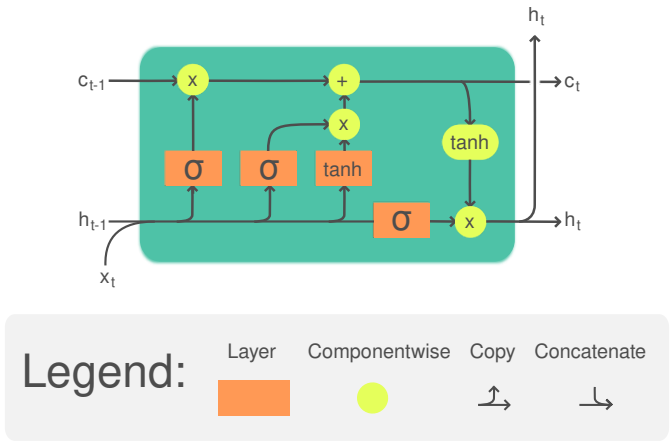


Figure 3: Diagram of LSTM cell Odhiambo et al. (2022).

$$w \cdot g_1 \cdot w \cdot g_2 \cdot \ldots \cdot w \cdot g_n = w^n g_1 g_2 \ldots g_n$$

If $|w| < 1$, then w^n will tend to 0 and the gradient will vanish. This will cause the network to learn extremely gradually Greff et al. (2017).

The way that LSTMs avoid the vanishing gradient problems is due to how the cell state is updated. These are reflected in the update equations:

$$\begin{aligned}
f_t &= \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \\
i_t &= \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \\
o_t &= \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \\
\tilde{c}_t &= \sigma(W_c x_t U_c h_{t-1} + b_c) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\
h_t &= o_t \odot \sigma_h(c_t)
\end{aligned}$$

where $\odot$ represents the elementwise product. Even though the equations looks complicated, notice that the cell state c_t is updated with the **sum** of vectors whose elements are between 0 and 1. This avoids the multiplicative effect above that would cause the gradient to vanish Hu et al. (2018).

There is more to say about the LSTM layer, including the functions of each "gate" and the mechanics of the formation of new memories. However, we deem that this level of detail is unnecessary for justifying the use of LSTMs for this particular task. We hope that this section has provided enough motivation and insight for our use of LSTMs.

For our project, we chose to train a **single** layer LSTM as they are more suitable for simple tasks and less prone to overfitting Breuel (2015). We also chose the cross-entropy loss metric because it has been shown to work well for text classification problems Breuel (2015).

3.2 Data cleaning and balancing

We first prepared our data for training by removing punctuation, new line characters, hyperlinks, emojis, and extra whitespace. Then, we experimented with training the LSTM no the dataset balanced by deletion, duplication, and no balancing (Figure 4). This initial LSTM considered the 20000 most common words in its embedding, no dropout rate, and no regularization. While balancing by duplication produced the greatest test accuracy with about the same test/train difference, when testing custom phrases with the model, we found that balancing by duplication trained an algorithm that gave all unrelated prompts scores leaning the majority, pro-choice side. While balancing by deletion produced a slightly lower test accuracy, the models it produced did not have this issue, and so we chose to balance our data by deleting the appropriate number of pro-choice entries. Finally, when preparing the newly balanced data for training, we partitioned it into 20% for testing and 80% for training.

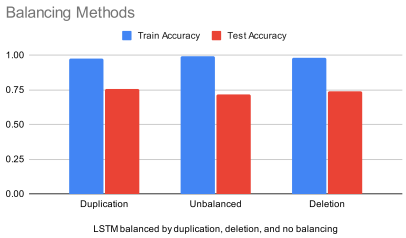


Figure 4: Train and test accuracy of balancing methods.

3.3 Choosing Parameters

Because the purpose of this model was to produce non-polarized opinion distributions, we prioritized the prevention of overfitting to raw test accuracy when selecting parameters, and did so through rigorous empirical testing of model performance (in both train and test accuracy) when altering parameters. We set a goal to achieve a model with more than a 70% test accuracy and less than a 5% difference in test and train accuracies.

We first determined the minimum number of words the LSTM could consider without compromising accuracy. Allowing too many features could contribute to overfitting, with the model considering extremely uncommon and unmeaningful words as strong indicators of either side. A peculiar example of this was how our initial model, which considered a large 20000 word vocabulary, interpreted the word “cat" as a strong indicator of pro-choice affiliation. Appending various amounts of the word “cat" to the blatantly pro-life phrase “God and the holy spirit love babies" transformed the tweet from holding a strong pro-life opinion (score of 0.9992) to a strong pro-choice opinion with 8 “cat"-s appended (score of 0.0002).

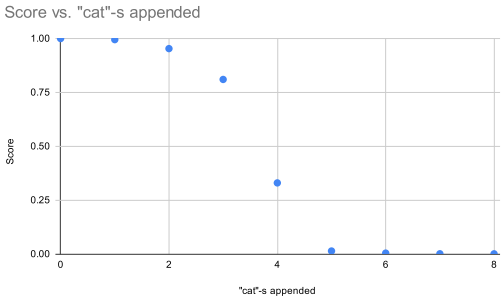


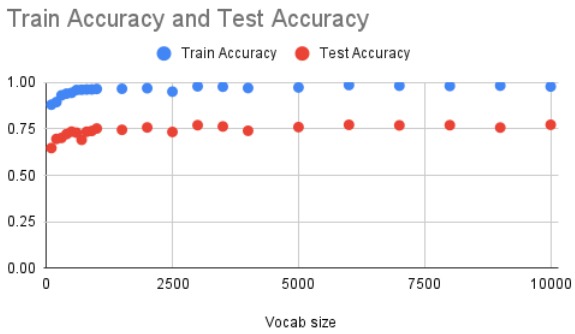
Figure 5: The effect of appending “cat"-s to a pro-life statement.

Aside from the difference of test and train accuracy, we also gauged model performance and overfitting with six custom phrases:

1. “Women’s rights are so important" (pro-choice)
2. “my body my choice, the government should keep their hands off women" (pro-choice)
3. “God loves babies thank god" (pro-life)
4. “We have cats, cats are really cute and fuzzy"
5. “Potatoes"
6. “According to all known laws of aviation, a bee should not be able to fly."

An accurate, non-overfit model would score (1) and (2) towards the pro-choice side (i.e., closer to 0) and (3) towards the pro-life side (i.e., closer to 1), but (4-6), being unrelated to training data and having no relation to abortion or Roe v. Wade, should not be scored extremely close either extreme (ideally at around 0.5). While initial, unregularized model scored the first 3 samples close to their true labels, it also confidently scored (4), (5), and (6) extremely close to either 0 or 1.

Before regularization, we experimented to find the minimum number of words the LSTM could consider without compromising accuracy. As shown below, we estimated that 3000 words was the minimum vocabulary size for which accuracy was not diminished.



Next, we experimented with applying L1 and L2 regularization with a variety of regularization constants. We aimed to find a regularization method and coefficient that minimized the difference between train and test accuracy while sacrificing the least amount of test accuracy. With L1 regularization, we found that producing a test/train accuracy difference less than 5% required a test accuracy less than 70%. See figure 6. Additionally, decreasing the constant slightly resulted in dramatic jumps in test/train accuracy difference. These unpredictable results may be attributed to how L1 regularization encourages sparsity (i.e., ignoring certain features), and so the sudden inclusion/exclusion of certain irrelevant words may lead to dramatically more/less overfitting.

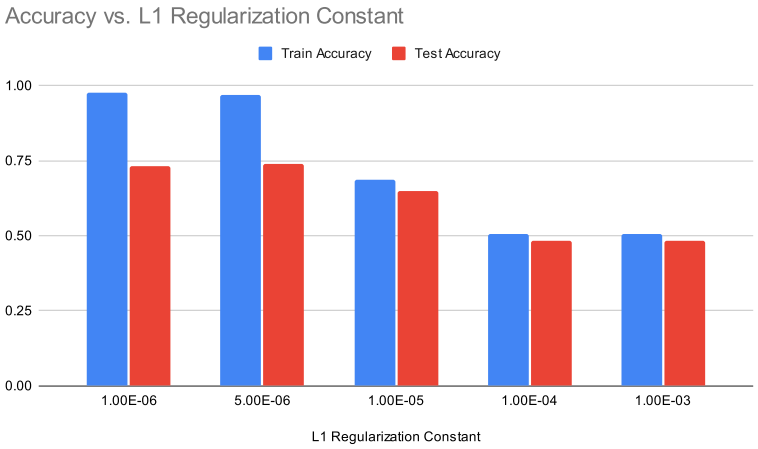


Figure 6: Accuracy vs. L1 Regularization Constant

When experimenting with L2 regularization (which does not reward sparseness), we found that a regularization constant of 8×10^{-5} produced the desired results of (1) a test/train accuracy difference less than 5% and (2) a test accuracy greater than 70%. We also experimented with combining L1 and L2 regularization, but found that both test/train accuracy difference and raw accuracy performed worse with the combined regularization than with each individual regularization.

Finally, we experimented with altering the dropout rate. But because we were using a single layer LSTM, we expected this to have no effect on the output, as there are no hidden connections between layers to ignore. Our experiments confirmed this, with dropout rate having no visible impact on the test or train accuracy (Figure 8).

After this experimentation, we chose to let our single-layer LSTM to consider a vocabulary size of 3000, incorporate L2 regularization with a coefficient of 8×10^{-5} (no L1 regularization), and have no dropout rate.

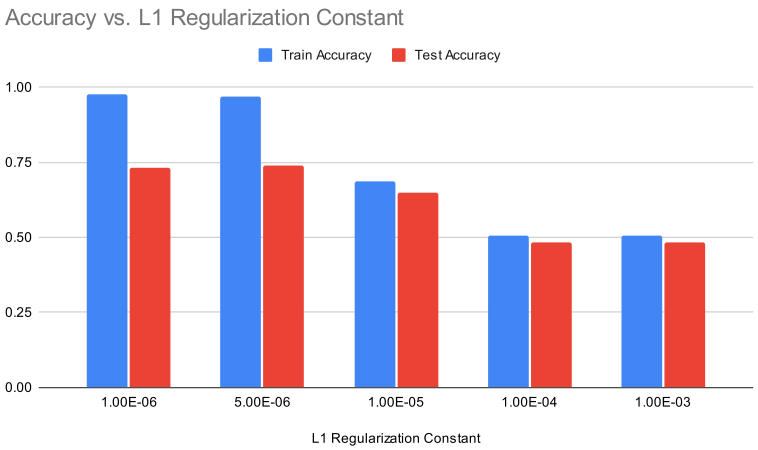


Figure 7: Accuracy vs. L2 Regularization Constant

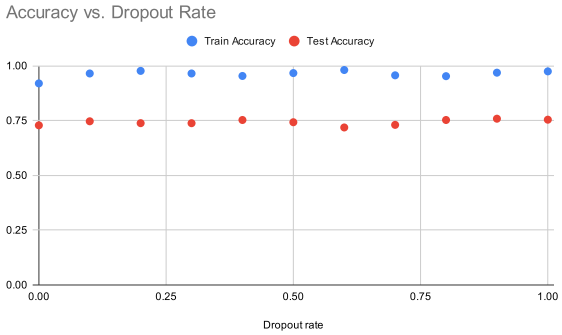


Figure 8: Accuracy vs. L2 Regularization Constant

4 Results

Recall that the original model suffered from excessive polarization. That is, the predictions it would make would be "aggressive" towards one end or another. This is indicative of overfitting since the model is not adapting to general patterns in input data.

In our experiments, we attempted to ameliorate this problem by decreasing the vocabulary size, and introduced L2 regularization to promote generalization. Let us observe the final effect of these changes.

The first noticeable change occurred during training. For the original model, the final training accuracy was 97.95%, and test accuracy was 73.6%. This corresponds to a gap of roughly 25% between the train and test dataset. For the new model, training accuracy levelled off at 83.41%, and test accuracy at 77.74%. Notice that the training and test accuracies have a much smaller gap between them (5%). Furthermore, performance on the test dataset increased by 4% between the original and new model. This is all indicative that the model has learned more general features about the dataset.

Prompt	Output
women's rights are so important	0.04714859277009964
God loves babies thank god	0.995335042476654
We have cats	0.030222343280911446
Potatoes	0.5644868016242981
According to all known laws of aviation, a bee should not be able to fly.	0.8318601846694946

Figure 9: Regularized model performance on custom prompt.

```
Score/Probability Pro-Life:
women's rights are so important                                          0.2429903745651245
my body my choice, the government should keep their hands off women      0.34514573216438293
God loves babies thank god                                               0.7602097988128662
We have cats, cats are really cute and fuzzy                             0.42585113644599915
Potatoes                                                                 0.5442848205566406
According to all known laws of aviation, a bee should not be able to fly. 0.4795265197753906
```

Figure 10: Regularized model performance on custom prompt.

For this small sample of potential tweets, note that the polarization problem is almost entirely absent. While the obvious pro-life and pro-choice tweets are labelled as such, the unrelated tweets have a score closer to the middle. This indicates that this model is not as overfit as the original one.

To evaluate model performance further, we tested it against a different dataset of Roe v. Wade tweets. (Figure 10). Notice that while the model does not perform as optimally (particularly for identifying pro-choice tweets), we still have a more even distribution than the original model. This indicates that we may need more data and less regularization to allow the model to extract more patterns from the dataset. Finally, we tested the model on a dataset of unrelated tweets pertaining to COVID-19 (Figure 11)

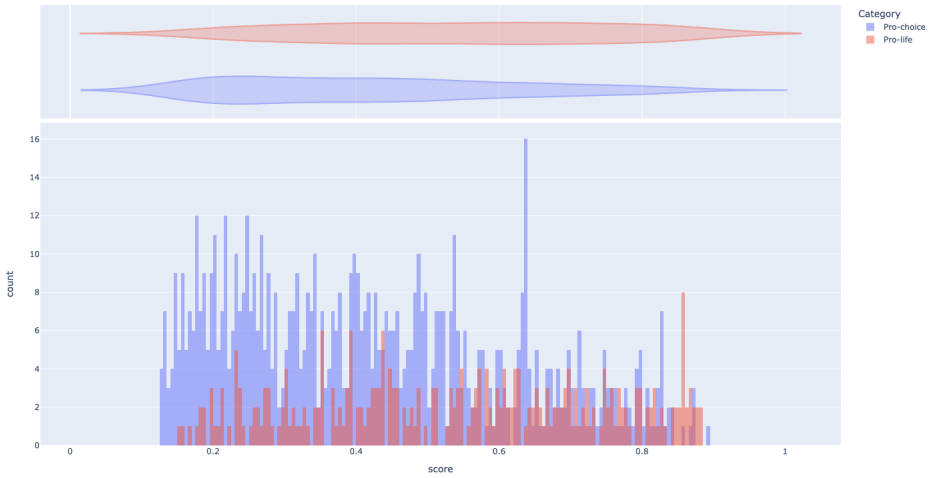


Figure 11: Model performance on adjacent Roe v. Wade tweet dataset

We see that the model outputs probabilities largely clustered around the center. This is more evidence that the model is no longer overfit, and is outputting reasonable probabilities based on the input data. Incidentally, notice that the mean of this distribution skews to the side of "pro-choice". It is unclear whether this is just a quirk of the model, or if it can tell us about the commonalities between COVID-19 and Roe v. Wade tweets.

5 Conclusion and future work

Through experimentation with model parameters, we trained an LSTM that was able to classify Roe v. Wade tweets with reasonable accuracy without overfitting to the training dataset. In this sense, we achieved the goal that we had originally set out for ourselves for this project. However, there are still some things to make mention of.

The first is that the performance of the model is still less than stellar. We believe that significanty more text data would need to be provided to significantly improve model performance. Furthermore, transfer learning approaches and Transformer-based models could potentially be superior for this problem as they are able to better capture temporal context. Future work may include fine-tuning such kinds of models, and observing how well they adapt to this particular problem.

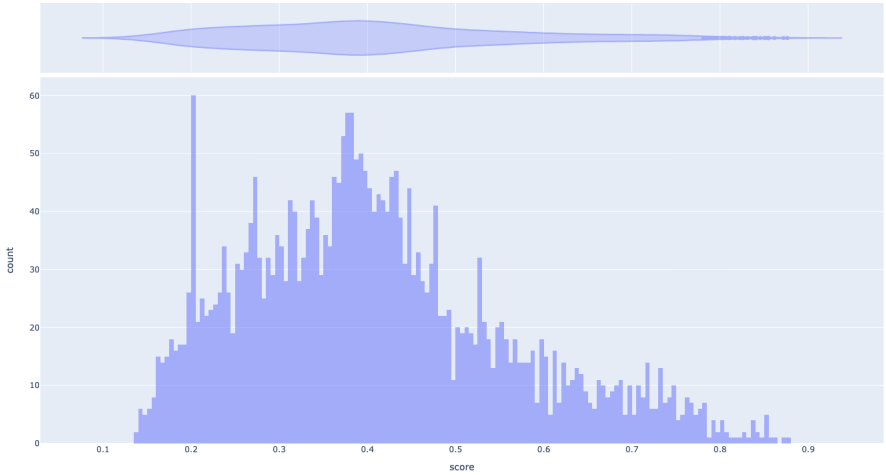


Figure 12: Model output histogram for unrelated (COVID-19) tweets

Of further interest is what the tradeoffs are between training a custom LSTM in place of using a fine-tuning approach on an existing model. While creating our own LSTM allows greater control over the network architecture, fine-tuning an existing language model has the benefit of reducing the computational demand, and potentially the quantity of data required. Depending on the use case, it could be useful to investigate which kinds of methodologies are most appropriate for particular situations.

6 Broader impacts

Training opinion classification models to generate accurate opinion distributions is the first step to creating data-fitted bounded confidence models. Such models would more accurately reflect real-world opinion spread and would let us study and simulate the mechanisms underlying online political polarization, echo chambers, and the spread of information. Our work is one small step towards creating accurate opinion dynamic models that may one-day aid in policy and social media design solutions that mitigate the polarizing effect of social media.

7 Code

Here is a link to our Github repository: https://github.com/kerrlya/XNNs

References

Breuel, T. M. (2015). Benchmarking of lstm networks. *arXiv preprint arXiv:1508.02774*.

Brooks, H. Z. and Porter, M. A. (2020). A model for the influence of media on the ideology of content in online social networks. *Physical Review Research*, 2(2):023041.

Chen, G., Su, W., Mei, W., and Bullo, F. (2020). Convergence properties of the heterogeneous deffuant–weisbuch model. *Automatica*, 114:108825.

Dupond, S. (2019). A thorough review on the current advance of neural network structures. *Annual Reviews in Control*, 14(14):200–230.

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., and Schmidhuber, J. (2017). Lstm: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10):2222–2232.

Hu, Y., Huber, A., Anumula, J., and Liu, S.-C. (2018). Overcoming the vanishing gradient problem in plain recurrent networks. *arXiv preprint arXiv:1801.06105*.

Odhiambo, C. O., Saha, S., Martin, C. K., and Valafar, H. (2022). Human activity recognition on time series accelerometer sensor data using lstm recurrent neural networks. *arXiv preprint arXiv:2206.07654*.

P., D. K. (2020). Covid19 tweet dataset. *https://www.kaggle.com/datasets/imdevskp/corona-virus-report*.

SurgeAI (2022). Surgeai abortion tweet dataset. *https://www.surgehq.ai/datasets*.