Detecting Climate Denial on Social Media

Haley Johnson
STATS 315 Final Project

Introduction:

Despite widespread scientific evidence that human activity has caused the impending climate crisis, nearly 20% of Americans deny the existence of anthropogenic climate change (Pew Research Center, 2019). Climate denial, often accompanied by pseudoscience and misinformation, runs rampant on social media (Treen et al., 2020). This project seeks to develop a flexible neural network architecture that can accurately and reliably detect tweets that express climate denial.This model is not a full-fledged content moderation solution. Instead, it is intended to be a tool that can flag potentially harmful content for further review, ideally by trained content moderation specialists or by individuals with subject-area expertise. This work represents an incremental step towards combating climate denial on social media.

Dataset:

This project uses a data set of over 40,000 Tweets about climate change. The data set was originally created by researchers at the University of Waterloo. Tweets were collected between April 27, 2015, and February 21, 2018. Each tweet's sentiment was independently reviewed by three annotators; only tweets where the annotators made a unanimous decision were included in the final data set. The four sentiment labels are:

- **1:** Denies the existence of anthropogenic climate change (9.08% of data set)
- **2:** Express neutral beliefs on climate change (17.56%)
- **3:** Express support for the existence of anthropogenic climate change (52.25%)
- **4:** Shares factual information about climate change, does not express support or denial (21.11%)

It is not clear what sampling procedure, if any, was used when the original researchers collected data from Twitter. Therefore, they may be bias in the data set and models trained on it could suffer from low generalizability.

Methodology:

Working with Tweet data comes with several challenges. First, tweets are limited to just 280 characters. In this data set, the median length after preprocessing was 115 characters. Second, tweets are highly contextual; 9.68% of data points begin with an at-mention towards another user, indicating that they're part of a longer conversation. The text of the tweets it self may not encapsulate all the necessary information and context to confidently label it's sentiment. Likewise, short texts are notoriously difficult

to apply NLP techniques to (Sakor et al., 2019), and these challenges are compounded by the nature of conversation on social media.

Moreover, there is a large class imbalance in the dataset. This poses a significant challenge because the class this project is more interested in accurately classifying — climate denial — makes up less than 10% of the dataset. To address this, each model was evaluated on two training sets: one that preserved the class imbalance in the original dataset and one that downsampled three majority classes to achieve equal representation (i.e. each class is 25% of the training set). The balanced training set generally yielded better results than the unbalanced set and **all models and results reported here use the balanced, downsampled training set.** The validation and testing datasets were not downsampled and contain the class distribution found in the original dataset. Consequently, achieving high accuracy is a more difficult task on the training dataset.

Decisions about how text data is represented can significantly influence the quality of predictions made by deep learning models. This project utilizes three different techniques to capture semantic features: GloVe embeddings, BERT embeddings and n-grams.

Baseline Model:

A logistic regression model was fit to serve as a simple baseline to evaluate more complex architectures against. Data was one-hot encoded before being fed into the regression model.

This baseline achieved just 21.56% accuracy on the validation dataset and 22.10% accuracy on the training dataset. A closer inspection of the model's predictions revealed that it labeled 94.77% of examples as neutral,' despite this class making up just 17.56% of the dataset.

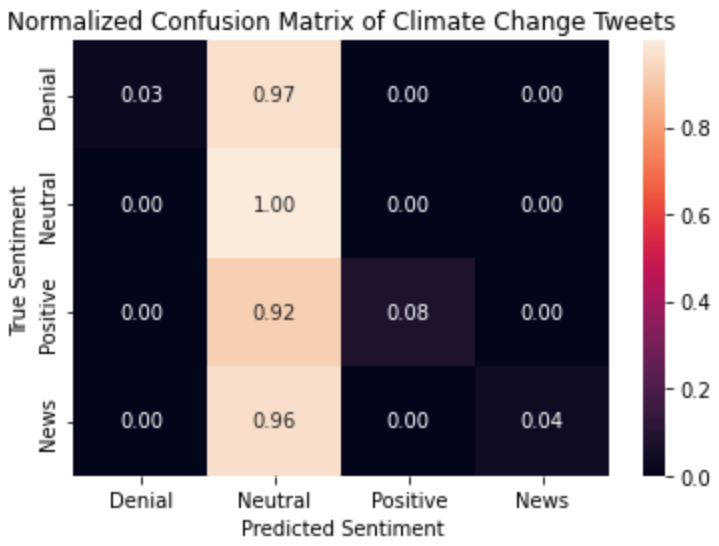


Figure 1: Confusion matrix of test set predictions from logistic regression baseline model

Glove Embeddings:

The first three models this project used leveraged GloVe embedding. GloVe, an acronym for Global Vectors for Word Representation is an unsupervised algorithm that leverages global word co-occurrences in a corpus to produce vectors with a meaningful and informative sub-structure (Pennington et al., 2014). I utilized vectors that were pre-trained on Twitter data to construct an embedding layer. Tweets were tokenzier before being inputted into my models

<u>Model 1:</u>

This model utilizes 5 stacked long-short-term-memory layers and two dense layers. This model achieved 36.75% accuracy on the validation set and 37.42% accuracy on the training set.

Notably, during training the model had one epoch with exceptionally high loss. The loss returned to normal after this epoch.

```
Epoch 1/100
84/84 [==============================] - 192s 2s/step - loss: 1.3894 - accuracy: 0.2463 - val_loss:
Epoch 2/100
84/84 [==============================] - 187s 2s/step - loss: 55192186880.0000 - accuracy: 0.2550 -
Epoch 3/100
84/84 [==============================] - 178s 2s/step - loss: 1.3916 - accuracy: 0.2800 - val_loss:
```

Figure 2: Results from model 1's first, second and third training epochs

This also skewed the appearance of the training curves.

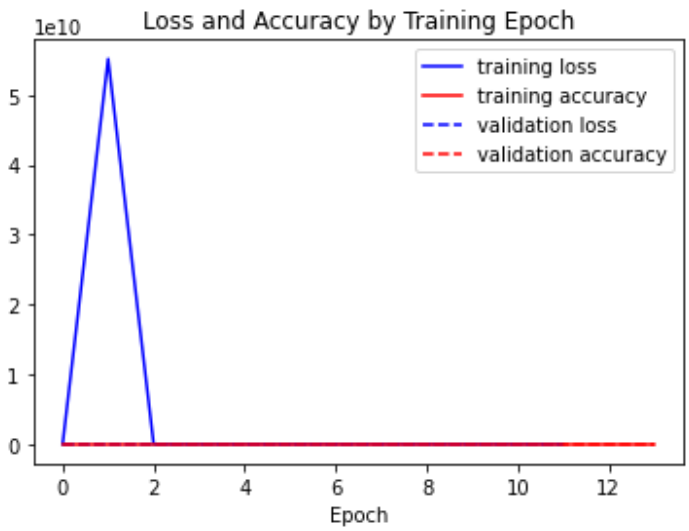


Figure 3: Training curves for model 1

The model predicted most tweets to be 'denial,' 'positive,' or 'news.' It rarely predicted tweets to be 'neutral.' Compared to the baseline, however, this model made better use of all the classes available.

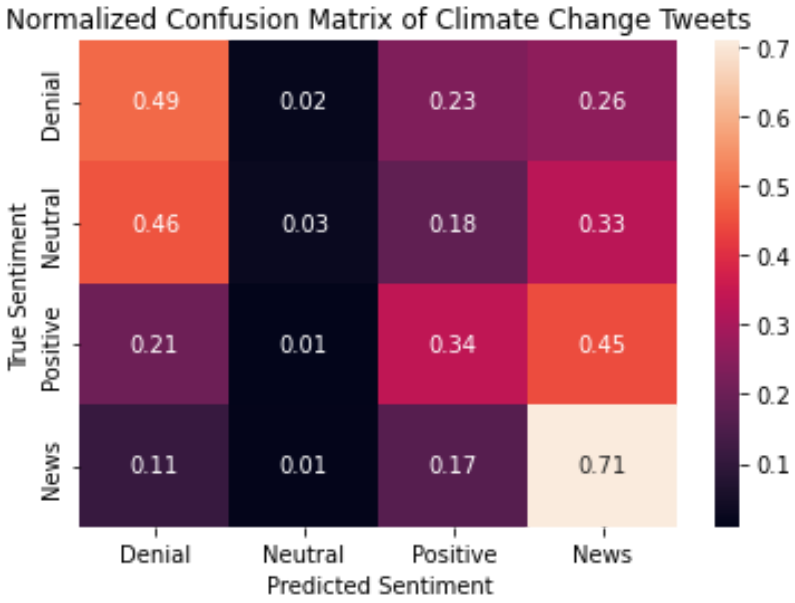


Figure 4: Confusion matrix of test set predictions from model 1

Model 2:

This model's architecture was made up of an embedding layer, a bidirectional gated recurrent unit, a global average pooling layer, and then three dense layers. It achieved 40.07% accuracy on the validation data and 40.86% accuracy on the test data. Similar to

the first model, this model was the best at correctly predicting if a tweet belonged to the 'news' class. It's accuracy at predicting climate denial tweets was about as good as randomly guessing — climate denial were only correctly classified 23% of the time.

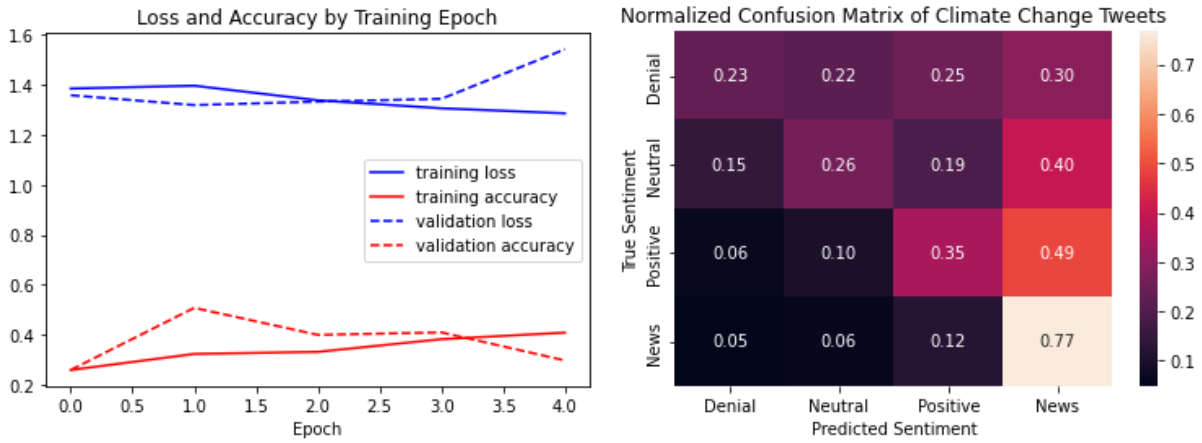
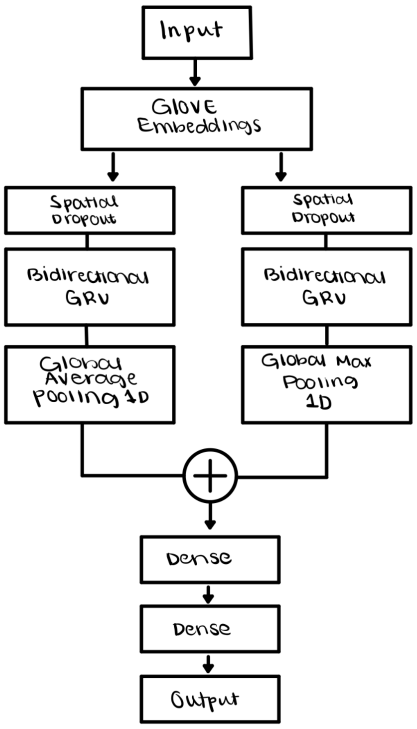


Figure 5: Training curves and confusion matrix of test set predictions from model 2

Model 3:



This non-sequential model builds off the improvement made by the second model, particularly the use of bidirectional layers with gated recurrent units. It also leverages spatial dropout. Spatial dropout is commonly applied to convolutional neural networks, but can also be applied after an embedding layer to randomly remove word from the embedding matrix (Gal & Ghahramani, 2015). Pooling effectively compresses the information in the sentence into a more simplified representation. This non sequential model combines the results of two blocks: one that applies global average pooling to the outputs of the bidriectional gated recurrent units and one that applies global maximum pooling to these outputus. Then, it passes the sum into a series of dense layers.

Figure 6: Flow of inputs through model 3

Overall, this model achieved 45.77% accuracy on the validation data and 45.53% accuracy on the training data. It also makes predictions in all of the categories available, although tweets were commonly mislabelled as having positive sentiment.

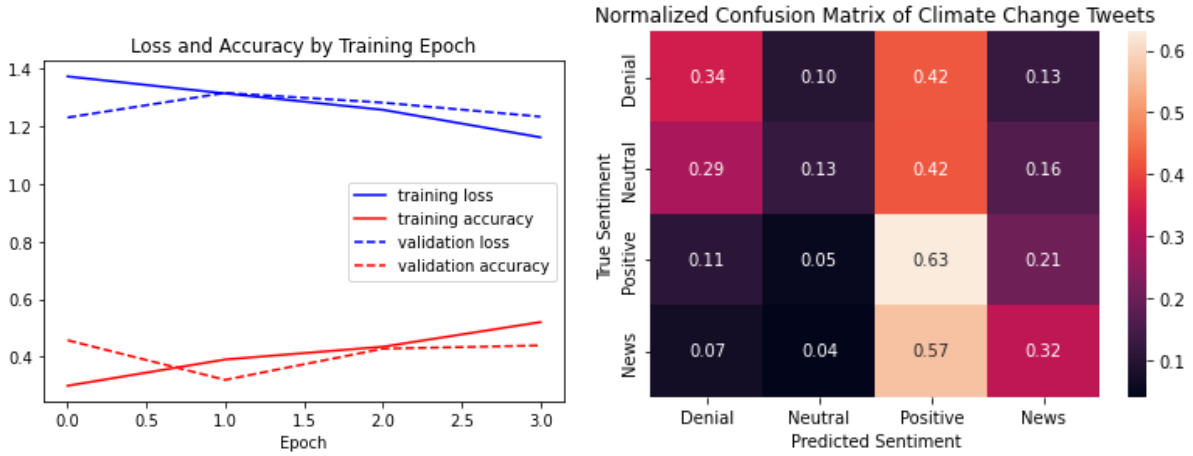


Figure 7: Loss curves and confusion matrix results test set predictions for model 3

BERT Embeddings:

Bidirectional Encoded Representations from Transformers, better known as BERT, are the current state-of-the-art method for creating word embeddings (Devlin et al., 2019). Although BERT was originally trained on lengthy text from Wikipedia, it has also been successfully used in models that work with short texts from social media (see D'sa et al., 2020, Kaviani & Rahmani, 2020, and Hamid et al., 2020)

Model 4:

This model feeds BERT embeddings into a dense neural network. It uses a pre-trained embedding layer hosted on TensorFlow Hub and contains the weights that were released by the original BERT authors. Since the texts in this dataset are relatively short, each tweet was fed into the embedding as one "sentence," even if the original tweet contained multiple sentences.

The outputs from the BERT embedding layer were passed into a dense neural network. There were six hidden dense layers and a final output dense layer. This model achieved 43.89% accuracy on the validation set and 44.16% accuracy on the training set. This model correctly classified 'denial,' 'positive,' and 'news' the majority of the time, but performed poorly at detecting neutral sentiment.

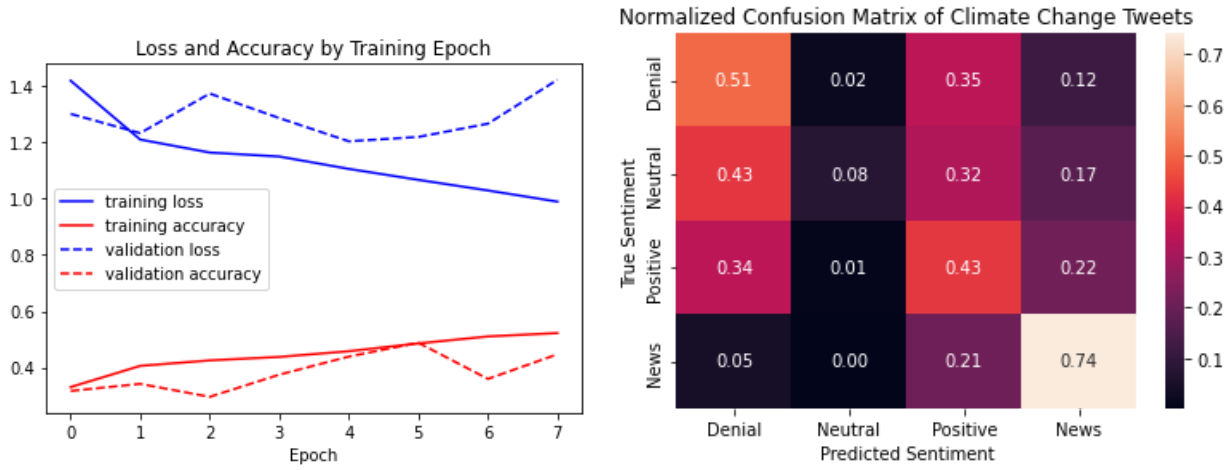


Figure 8: Training curves and confusion matrix of test set predictions from model 4

N-Grams:

Finally, this project developed two models that represent the tweets as n-grams. All tweets were passed into sklearn's CountVectorizer. The vectorizer was limited to the 5,000 most common unigrams, bigrams, and trigrams in the text. No embedding layer was used in either model.

Model 5:

This model did not use any information about the order of features of the text. Data was fed into a dense layer with 256 neurons and then the model alternated between dense layers and 15% dropout layers. Heavy dropout was used to combat overfitting.

This model achieved surprisingly high accuracy. It classified 56.11% of validation examples and 56.68% of training examples correctly. For each class of training examples, the most common class that was predicted was the true label (i.e. it didn't mislabel 'news' and 'positive' most of the time).

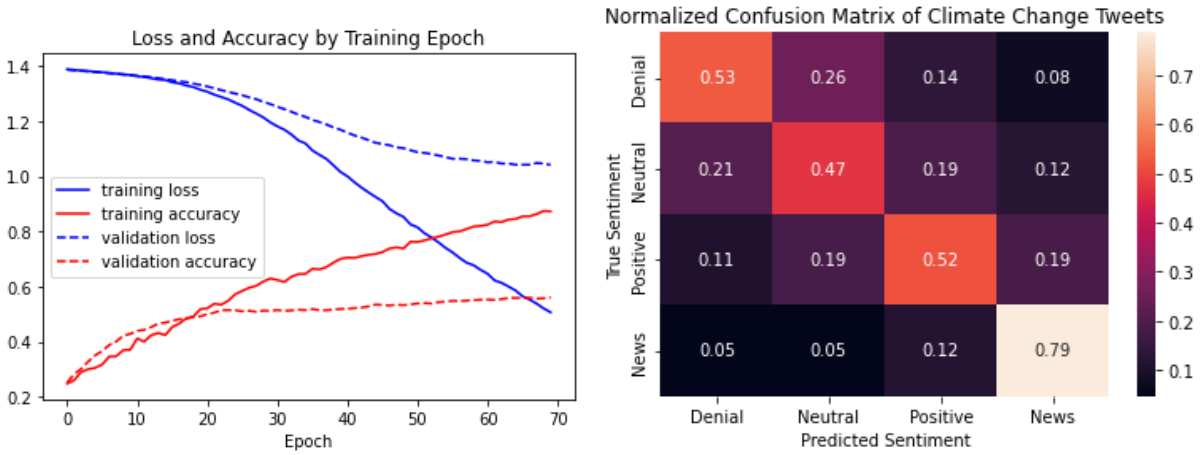


Figure 9: Training curves and confusion matrix of test set predictions from model 5

Model 6:

The final model this project examined passed the count vectorized data into a recurrent neural network. This recurrent neural network used two stacked gated recurrent units, the second of which had 10% recurrent dropout. Then, the outputs of the recurrent units were passed into a series of alternating dense layers and 10% dropout layers, before being passed into a final output dense layer.

This model achieved 20.35% accuracy on the validation data and 19.73% accuracy on the training data. This model performed very poorly and only predicted one class.

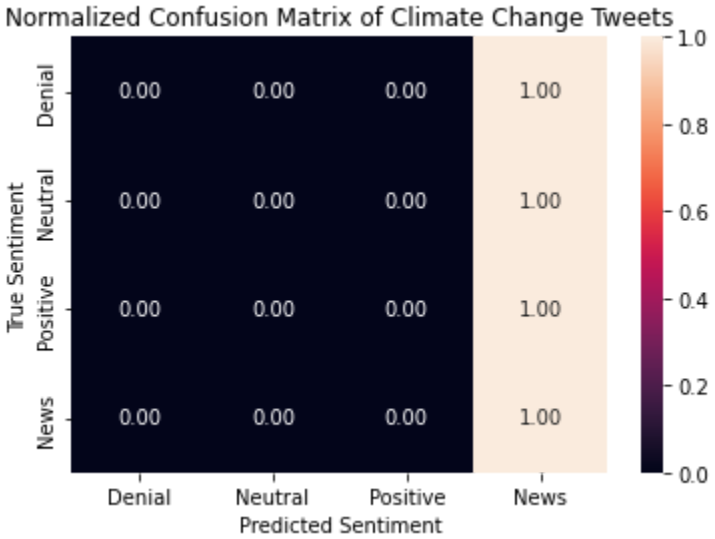


Figure 10: Confusion matrix of test set predictions from model 6

Summary:

Model Name	Architecture Description	Validation Accuracy	Test Accuracy	Results
baseline	Logistic regression model Fitted on one-hot encoded vectors	21.56%	22.10%	Predicts that almost all examples are 'neutral' Correctly classified just 3% of climate denial tweets
model 1	Recurrent neural network using GloVe embedding layers, five LSTM layers and two dense layers Fitted on texts that had been tokenized	36.75%	37.42%	Rarely predicted 'neutral' but made better use of all classes than baseline Correctly classified 49% of climate denial tweets
model 2	Recurrent neural network using GloVe embedding, one bidirectional GRU layer, global average pooling layer, and two dense layers Fitted on texts that had been tokenized	40.07%	40.86%	Predicted all classes, still had the strongest performance on news Correctly classified 23% of climate denial tweets. The most common misclassification of climate denial tweets was as 'news' (30%)
model 3	Non-sequential neural network using GloVe embedding layer. Utilizes 10% spatial dropout, two bidirectional GRU layers, global average pooling, and global max pooling. Combined pooling results and fed them into 3 dense layers (see figure 6) Fitted on texts that had been tokenized	45.77%	45.53%	Misclassified the majority of 'news' tweets as positive Correctly classified 34% of climate denial tweets. The most common misclassification of climate denial tweets was 'positive' (42%)
model 4	Neural network using BERT embeddings and 7 dense layers Fitted on texts that had been tokenized	43.89%	44.16%	Did not make much use of news class Correctly classified 51% of climate denial tweets. The most common misclassification of climate denial tweet was a 'news' (35%) Classification matrix suggests this model generally struggles to

				distinguish between ‘denial’ and ‘positive’
model 5	Dense input layer with 256 neurons, alternating dense layers and 15% dropout layers. Fitted on texts that had been count vectorized.	56.11%	56.68%	Made the best use out of all the classes available Correctly classified 53% of climate denial tweets. The most common misclassification of climate denial tweet was a ‘news’ (26%)
model 6	Two GRU layers, alternates between dense layers, and 10% dropout for 5 layers, final output dense layer. Fitted on texts that had been count vectorized.	19.73%	20.35%	Predicted that all examples were news. 0% accuracy on climate denial Tweets

Experiments:

The two best performing model were model 5, a dense neural network using count vectorized texts, and model 4, which used a BERT emedding layer followed by several dense layers. Not only did these two models achieve high overall accuracy, they also correctly classified climate denial tweets the majority of the time. The success of these two simple, dense architectures over models that tused more complex recurrent cells suggests that recurrent units may not perform well on this specific dataset, or that information about order isn’t particularly informative when making predictions.

The confusion matrix of the BERT model’s predictions suggest that the classifier has difficulty discriminating between tweets that express support for the existence of athropogenic climate change and tweets that deny it. For instance, 35% of climate denial tweets were misclasssifieid as being pro-climate change and 34% of pro-climate change tweets were misclassified as denial. Model 5, the dense neural network was much less likely to confuse pro and anti climate change sentiment. In model 5, climate denial tweets were much more likely to be misclassified as ‘neutral’ and tweet’s supporting climate change were more likely to be classified as ‘neutral’ or ‘news’ than denial. The differences between the neutral, news and positive classes are important. However, this project’s primary objective is being able to reliably detect climate denial. It is less important if the classifier confuses ‘positive’ and ‘neutral’ than if it confuses ‘denial’ and ‘positive.’ Thus, by examining where the classifier make mistakes, we can conclude that model 5 achieves the best overall performance on the task this project is interested in.

Conclusion:

The modest results this project achieved suggest it's difficult to definitively label a tweet as climate denial. This underscores the challenge of detecting harmful content for human moderators and algorithms alike — the lines between debate, healthy skepticism and outright denial are often thinner than we'd like them to be.

While downsampling improved the results of the models examined in this project, it also greatly reduced the size of the training set. The final training set had ~2671 tweets, compared to 34,887 tweets in the unbalanced set. It may be worthwhile to see if upsampling minority classes yields better results without hampering generalizability.

Future work should evaluate if using the context surrounding the tweet improves model performance. For example, many tweets in this dataset are replies. Could including the tweet that a user replied to lead to more accurate predictions? Moreover, this project used GloVe embeddings that were specifically designed for Twitter, but used BERT embeddings that were trained on Wikipedia and BookCorpus. Using a BERT embedding layer trained on social media data will likely yield better results and make comparisons between the models using BERT and GloVe embeddings more meaningful.

Works Cited

A. G. D'Sa, I. Illina & D. Fohr, BERT and fastText Embeddings for Automatic Detection of Toxic Speech, 2020 International Multi-Conference on: “Organization of Knowledge and Advanced Technologies” (OCTA), 2020, pp. 1-5, doi: 10.1109/OCTA49274.2020.9151853.

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. ArXiv, abs/1810.04805.

Hamid, A.F., Shiekh, N., Said, N., Ahmad, K., Gul, A., Hassan, L., & Al-Fuqaha, A. (2020). Fake News Detection in Social Media Using Graph Neural Networks and NLP Techniques: A COVID-19 Use-Case. ArXiv, abs/2012.07517.

Gal, Y., & Ghahramani, Z. (2015). A Theoretically Grounded Application of Dropout in Recurrent Neural Networks. NIPS.

Kaviani, M., & Rahmani, H. (2020). EmHash: Hashtag Recommendation using Neural Network based on BERT Embedding. 2020 6th International Conference on Web Research (ICWR), 113-118.

Pennington, J., Socher, R., & Manning, R. (2014).“GloVe: Global Vectors for Word Representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Pew Research Center. (2019). US public views on climate and energy: Democrats mostly agree the federal government should do more on climate, while Republicans differ by ideology, age and gender. ScienceDaily. Retrieved December 11, 2022 from www.sciencedaily.com/releases/2019/11/191125121009.htm

Sakor, A., Mulang, I.O., Singh, K., Shekarpour, S., Vidal, M., Lehmann, J., & Auer, S. (2019). Old is Gold: Linguistic Driven Approach for Entity and Relation Linking of Short Text. North American Chapter of the Association for Computational Linguistics.

Treen, K. M. D. I., Williams, H. T., & O'Neill, S. J. (2020). Online misinformation about climate change. Wiley Interdisciplinary Reviews: Climate Change, 11(5), e665.

Appendix

This appendix contains 4 code files. In the order they appear, they are:

- **final_project.ipynb:** this Jupyter Notebook contains code to train all the models used in this project
- **exploratory_analysis.ipynb:** this short Jupyter Notebook contains basic exploratory data analysis, including plots
- **helper_functions.py:** this python file contains code for functions I made to help me analyze, preprocess and plot my results. This program helps reduce the length and complexity of final_project.ipynb
- **process_text.py:** this python file contains code that preprocesses the dataset used in this project. The code normalizes the text and removes stopwords, non-English Tweets, non-ascii characters, emojis and other undesirable features from the text