Trump Tweets Generator , Natural Language Processing Final paper:

Submitted by : Bar Oren 206548752 ,Pnina yonayov 315670182,Dan Shklarnik 316555663

In this report, we will provide an overview of our work in building a Trump tweet generator. Our approach involved using a dataset of existing Trump tweets to train a language model capable of generating new tweets. The model can generate tweets based on user input or generate a completely random tweet.

Throughout the report, we will discuss the methods we used to build and train the model, as well as the tools and technologies we leveraged. We will also provide an evaluation of the system's performance and discuss potential areas for future development.

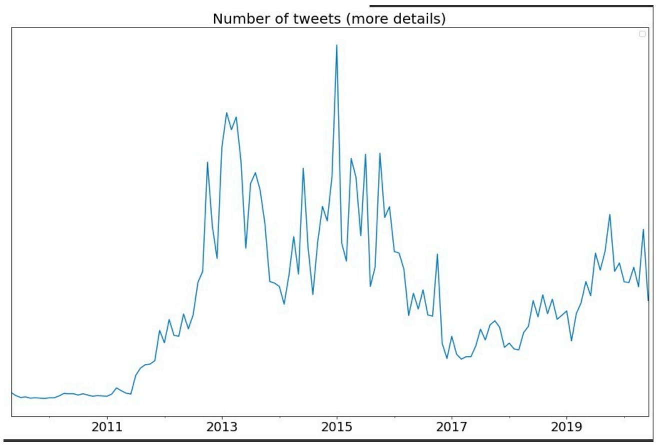
Our goal is to provide a comprehensive analysis of the work we have done, and to demonstrate the capabilities of our tweet generator system. We hope that our findings will be useful to those interested in natural language processing, machine learning, and social media analysis.

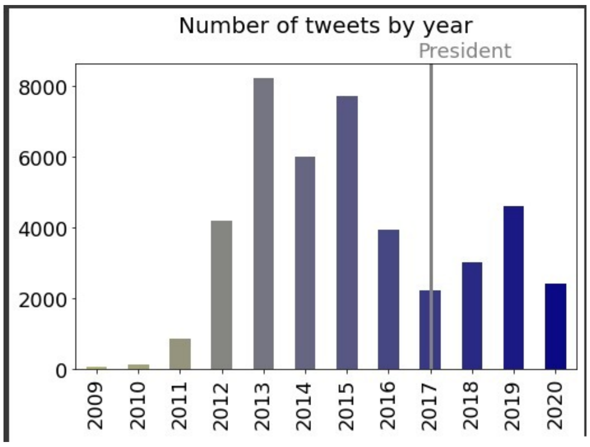
Data Exploration:

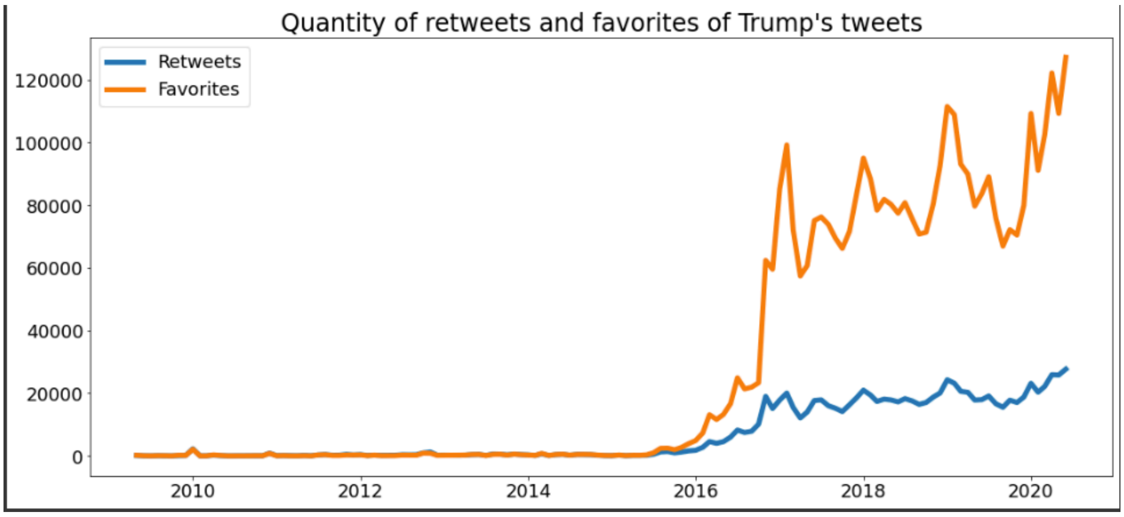
We used the data set: https://www.kaggle.com/datasets/austinreese/trump-tweets

Basic analysis of the data:

More information can be found on Trumps Tweet analysis.ipynb







Model Exploration:

During our model exploration, we opted to use the fine-tuning method with the GPT-2 language model and transforms. Fine-tuning involves training a pre-trained model on a specific dataset to learn the patterns and structure of the language specific to the task at hand. In our case, the task was to generate new Trump tweets based on a given prompt or a completely random tweet.

To fine-tune the GPT-2 model, we adjusted its hyperparameters and trained it on the dataset of existing Trump tweets. This enabled the model to learn from the patterns and structure of the language used in these tweets, allowing it to generate new tweets that closely match the style and tone of Trump's tweets. By adjusting the model's hyperparameters and fine-tuning it on a specific task, we were able to optimize the model for our desired output.

While our approach of using fine-tuning and transforms was effective in generating new Trump tweets that closely match the style and tone of the original tweets, it is important to note that the model sometimes generates random and nonsensical results as well. This can be due to a variety of factors, such as insufficient training data or inappropriate hyperparameters.

We monitored the performance of our model during the training process using a loss function, and the graph below indicates that the model did train and improve over time. However, it is important to note that even with optimized hyperparameters and adequate training data, language models are not always perfect in their output.
In conclusion, while our approach was effective in generating new Trump tweets, it is important to exercise caution and critical evaluation of the outputs generated by the model.

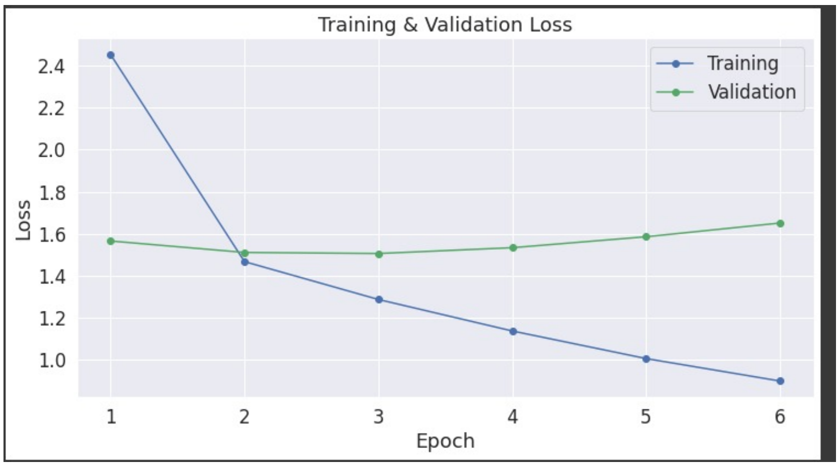
Additionally, we experimented with two different pre-processing techniques during training. In the first iteration, we removed only the links from the tweets, while in the second iteration, we performed full pre-processing, which included removing stop words, punctuation, other special characters, and more.
Interestingly, we found that the model trained on the pre-processed data with only links removed produced better results than the model trained on fully pre-processed data. This could be because removing too much of the text during pre-processing could potentially remove valuable information and context from the tweets, resulting in less accurate outputs.

Overall, our experimentation with different pre-processing techniques highlights the importance of finding the right balance between cleaning the data and preserving the valuable information within it. The metric chosen to evaluate the performance of the model was the average loss, which was calculated during training and validation. The goal was to minimize the loss during training, which indicates how well the model is learning to generate coherent and meaningful text based on the input.

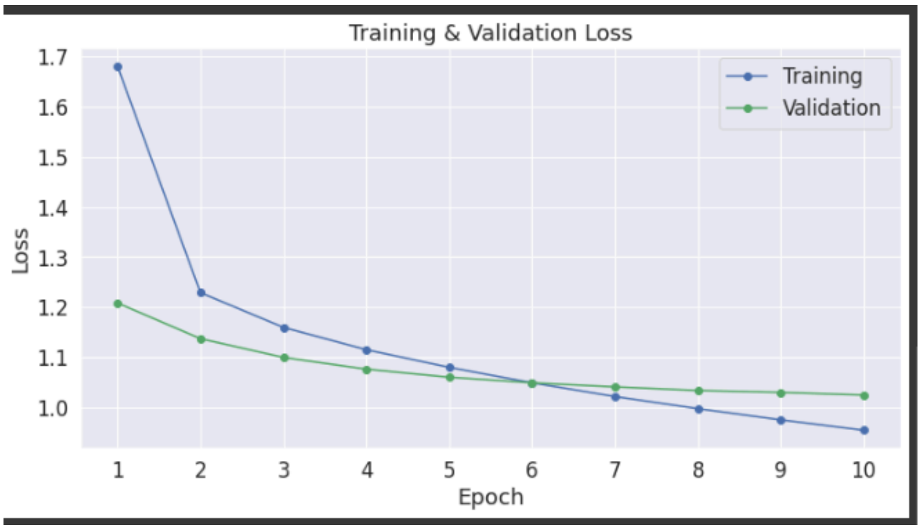
The results of the model were mixed. Some generated tweets were coherent and meaningful, while others were nonsensical and lacked context. According to the loss graph, the model seemed to train reasonably well, with the loss decreasing over time during both training and validation.
*The full code can be found on TrumpTweet.ipynb, the second attempt with more pre processing can be found on TrumpTweetpp.ipynb

Results:

In the beginning, we encountered an overfitting problem where the model was performing well on the training set but poorly on the validation set.



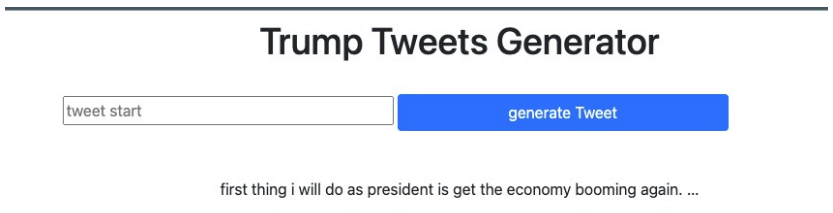
To address this issue, we implemented dropout regularization during the training cycle. By setting the dropout probability to 0.081, we were able to reduce overfitting and improve the generalization performance of our model. This allowed our model to achieve better results on the validation set and generate more coherent and realistic tweets.



Model evaluation test “this is crazy:”:
0: this is crazy @ billmaher has $700M to show up at the @ nytimes in his first two years. # trump
1: this is crazy this is happening and our country is in a financial mess. @ BarackObama is a weak & ineffective leader.
2: this is crazy but the media is trying hard they just aren't as sharp & unbiased as we used to be.
3: this is crazy @ AlexSalmond says Scotland will be taxed up to £300 a month.
4 this is crazy the number of illegal immigrants that the Border Patrol Agents are getting is far higher than the total number of the illegal immigrants who have committed crimes, including murder and that is a total FAKE NEWS story. I have been doing this for many years, and if they are afraid to report it, I will report it anyway. Not good. The Wall is being built!

Model implementation :

To demonstrate the capabilities of our model, we chose to implement it in a simple website. To build our web application, we utilized Flask and Pyngok. The website's backend was developed in Python, which allowed us to run the model from Google Colab and present the resulting tweets to the user. we used a model that was trained in Google Colab. This allowed us to leverage the power of Google's cloud infrastructure to run the model, which in turn made it possible for us to handle large amounts of data quickly and efficiently.



first thing i will do as president is get the economy booming again. ...

*simple website that we implemented the model in

potential avenues for further development that could be explored in the future:

Firstly, the dataset that was used for the project was quite large, but it contained many repeated tweets or tweets that were responses to other people. This made it possible for the system to learn the patterns and structure of tweets as well as Trump's responses. However, it may be beneficial to further clean the data to eliminate any unnecessary repetition or noise.

Secondly, the GPT-2 model used for the project required a lot of time, resources, and data to train effectively. Therefore, it might be worth exploring alternative models like RNNs (Recurrent Neural Networks) to achieve better results in a more efficient manner.

Furthermore, the team discussed the possibility of classifying the data so that the user could choose what kind of tweet they wanted to generate. This could include categorizing tweets as happy, mad, or specific, among other possibilities. By doing this, it would be possible to generate more accurate and tailored tweets that match the user's desired tone or message.

Lastly, the team discussed the possibility of adding more tweets from famous people to the dataset. This would give the user the ability to choose which person's tweets they want to generate. For instance, the user could select Donald Trump, Barack Obama, or Elon Musk, and the system would generate tweets in their style. This would provide a more diverse range of content and would allow the user to experiment with different styles and tones. Another possibility would be to incorporate user feedback into the model training process. This would allow the system to learn from user interactions and improve the quality of generated tweets over time.

Another possibility would be to integrate the model into a more user-friendly and visually appealing web application, which could make it more marketable to potential users. Finally, the team could explore incorporating more advanced features such as image recognition or speech-to-text conversion (from videos for example) to generate more complex and diverse tweets. This would make the system more versatile and allow it to generate tweets in response to a wider range of input types.