Advanced Machine Learning, Data Mining, and Artificial Intelligence
Final Project Report
Dec 16, 2020

Topic: AI Chatbot using Multi-head Attention Transformers and GPT2

Team Partners: Vanessa Moody, Lalitanjali Bondili

Table of Contents:

---

1. Motivation and Problem Statement:

Transformers are a recent and promising neural network architecture which have promoted significant breakthroughs in NLP research. The signature architecture of transformers is their use of “attention” heads rather than recurrent network components for modelling sequential data with temporal correlations. This architecture has proven adept at handling tasks such as question answering, sentiment analysis, neural translation, text summarization, and many more. However, as a class of machine learning models, they require onerous amounts of training data and compute to perform reasonably or even achieve grammatical correctness on sequence to sequence tasks; for example, GPT-2 trained on approximately half of a trillion word tokens and used 256 TPUs, and XLNet racked up $245,000 worth of expenses using 512 TPUs. Luckily for us, many of these models are open sourced. Our motivation for this final project was to leverage open sourced transformer architectures and compare and contrast their performance in different domains to a transformer with multihead attention we built from scratch.

We identified two problem areas to explore in training our respective transformers. The first problem area we investigated was building an artificially intelligent chatbot. An AI chatbot is a piece of software that uses machine learning methods rather than a scripted ruleset to communicate with humans using natural language. While AI chatbots have the potential to be incredibly useful for helping users learn more about a specific domain, this is a domain which has peaked interest -- and critique -- very recently, due to the Microsoft chatbot “Tay”. Tay was a machine learning chatbot that was released on Twitter in 2016 and given free reign to interact with users, with the unfortunate end result of users “biasing” Tay to tweet insensitive messages and even hate speech. The age old adage of “garbage in, garbage out” certainly held true in Tay’s case, and over the course of less than 24 hours Tay was removed from Twitter. We were interested in Tay’s capacity to essentially reflect the data it was trained on, and wanted to explore ways to use this idiosyncracy for good by training a chatbot transformer on subreddit data. Our heuristic is very different from the one used to train Tay, as the transformer *samples* data rather than interacting with users on the platform, so users cannot bias the transformer training. Following training, we can ask questions of our transformer chatbot, to examine the chatbot’s question/answering capabilities and explore differences between transformers trained on different subreddits.

The second problem area we decided to investigate in the scope of our final project was text summarization. Text summarization involves succinctly stating the most important content given a greater amount of text, and we felt was the next step up in difficulty for testing our transformer models. Specifically, while AI chatbots are difficult to assess, as even “wrong” answers could be interpreted as sarcasm, text summarization output can be assessed based on how apt the summarization was of the original content. In the scope of our final project, we created both a transformer from scratch and a transformer finetuned on the existing GPT-2 small model, trained on different subreddits comprising two major subjects (news and science) to produce AI chatbots, compared and contrasted abstractive summarization using a finetuned GPT-2 model to extractive summarization, and evaluated the final results.

2. Dataset:

We used the PRAW python library to extract data from multiple subreddits for training our transformer models. We looked at a variety of different subreddits for training, and eventually decided to use the askscience, science, news, and worldnews subreddits for a few different reasons.

First, we needed to extract enough data for training our transformers (on the order of megabytes), so we only looked at the “top”, or most popular, subreddits. After exploring threads from several top reddit accounts, we realized an additional stipulation we needed to take into consideration was the frequency of media or links to external content in different subreddits. Since this data cannot be picked up on by our transformers, training transformers on comments and replies generated from this data would decrease the overall performance of the transformer. We eyeballed the top 15 subreddits to determine which had high frequencies of image media or links to external content, and ruled out the Funny, Gaming, Aww, Pics, Music, Videos, and Movies subreddits as a result. We then decided to focus on Science and Worldnews, which were more of interest to us, and augmented them with Askscience and News in the interest of collecting more data for training.

We scraped datasets from these subreddits using a variety of different approaches according to the data formats required by our respective transformer models. Most of these datasets were too large to include at larger than 10 MB, and we’ll describe those in detail in the following sections. We were able to include the datasets used for finetuning the GPT2 chatbot model in the zip file.

3. Our Approach:

We delimited the final project into the following high-level sections:

1) Building an AI chatbot using multi attention-head transformers and transfer learning with GPT2
2) Extractive summarization using spaCy and abstractive summarization using transfer learning with GPT2
3) Data exploration using LDA for topic modeling, cosine similarity distances to evaluate similarity across posts, and PCA to visualize word embeddings

3.1 Building an AI chatbot

Conversational chatbots exist in two main forms. The first one is the traditional dialog system which is very robust in executing task specific commands (eg: A restaurant reservation or customer service chatbot). These systems do not have the capability to reply to uninformed conversational utterances and can only reply to utterances in their specific domain. These traditional chatbots are built mainly using hard-coded templates and language rules.

The second type of a chatbot is an AI chatbot which is non-domain specific as it is able to hold a regular arbitrary conversation. The chatbot should be able to have a conversation that is engaging along with the capability to use large vocabularies and reply with precision to a broad range of conversational topics. We intend to build an AI chatbot and further train this model on specific subreddit knowledge bases(namely Science, AskScience, News and WorldNews) so that it can also reply coherently when conversing on such subjects.

Our ambition is to build a chatbot that should not be distinguishable from a real human. To accomplish this task, we built a state-of-the-art transformer model which works by utilizing the multi-head attention mechanism and will be trained on large amounts of data to learn the process of generating relevant and grammatically correct responses.

To further improve the results of this chatbot based on multi-head attention, we used the GPT2 language model pre-trained on a very large corpus of text which is capable of generating long stretches of contiguous coherent text. We implemented transfer learning to finetune the pre-trained 124 M GPT-2 model on our subreddit knowledge bases. GPT-2 is an improvement over the vanilla attention transformer model with respect to its architecture/structure, and also sees large performance improvements due to the large database of over 8 million webpages and computation power of hundreds of TPUs used to produce the pretrained model. We use the paradigm of fine tuning on both the smaller reddit and cornell movie dataset using the fine tuning API supplied by https://github.com/nshepperd/gpt-2.

Finally, we conducted a detailed analysis of how the GPT2 model performs better than the vanilla transformer model with multi-head attention on the trained dataset by comparing replies to the same utterances and assessing the quality of the replies.

3.2 Extractive and abstractive summarization

We address the problem of summarization of comments received to a subreddit post in two directions:

1) **Extractive summarization** extracts and concatenates important spans of the reddit comments. This is done by calculating the importance of each sentence based on the number of appearances of important keywords and their calculated normalized weights.

Here each comment is tokenized using scaCy's language model. This summarization model is only capable of picking the most important sentences from a document corpus but is not capable of generating new sentences to form a summary.

2) **Abstractive/Inferential summarization** generates a summary by paraphrasing the subreddit comments. We have fine-tuned a pre-trained GPT2 model on the scraped reddit dataset using the standard language model objective to leverage the powerful text generation capability of the model. This abstractive approach is much more demanding as the model must be able to represent semantic information of the source text and then use this semantic representation to generate a paraphrase. However, the model may gain the ability to make a creative use of words or ability to make inference from the source text.

3.3 Data Exploration

To better understand characteristics of our dataset, we additionally applied LDA for topic modeling of our dataset, cosine similarity distances to evaluate similarity across reddit posts, and PCA to visualize word embeddings given all tokens identified within the dataset.

We first used LDA for topic modeling, a branch of unsupervised natural language processing which is used to represent a set of documents within several distinct topics that can best explain their underlying information. We used LDA to generate different sets of topics to examine the dataset used for GPT2 chatbot training.

We then moved on to adapting the cosine similarity metric for examining similarity across subreddit posts with respect to their high level topics and the content in the comments. Cosine similarity is a metric used to measure how similar the documents are irrespective of their size. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. Here, the vectors refer to tokenized word representations of a subreddit post. Finally, we leveraged tensorboard to display the results of the GPT2 chatbot model word embeddings using Principal Component Analysis (PCA).

4. Results

4.1. Building a Chatbot

4.1.1 Transformer with Multi-head Attention Mechanism:

(Code in **multi_head_attention_chatbot.ipynb)**

The transformer model was first proposed in the "Attention is all you need" paper, published in 2017. As described by the authors of "Attention is All You Need", Self-attention is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence. In other words, Self-attention is the method the Transformer uses to incorporate the "understanding" of other relevant words in the sentence and their positions into the sentence we are processing.

Model Architecture:

The transformer architecture consists of the Encoder-Decoder framework. The encoder converts the original input sequence into its latent representation in the form of hidden state vectors. The decoder tries to predict the output sequence using this latent representation.

This novel model uses an attention architecture and is heavily parallelizable which is its biggest benefit. Another improvement is that the attention-based model would inadvertently give a higher weightage to the elements in the sequence closer to a particular position.

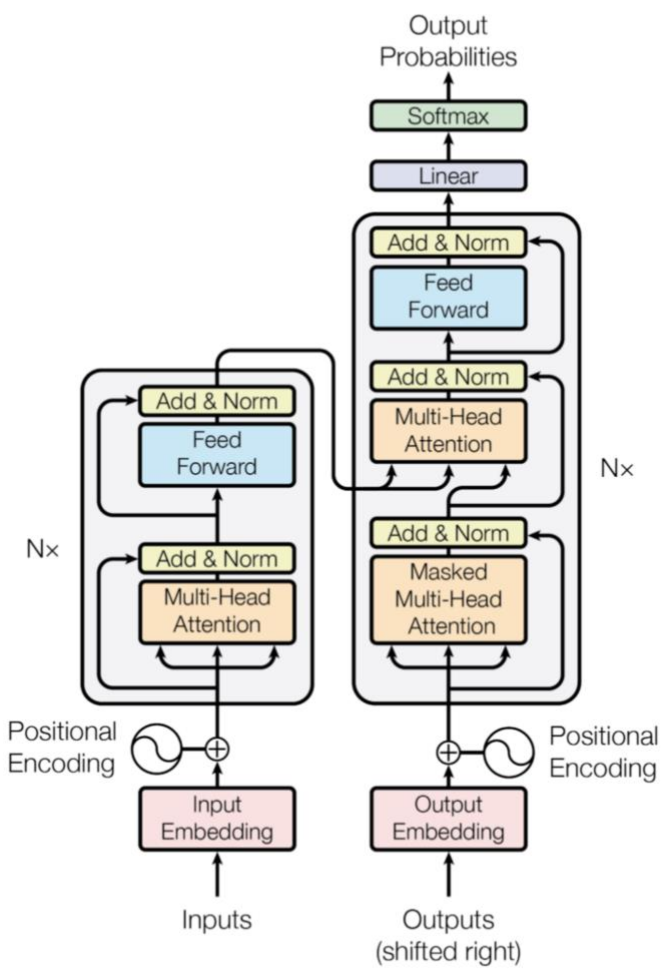


Fig 1

Image source: https://medium.com/tensorflow/a-transformer-chatbot-tutorial-with-tensorflow-2-0-88bf59e66fe2

The encoding component in the model we implemented is a stack of 6 encoders and the decoding component is a stack of decoders of the same number. The number of encoders and decoders is not fixed and can be changed to improve the model accuracy. Although all the encoders are identical, they do not share the same weights.

The encoder’s inputs first flow through the multi head attention layer. Attention mechanism relates to the different positions of a single sequence in order to compute a representation of the same sequence. The attention function maps a query(Q) and a set of key(K) - value(V) pairs to the output. The query, key and values are all vectors. The attention function is called a scaled dot product attention and its formula is given below(formula 1). The input consists of queries and keys of dimension dk, and values of dimension dv.

$$Attention(Q, K, V) = softmax_k(\frac{QK^T}{\sqrt{d_k}})V$$

Formula 1

The output is created by multiplying the attention weights and the value vector. This process ensures that the words we want to focus on are kept like they are, and the irrelevant words are flushed out. Multiple sets of query, key and value are learned using the same words and thus it is called a multi head attention mechanism.

RNNs and LSTMS use the sequential nature of data. In the attention model, there is no component which explains the sequential nature of the data; thus, we need to inject some information about the relative or absolute position of the tokens in the sequence. This is done using the positional encoding vector and is added to the embedding vector at the bottom of the encoder and decoder stack.

The outputs of the multi head attention layer are fed to a feed-forward neural network. The same feed-forward network applied to each position separately and identically.

The decoder has both those layers, but between them is an attention layer that helps the decoder focus on relevant parts of the input sentence. The output of the decoder is the input to the linear layer and its output is returned as the final result.

Dataset:

Our initial intention was to build a chatbot solely based on comments scraped from Reddit. We then realized that such a dataset would be small, and it would be hard for the model to learn enough about the language and incur the common sense to generate fluent and relevant responses. Thus, we built a dataset by combining conversations from the famous Cornell University's Movie Dialogue Corpus with the posts and comments scraped from the subreddits specifically News, Worldnews, Science and Askscience.

Cornell Movie dialog corpus:

The Cornell Movie dialog corpus contains a metadata-rich collection of fictional conversations extracted from raw movie scripts with over 220,579 conversational exchanges.

We have used the following files from the Cornell Movie dialog corpus to extract the conversational exchanges. The movie_conversations.txt contains a list of the conversation IDs and movie_lines.text contains the text associated with each conversation ID. For convenience, we created a neatly formatted data frame in which each row contains a query sentence and a response sentence pair.

We also found a pre-processed dataset with reddit data that can help build the conversation model. This data came from Reddit posts/comments under the r/CasualConversation subreddit. The conversations under this subreddit were significantly more 'conversation like' when compared to other subreddits (Ex. r/AskReddit). We used this to enhance the conversations already provided in the cornell movie dialog corpus

Subreddit Scraping:

We used the PRAW reddit API to scrape posts and comments from the News, Worldnews, Science and Askscience subreddits.

For each of the subreddits, we scraped all hot posts along with their title and body. A query-answer dataframe was created by mapping each comment to its corresponding post. As the comments are nested in Reddit, we also scraped the second level of comments in a structured way and appended these to the query-answer dataframe while preserving the reference of a comment either to its parent comment or to the subreddit post. By this process, we were able to create over 195,000 query-answer exchanged from reddit.

We finally merged the extracted Cornell Movie dialog dataset with the reddit dataset and had it ready for pre-processing. The data frame consists of two columns namely questions and answers. A portion of this data frame used to train is displayed in Fig 2.

	questions	answers
0	Can we make this quick? Roxanne Korrine and A...	Well, I thought we'd start with pronunciation,...
1	Well, I thought we'd start with pronunciation,...	Not the hacking and gagging and spitting part....
2	Not the hacking and gagging and spitting part....	Okay... then how 'bout we try out some French ...
3	You're asking me out. That's so cute. What's ...	Forget it.
4	No, no, it's my fault -- we didn't have a prop...	Cameron.

Fig 2

Data-preprocessing:

First, we substituted the contractions of a phrase within a sentence with the phrase itself. For example: “I'm” is simply the contraction for “I am”. Thus, we substituted “I’m” with “I am” and this was done for all possible contractions. We then converted every sentence to its lower case format, substituted some punctuation with empty spaces and added a start and end token to each sentence. We then removed rows where the questions or answers are shorter than two words or longer than 20 words.
Next, a tokenizer was built using the TensorFlow Datasets SubwordTextEncoder to map text to its ID and every ID to its text. To each tokenized sentence, a START_TOKEN and END_TOKEN were added to indicate the start and end of each sentence.

We then set the maximum length of a tokenized sentence=40 and filtered out rows if either the question or answer exceeded this limit. Finally, if any tokenized sentence was less than the length of 40, padding was added to that sentence.

Model building and Training:

The transformer consists of the encoder, decoder and linear layer. To build the model, we utilized some of the code blocks available in the tensorflow transformer chatbot implemented by google:
https://colab.research.google.com/github/tensorflow/examples/blob/master/community/en/transformer_chatbot.ipynb#scrollTo=iZMuzj0cVr3E
https://colab.research.google.com/github/tensorflow/examples/blob/master/community/en/transformer_chatbot.ipynb#scrollTo=iZMuzj0cVr3E
Once the model architecture was setup, we defined the loss function, optimizer and metrics. Finally, model.fit() was called to train the model. We trained the model for a minimum of 300 epochs. Although, training the model further could have improved results, there was not a significant decrease in the loss function towards the end and training time was approximately 15-16 hours per 100 epochs.

Output:

Since we trained the model on both data from the cornell movie data corpus and reddit, we divided the testing into predicted responses related to regular conversation vs responses related to news and science.

Below in Figure 3 and Figure 4, we have asked the transformer to predict responses to general conversational topics such as “How are you doing” and “What are you doing this month?”. Considering that the model used a fraction of the data used to train models like the GPT2, we were surprised at the accuracy of the responses. Some of our observations are listed below:

- The model is also able to generate coherent responses when the input is a regular conversation topic
- The model is able to hold the conversation and is able to produce large amounts of text
- There is tremendous repetition with the usage “i m”, “ i have” to form the sentence structure

```
[72] output = predict('How are you doing?')

     Input: How are you doing?
     Output: i m doing pretty good . how are you ?
```

Fig 3

```
# feed the model with its previous output
sentence = 'What are you doing this month?'
for _ in range(3):
  sentence = predict(sentence)
  print('')

Input: What are you doing this month?
Output: i m working on a tv show , i m just watching youtube , and i have a pc game so i m going to try and get a few days .

Input: i m working on a tv show , i m just watching youtube , and i have a pc game so i m going to try and get a few days .
Output: i have never played the game before . i just had a small business , but i would love to play the game with my friends .

Input: i have never played the game before . i just had a small business , but i would love to play the game with my friends .
Output: i m sure you will be fine
```

Fig 4

When asked to predict responses related to news or science queries(Fig 5), we noticed some of the following characteristics in the responses received:

- Although the replies do not seem completely illogical, there is again significant repetition in the usage of “I m” or ‘I think” to frame the sentence. There is also repetition in the usage of phrases within the sentence such as “it is a bad thing it is not a bad thing”.
- The context of the response does not always align with the query.
- The sentence structure is also not always coherent when the input is related to a subreddits.
- Although not displayed in the image below, in some cases, there was a lot of repetition between outputs generated from the model.

```
[73] output = predict("When will we get a stimulus check?")

     Input: When will we get a stimulus check?
     Output: i m not sure what you are talking about , but i m not saying it is a bad thing but it is not a bad thing to do with it .

[74] output = predict("How do we know stars are made of gas?")

     Input: How do we know stars are made of gas?
     Output: i think it is a huge metal container will be very rare .

[76] output = predict("Are stars are made of gas?")

     Input: Are stars are made of gas?
     Output: yes . and i have seen the movie with the movie with the movie open .

[77] output = predict("Can I eat plastic?")

     Input: Can I eat plastic?
     Output: sure , but i m not hungry cookies and i will eat the cake
```

Fig 5

4.1.2 Transfer Learning with GPT2

(Code in **(gpt2_transformer.ipynb))**

GPT2, the successor to GPT, is a recent transformer model capable of achieving state of the art performance on 7 out of 8 tested language modeling datasets and used for projects such as AI dungeon and Tabnine. GPT2 has numerous applications which researchers have identified, including but certainly not limited to: reading comprehension, summarization, translation, question answering, chatbot models, auto-completion, and others. Perhaps more importantly, different sizes of the GPT2 model -- pre-trained on 8 million webpages -- is now open source for researchers to use and build off of with techniques such as transfer learning. This has resulted in projects such as AI dungeon, which acts as a “dungeon master” and guides players engaging in an unorthodox dungeon and dragons’ game, and Tabnine, a tool which adds auto-completion across different code scripting languages.

We were interested in using one of the smaller GPT2 models which allows finetuning on a single GPU to understand the capabilities of this model and compare it to our transformer from scratch. We first finetuned this model on the Cornell movie dialog dataset to augment its conversational capabilities to create the base conversational model. Then, using this base conversational model, we finetuned two models using different datasets -- 1) the combined Science and Askscience subreddits, and 2) the combined News and Worldnews subreddits -- to produce one “science” and one “news” chatbot.

Data Scraping:

For GPT2 data scraping, we used a slightly different heuristic to match the data format expectations of the pre-trained GPT2 model. Specifically, the GPT2 model is interesting because it can generate variable length text off of a “prefix” passed in. To train and evaluate the model, we need to feed in meta context in the form of manually inserted delimiters into the original text, which the model learns to use in its prediction. For the conversational AI task the text summarization task, the following format was used.

<|startoftext|>
speaker 1
> speaker 2
speaker 3
> speaker 4
<|endoftext|>

Then, for data generation a prefix in the following form was passed.

<|startoftext|>
speaker 1
>

The model’s response was then truncated at <|endoftext|>.

We first created a dataset for training the base conversational model by reformatting the Cornell movie dialog dataset to adhere to the above format. Then, to produce the subreddit datasets, we used “top comment max depth recursion” to produce “comment trees” of multiple replies matching certain criteria. With this approach, for each top level comment, we grab the top reply, and continue until reaching the end of the tree. The criteria we use are that 1) the comment is not ‘deleted’ or ‘removed’, and 2) the length of the comment is less than 150, as at greater lengths we observed it was more likely that the user was going off on an unrelated tangent. If these criteria are not met, then we end the comment tree at that length.

During the data scraping process, we created a few different datasets before settling on the appropriate formatting and methodology, as we learned lessons about characteristics of the GPT2 model. The first lesson was on the amount of data required to train GPT2 to prevent overfitting. The original dataset curated used only the “hot” reddit threads and was around .5 MBs, and we quickly realized overfitting was occurring by matching the output’s of the model to comments in the original dataset. After researching into this problem, we came up with two possible -- and somewhat obvious -- solutions 1) reducing the amount of training time, and 2) increasing the dataset to a threshold where the samples didn’t repeat during the course of training (which is dependent on the training time chosen in #1). We modified our data scraping approach to construct larger datasets, by amalgamating the “top” and “hot” reddit threads, and were able to create datasets of sizes 4 MB (r/news and r/worldnews subreddits) and .8 MB (r/science and r/askscience). Finally, we verified that overfitting was no longer occurring by examining both of the trained transformer models’ predictions to verify they were no longer plagiarized from the original dataset.

The next lesson learned was on the importance of cleaning the data by expunging ‘removed’ and ‘deleted’ comments. In the first few datasets we produced, we didn’t remove these comments, and quickly realized that transformers trained off of these datasets had a high frequency of predicting ‘deleted’ and ‘removed’ when asked to generalized to data out of distribution with questions that we input into the model after training [Figure 6]. In hindsight, this makes a lot of sense -- the transformer should guess ‘deleted’ or ‘removed’ when out of distribution, since these occur with a reasonably high frequency. We solved this issue by cleaning these comments from our dataset.

```
<|startoftext|>
 Will we get a stimulus check?
[removed]
[removed]
I think it's a complicated question, but it's quite easy to think of it as
a
====================
```

Figure 6: Poor transformer performance on the uncleaned subreddit dataset. Based off of the prefix “Will we get a stimulus check?”, the model predicts ‘[removed]’ twice, and doesn’t fully answer the question.

GPT2 Finetuning:

To finetune a pre-existing GPT2 model, we used the GPT2 simple API from https://github.com/minimaxir/gpt-2-simple, which provides an easy to use wrapper around the open AI GPT2 release at https://github.com/openai/gpt-2 for working with GPT2 transformer models. We also selected the model size of 124 M, or 124 million parameters, after confirming the larger models could not train locally or on google colab due to GPU memory constraints. This model is equivalent to the original GPT model according to OpenAI.

Our training pipeline was a little circuitous as we were restricted to using Google colab in order to access a GPU with 16 GB of RAM, and as a result had to circumvent issues due to frequent runtime disconnections. In order to finetune the GPT2 model, we loaded in the 124 M model, and used the provided finetuning method. We passed in arguments to the ‘save_every’ keyword argument to specify frequent checkpointing of our model. We also made use of the ‘restore_from’ argument to continue training in the event that the runtime did disconnect to specify that the training should reload the model from the latest checkpoint. Using this heuristic, we first finetuned GPT2 on the Cornell movie dialog dataset for 1000 epochs to produce the base conversational chatbot. Then, we finetuned the base conversational chatbot further to produce two separate chatbots trained on the Worldnews + News datasets, and Science + Askscience datasets. We trained these transformers for 1000 and 200 epochs respectively, to prevent overfitting, and then generated data from these models.

Results: News chatbot

After training, we asked both of our chatbots “How are you” to better understand their conversational performance. Neither the news or the science chatbots perform highly or at a level near human mistakable, and it is difficult to characterize which chatbot is more performant. The science chatbot attempts to link a worldhealth article, interestingly enough [Figure 7, Figure 8].

```
<|startoftext|>
 > How are you?

A lot of people are like this and have no idea what to do.
 I was thinking they were just waiting for Trump to do something.
====================
**** the system, **** these fuckers and I hope they get someplace to go.
 They are trying to be transparent with the media.
====================
A doctor.
====================
 I'm still in shock.
====================
 Unlike some videos where you are a violent ****** with guns and don't call out anyone else.
====================
```

Figure 7: News chatbot performance on “How are you” input. Different sample generations are delimited using “=====”. The model underperforms, and doesn’t produce human mistakable output for any of the five generated samples.

```
<|startoftext|>
 > How are you?

https://www.worldhealth.nlm.nih.gov/pmc/articles/PMC2608231/
====================
 I'm in my 40s, 5ft, broad chested, with dark eyes and a deep cut on my face.
====================
Didnt you work as a nurse etc?
====================
 Good to know.
```

```
====================
 I’m 24.
====================
```

Figure 8: Science chatbot performance on “How are you” input.

Different sample generations are delimited using “=====”. The model underperforms, but does mention its “age” which might indirectly answer the question.

We next proceeded to ask more topical questions of the transformers. First, we asked the news chatbot “When will we get a stimulus check?”. In response, even though this question is currently unanswerable, the chatbot produced human mistakable output. It references the date of the election, conflating the two incorrectly, in one generated response. It also provides wishy-washy responses such as “I think it will start in the coming weeks” and “It’s been 2 weeks and we haven’t heard anything from them” that we characterized as strong responses (that would pass the Turing test!) [Figure 9].

```
<|startoftext|>
 When will we get a stimulus check?
 > I think it will start in the coming weeks.
====================
 > When they send their kids to school.
====================
 > It has been delayed until Nov 4th
 ====================
 > I thought they were just giving out free college tuition to
students.
====================
 > It’s been 2 weeks and we haven't heard anything from them.
====================
```

Figure 9: News chatbot performance on “When will we get a stimulus check?” input. Different sample generations are delimited using “=====”. The model is performant, and produces human mistakable outputs. Many of the answers are not technically incorrect, or might reflect the thoughts of a subreddit poster.

We then asked the science chatbot “How do we know stars are made of gas?”. This is a difficult “how” question, and the responses weren’t as strong. The chatbot cited a non-existent paper by Einstein, likely mistaking the Einstein supernova award for a paper from him [Figure 10]. However, the model does successfully link a wikipedia article on astronomy that actually successfully answers this very difficult question, which is an exciting behavior we hadn’t predicted! We followed up this question with the easier question of “Are stars made of gas?”, and recorded much more human understandable output such as “stars themselves are made of gas” and “stars form when stars are hot”, which isn’t exactly technically incorrect.

```
<|startoftext|>
 How do we know stars are made of gas?
```

```
> Einstein explained this in his paper "Supernovae"
 ====================
 > We know that planets are made of gas with protoplanets that orbit
them.
====================
 > There are lots of theories for this.

https://en.m.wikipedia.org/wiki/Astronomy#/wiki/
====================
 > Well, they aren't making gas, you can see in the picture in the
title.

> But stars are made of gas.
====================
> Stars simply don't exist without Planck' instruments.
====================
```

Figure 10: Science chatbot performance on “How do we know stars are made of gas?” input. Different sample generations are delimited using “=====”. The model is performant. The second and third answers are both strong responses, and the model successfully links a reference which is very interesting.

```
<|startoftext|>
 Are stars are made of gas?
 > /u/sillyman
====================
 > Yes. The stars themselves are made of gas.
====================
 > Yes
 > Is there some sort of biological origin of the stars?
====================
 > Yes. Stars form when stars are hot. Stars are made of carbon.
====================
 > Yes. Maybe stars are made of hydrogen?
 > Yes. The Kuiper Belt objects are thought to have formed
relatively early.
 <
====================
```

Figure 11: Science chatbot performance on “Are stars made of gas?” input with temperature set to .7. Different sample generations are delimited using “=====”. The model is performant, answering yes in four out of the five responses and elaborating with the names of specific atmospheric gases.

We also looked at modulating the temperature parameter, which affects the randomness of predictions and can vary between 0 and 1 (with 1 indicating more randomness) [https://github.com/huggingface/transformers/issues/2029]. We tried a lower temperature of .3

(previous data generations used a temperature of .7), and observed that the predictions were much more stable and correct [Figure 12].

```
<|startoftext|>
 Are stars are made of gas?
 > Yes, they are made of hydrogen.
https://en.m.wikipedia.org/wiki/Hex
====================
 > Yes, the gas in the core is the primary gas.
====================
 > Yes. Stars are made of hydrogen.
====================
 > Yes, they are formed when the stars close in on one another.
https://en.m.wikipedia.org/wiki/Ast
====================
 > Yes. Stars are made of hydrogen.
https://en.m.wikipedia.org/wiki/Hexagon

====================
```

Figure 12: Science chatbot performance on "Are stars made of gas?" input with temperature set to .3. Different sample generations are delimited using "=====". The model is performant if repetitive, and answers yes in all five of the samples generated.

Finally, we tried the query "Can I eat plastic?" to examine the likelihood that the transformer would advise us to eat plastic. In three out of the five responses, the transformer not only gave the rejoinder "no", but also gave reasons why it was a bad idea -- that plastic was toxic, and a carcinogen. One reply was nonsense "plastic is plastic", and one reply was misleading "plastic is not toxic". In conclusion, listening to the output of transformers is probably still a poor idea, at least in 2020.

```
<|startoftext|>
 Can I eat plastic?
 > Plastic is toxic.
====================
 > Plastic is toxic.
====================
 > Plastic is a known carcinogen.
====================
 > Plastic is plastic.
====================
 > Plastic is not toxic. Plastic is just water.
====================
```

Figure 13: Science chatbot performance on "Can I eat plastic?" input. Different sample generations are delimited using "=====". The model is performant and correctly advises not to eat plastic in three out of the five responses.

4.2 Comments Summarization

4.2.1 Extractive Comments Summarization using spaCy's language model

(Code in **(data_exploration.ipynb))**

Our aim was to extract the top sentences from all comments within a subreddit post. Extractive summarization is the process of identifying the most important sentences within a text and extracting them verbatim to produce a subset/summary. Although this process is fast, it may not create a good flow between sentences and does not produce human-like summaries.

spaCy is a natural language processing (NLP) library for Python with word embedding models built into it and relatively fast performance. spaCy can be used to access sentences within each comment and named entities within the comment. Entities are the words or phrases that represent information about common things such as persons, locations, organizations, etc.

We have primarily used spaCy to tokenize words instead of using TF-DIF. Since we were dealing with multiple domains(news vs science), TF-DIF would not have been a good choice as it leads to biases in an unbalanced dataset which can greatly affect the sentences that are deemed most important.

Data Scraping and Pre-processing:

Posts and comments from the News, Worldnews, Science and Askscience subreddits that were scraped using the PRAW reddit API for the Transformer with Multi-head Attention Mechanism are reused here. Each post consists of a title and many comments. All the comments under each post have been joined together. Our intention is to run the extractive summarization model over all the comments under each post and extract the top sentences.

The data frame has two columns namely Post Title and Comments. We preprocess the comments to turn all text into the lower case format and remove any urls starting with https. We do not need to tokenize, lemmatize and remove stop words and we are trying to extract complete sentences from the comments.

***Algorithm*:**

1. We clean the comments by removing all extra whitespaces.
2. Tokenize the comment into words using the spaCy language model
3. Score each word based on its frequency and capitalization. Stop words and punctuation is not considered in the score. Reduce the score if the token is a digit.
3. Split the comment into separate sentences and score each sentence. Sentences with words with a higher word score will be ranked higher.
4. Return top ranked words and sentences in chronological order.

Output:

The title of the subreddit post was:

“people are desperate': california shutdown pushes businesses to breaking point.”

The original comments contained some of the following text. Only a portion of the comments is displayed below and the rest is truncated as the comments text is over 1000 words long:

“the issue is that there is little federal support for businesses or people disproportionately affected by shutdowns in the us. sucks that an particular industry was affected by a virus more than other industries while you do kinda have a point about it being a food fight, it is more of the republicans fault. the dems have passed multiple bills in the house to extend/expand the cares act and get more support to people and businesses in need. the rs have been fighting it tooth and nail because it helps people more than corporations. the rs also want a giant liability shield for companies so that they cant be sued if they arent following covid regulations like masking/cleaning/distancing. if the corps get that shield then they will start forcing people back to work in unsafe environments making this whole thing worse. the bottom line is they should have been 100 percent shut down but they needed federal aid for employees and owners to do that trump's government fully abandoned us and now we can't blame them for staying open so they can pay bills.”

The top sentences extracted were:

-bottom line is the reason we don’t trust people and the reason things are getting shut down, is because enough people aren’t even trying to do the right thing that it’s raising the risk for everyone by an unacceptable level.

-people can also wear a mask while shopping for groceries, and the fact that people are moving around makes spread less likely.

- every arm flailing rule and action has been done because if we just told people who were at risk (old, obese, poor health) that they and their immediate people needed to take actions to protect themselves but the rest of society had to suck it up and carry on they would have bitched and whined about how it was "unfair" to them and them alone.

Evaluation:

Notice how the extracted sentences are complete and coherent. A lot of improvement can be made in terms of enhancing the link between the different sentences in the output thus creating a story-like summary. There also needs to be a normalization between extremely long and short sentences. We have thus implemented summarization using GPT2 to overcome these challenges.

4.2.2 Comments Summarization using GPT2

(Code in **(gpt2_transformer.ipynb))**

*GPT-2 can be easily adapted to abstractive text summarization with a different dataset. In lieu of summary labels, we used the thread titles as our “summaries". Then, we trained GPT-2 to use the content in reddit threads to predict the thread title. We once again trained two separate GPT-2 models using a combination of the Science and AskScience subreddits for the first, and the News and Worldnews subreddits for the second. In the following sections, we’ll further discuss our methodology and preliminary results.

Data scraping:

To produce training data, we once again created our training dataset using a data format with added meta context in the form of delimiters. We adhered to the below data format.

<|startoftext|>
reddit thread
<|sep|>
thread title
<|endoftext|>

Then, for data generation, the following prefix was used.

<|startoftext|>
reddit thread
<|sep|>

The model’s response was then truncated at <|endoftext|>.

To produce the text summarization AI dataset, we scraped the top one hundred comments and the top level (i.e. we didn’t consider replies to those comments) as the “reddit thread” and the thread title as the “thread title” according to the previously described data format. We made the decision to scrape only the top 100 comments after observing that for too few comments (e.g. around fifteen), the model performance was compromised. We settled on one hundred in the interest of limiting the size of data accumulated per reddit thread, so we sample a diverse and relatively high number of reddit posts (on the order of thousands).

Results:

We evaluated the abstractive summarization capabilities of our science and news GPT-2 models on two different subreddit posts a piece. To generate the data, we used the same data scraping methodology and then truncated the scraped data to the first 1000 characters, after observing that the model outputted blank titles if the full string of top comments was used. The news summarization model performed reasonably well, depending on the complexity of the topic. For the reddit post “Canada-U.S. border closure extended to Jan. 21 as coronavirus cases soar”, the model output reasonable summaries such as “Canada to extend border closure to US border until 2021”, and even computed the math on how many months the border closure would extend for [Figure 14]. However, on the much more complex reddit post “2.6 terabyte leak of Panamanian shell company data reveals "how a global industry led by major banks, legal firms, and asset management companies secretly manages the estates of politicians, Fifa officials, fraudsters and drug smugglers, celebrities and professional athletes.", the news summarization model did not perform quite as highly, but did correctly identify the “Panama Papers and german journalists/media as key subjects, and produced reasonable summarizations such as “Panama Papers: German journalists have taken part in uncovering documents hidden from public view”. Only one response, “Panama Papers leak reveals how top German media figures hid money hidden offshore for decades”, is not supported by the thread content. We also noted that the news summarization model had an affinity for picking up on the stylistic conventions of titles in the news subreddit, such as using colons and the “__, says” title format.

Title:

```
    Canada-U.S. border closure extended to Jan. 21 as coronavirus cases
soar
```

Predictions:

```
Canada to extend border closure to US border until 2021

====================
Canada can extend border closure to US for 2 months, says minister

====================
Canada to extend border closure until 2021, says department

====================
Canada to extend border closure to mid-month

=====================
Canada: US border closed until Nov. 21, 2021

====================
```

Figure 14: News summarization model performance on ‘border closure’ thread. Different sample generations are delimited using “=====”. The model is performant, human mistakable in all five samples, and correct in four out of five samples.

Title:

```
2.6 terabyte leak of Panamanian shell company data reveals "how a
global industry led by major banks, legal firms, and asset
management companies secretly manages the estates of politicians,
Fifa officials, fraudsters and drug smugglers, celebrities and
professional athletes."
```

Predictions:

```
Panama Papers leak reveals how top German media figures hid money
hidden offshore for decades

====================
Panama Papers: German journalists have taken part in uncovering
documents hidden from public view

====================
Panama papers leak financial documents hidden from public view

====================
Panama papers leak documents revealing how wealthy and powerful the
elite are.

====================
Panama Papers: German media ask questions about the Panama papers.

====================
```

Figure 15: News summarization model performance on 'data leak' thread. Different sample generations are delimited using "=====". The model is performant and correctly identifies key actors such as the Panama Papers and german journalists/media.

Finally, we evaluated our science summarization model on two subreddit threads, and achieved reasonable performance on both. First, we trained on a "AskScience AMA Series" with a wide range of topics. The samples generated are hit or miss, with the first sample mentioning non-existent "biological cameras", the second mentioning the "human-occupied deep-ocean" (likely due to the mention of "human-occupied deep submergence" in the thread), and the third and fourth posing nonsensical questions (e.g., "How can we ensure that HOVs and drones are not alone in their use of the deep?"). At first glance they seem reasonable, but deeper inspection shows that many of the statements are illogical. Only is the last sample, "How can I get involved in the conversation of wildlife conservation?", relevant to a topic discussed in the thread (although it is not a full summary) [Figure 16].

Title:

```
AskScience AMA Series: Hello, I'm Dr Pen-Yuan Hsing from the
University of Bath in the United Kingdom & have worked in
ecology/conservation, founded a citizen science wildlife-monitoring
```

```
project and am also an active open science/open source advocate. Ask
me anything!
```

Predictions:

```
AskScience AMA Series: I'm Dr. Hsing, what is your role in bringing
awareness to the need for biological cameras and how could you
impact
====================
How can we better understand the human-occupied deep-ocean?

====================
How can we ensure that HOVs and drones are not alone in their use of
the deep?

====================
Why is it that the ocean's temperature varies as it expands?

====================
How can I get involved in the conversation of wildlife conservation?

====================
```

Figure 16: Science summarization model performance on 'AskScience' thread. Different sample generations are delimited using "=====". The model produces poor responses in four out of five of the generated samples.

We followed this analysis with another analysis of the science summarization model's performance on the subreddit thread: "How long does it take for comets to completely lose all their mass?" Surprisingly, the model was more performant on this post. The second and third replies are good summaries of topics discussed in the subreddit thread. However, the first comment is nonsensical, and the fourth and fifth comments bring up topics not discussed in the current reddit post [Figure 17]. It is clear that the science transformer model struggles to identify scientific truths when generating texts.

Title:

```
How long does it take for comets to completely lose all their mass?
I assume that, at the rate they expel gas and dust they would only
last maybe a few hundred orbits.
```

Prediction:

```
Why is the sun still pointed at the Earth on the Halley's comet, but
the other 13,000 light years away is slowly drifting toward the
====================
How does the Sun's gravity affect the comet Halley's?
```

```
====================
How far from the sun does an comet orbit?

====================
How do comet showers behave? Is there a limit to how long they last
on average?

====================
How does a comet like comet 67P/Churyumov-Gerasimenko manage to gain
mass from the Sun?
```

Figure 17: Science summarization model performance on ‘comet mass’ thread. Different sample generations are delimited using “=====”. The model produces two reasonable responses, and off-topic or nonsensical responses in the remaining samples generated.

While we were able to produce interesting abstractive summaries using this approach, we noted that the model was sometimes nonsensical or off-topic, bringing in subjects not discussed in the post under investigation. We think the second tendency is more worrying, as it demonstrates how the large dataset behind GPT2’s training can harm rather than help. Using the GPT2 transformer model for text summarization, and more broadly, to produce factual content is definitely a domain which merits further investigation.

4.3 Data Exploration

4.3.1 Latent Dirichlet Allocation

(Code in **(data_exploration.ipynb))**

LDA is used to extract topics from a document corpus. It can be used to build topics per document model or words per topic model. These are modeled as Dirichlet distributions.
The LDA model generates topics which are specific groups of words that can be used to represent the topic of a document/corpus.

Data Scraping and Pre-processing:

We used subreddit datasets(news, worldnews, askscience, and science to perform topic modeling using LDA. We scraped all the hot and top posts from the four subreddits to create the dataset. We will be using the subreddit post title and its body for LDA.

For LDA, consider each subreddit post to correspond to a document. We cleaned each post title and its by processing the following steps:

- converting all text to lowercase
- Removing any hyperlinks or URLs using regular expression
- Tokenizing words in each post into tokens

- Lemmatizing to convert any inflected forms of words into its base form
- Remove commonly occurring stop words as found in the english NLTK dictionary along with [http', 'com', 'www', 'askscience'] and other punctuation used in URLS or http text

When then used the simple text created from the preprocessing module above and converted it into its vectorized form using the TF-DIF vectorizer.

LDA Algorithm:

Prior to topic modelling, we convert the vectorized simple text into a bag of words model — which is basically a dictionary where the key is the word and value is the number of times that word occurs in the entire corpus. We use the gensim.corpora.dictionary to create these id to word tokens.

We filter out the words that occur less than 3 times or more than 80 percent of the time. Now for each pre-processed post we use the dictionary object created to convert that post into a bag of words corpus. i.e for each post we create a dictionary reporting how many words occur and how many times those words appear. We finally call the gensim LDA model by passing the bag of words corpus, id to word tokens and the number of topics we want to generate.

Results:

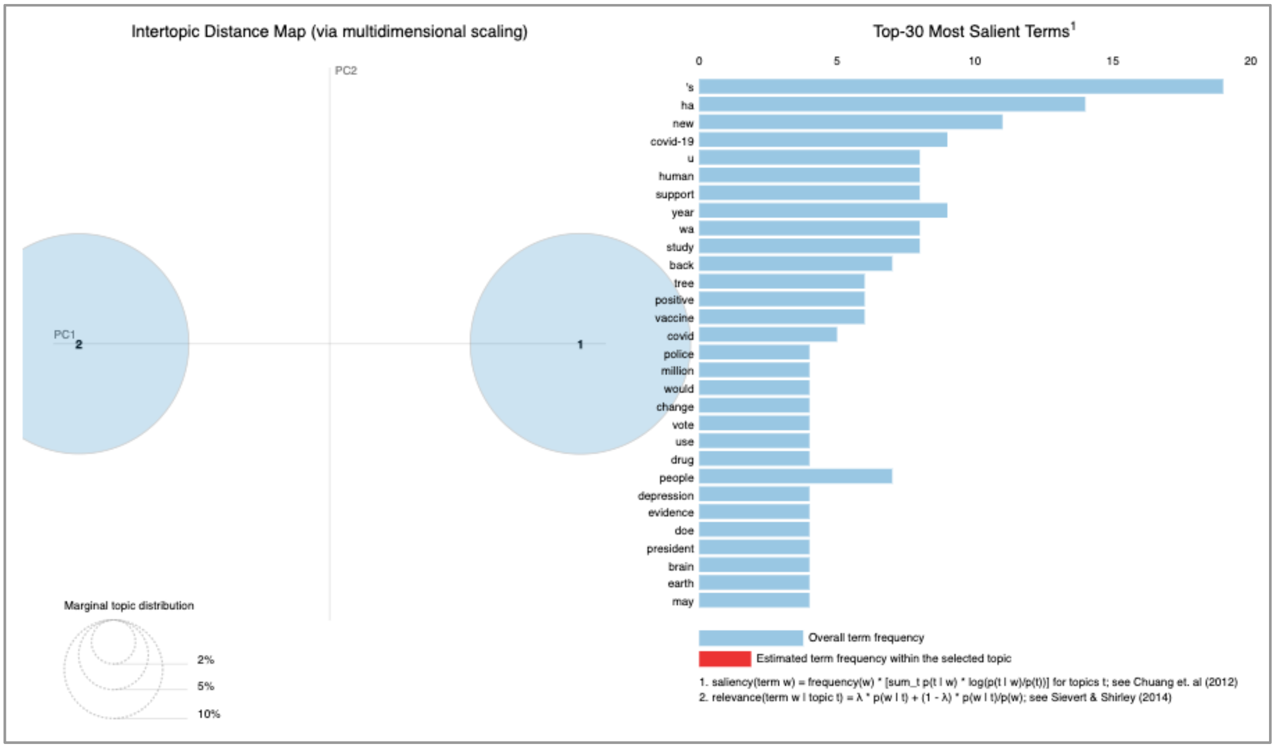
We chose to conduct a visual inspection of the topic clusters and salient features using the pyLDAvis package. pyLDAvis allows you to pass a gensim model and get a d3-built visualization within the notebook.

The following images (Fig 18 and Fig 19) have been generated using the pyLDAvis package. The results from pyLDAvis for number of topics=2 are displayed in these images.

In Fig 18, while the two generated topic models seem to be far apart in the intertropic distance map(possibly suggesting two different subreddits: news and science), by looking keenly into the top 30 salient terms detected in each cluster in Fig 19, we notice a lot of repetition of top selected words within the two topics. These repetitive words include: “Trump, president, covid, coronavirus, human, vaccine”.

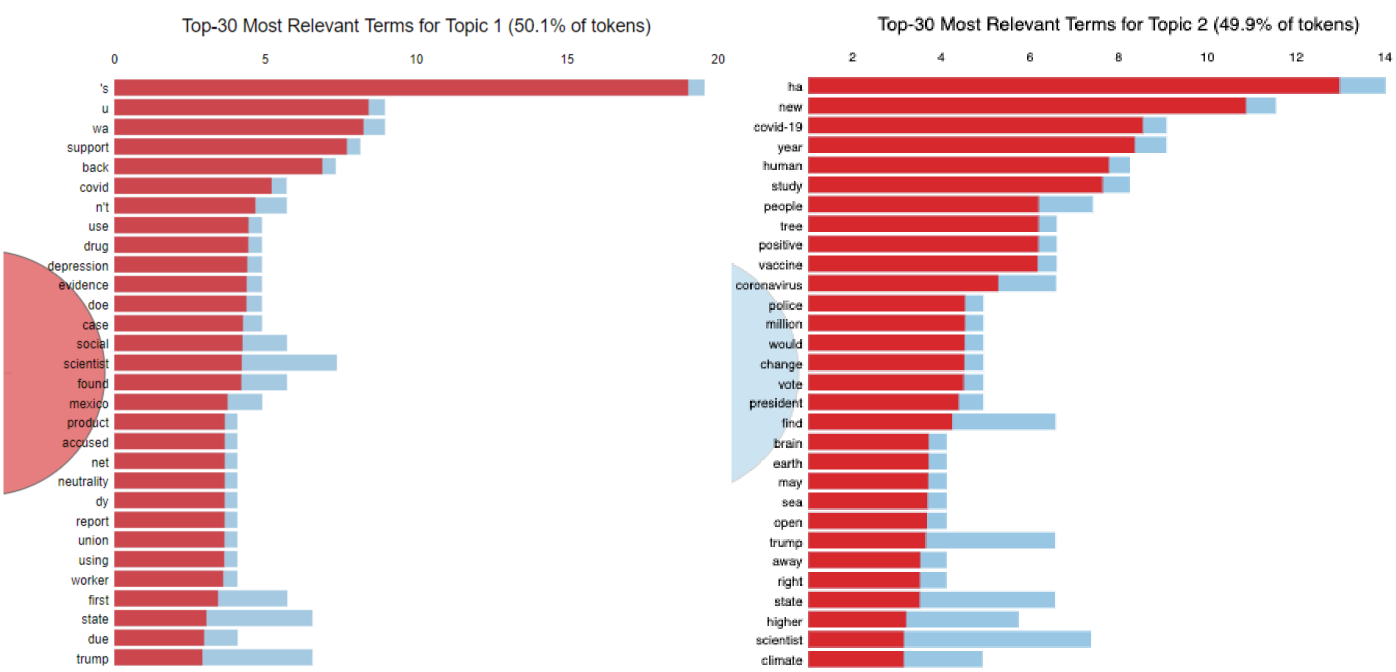
Topic 1 seems to be related to news with top salient features including words like the following: “ drug, evidence, social, mexico, accused, net, neutrality, covid and trump“.

Topic 2 seems to be related to science with top salient features including words like : “ covid -19, positive, vaccination, scientist, climate, tree, study, president”.



Intertopic Distance Map (via multidimensional scaling)
PC2
PC1
2
1
Marginal topic distribution
2%
5%
10%
Top-30 Most Salient Terms1
0
5
10
15
20
's
ha
new
covid-19
u
human
support
year
wa
study
back
tree
positive
vaccine
covid
police
million
would
change
vote
use
drug
people
depression
evidence
doe
president
brain
earth
may
Overall term frequency
Estimated term frequency within the selected topic
1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

Fig 18



Top-30 Most Relevant Terms for Topic 1 (50.1% of tokens)
0
5
10
15
20
's
u
wa
support
back
covid
n't
use
drug
depression
evidence
doe
case
social
scientist
found
mexico
product
accused
net
neutrality
dy
report
union
using
worker
first
state
due
trump
Top-30 Most Relevant Terms for Topic 2 (49.9% of tokens)
2
4
6
8
10
12
14
ha
new
covid-19
year
human
study
people
tree
positive
vaccine
coronavirus
police
million
would
change
vote
president
find
brain
earth
may
sea
open
trump
away
right
state
higher
scientist
climate

Fig 19

4.3.2 Reddit Posts Similarity using Cosine Similarity

(Code in **(data_exploration.ipynb))**

We use cosine similarity as a quantification of measurement to identify if two subreddit posts are similar to each other. Quantification of the similarity between two posts can be obtained by converting the words or phrases within the corpus or sentence into a vectorised form of representation. The vector representations of the posts are then used within the cosine similarity formula to obtain a quantification of similarity between them.

Data Scraping and Pre-processing:

Posts and comments from the News, Worldnews, Science and Askscience subreddits that were scraped using the PRAW reddit API for the Transformer with Multi-head Attention Mechanism are reused here. Each post consists of a title and many comments. All the comments under each post have been joined together. We run the comments from each post under the cosine similarity scanner to determine if the posts are similar.

The data frame has two columns namely Post Title and Comments. Once we have the dataset, we process the data further to make it usable for the model. First step of preprocessing would be converting each comment into an array of tokens, lemmatizing the tokens and removing unwanted characters/patterns and stop words. Tokenization is the process of breaking down text into words. We then apply Lemmatization to change multiple inflected forms of words into its base form. We then remove any commonly occuring stop words from the NLTK english dictionary. We have also removed stop words corresponding to the numbers, URLs and leftover components from lemmatized words. We then use TF-DIF to vectorize the simple text generated through pre-processing.

Cosine Similarity:

All the combinations of the subreddit comments are created and passed to the cosine similarity function to measure the similarity between the two vectors of an inner product space. The similarity is measured by the cosine of the angle between two vectors which helps to determine whether two vectors are pointing in roughly the same direction.
The closer the similarity score of a pair of subreddit comments is to 1, the more similar they are to each other.

Results:

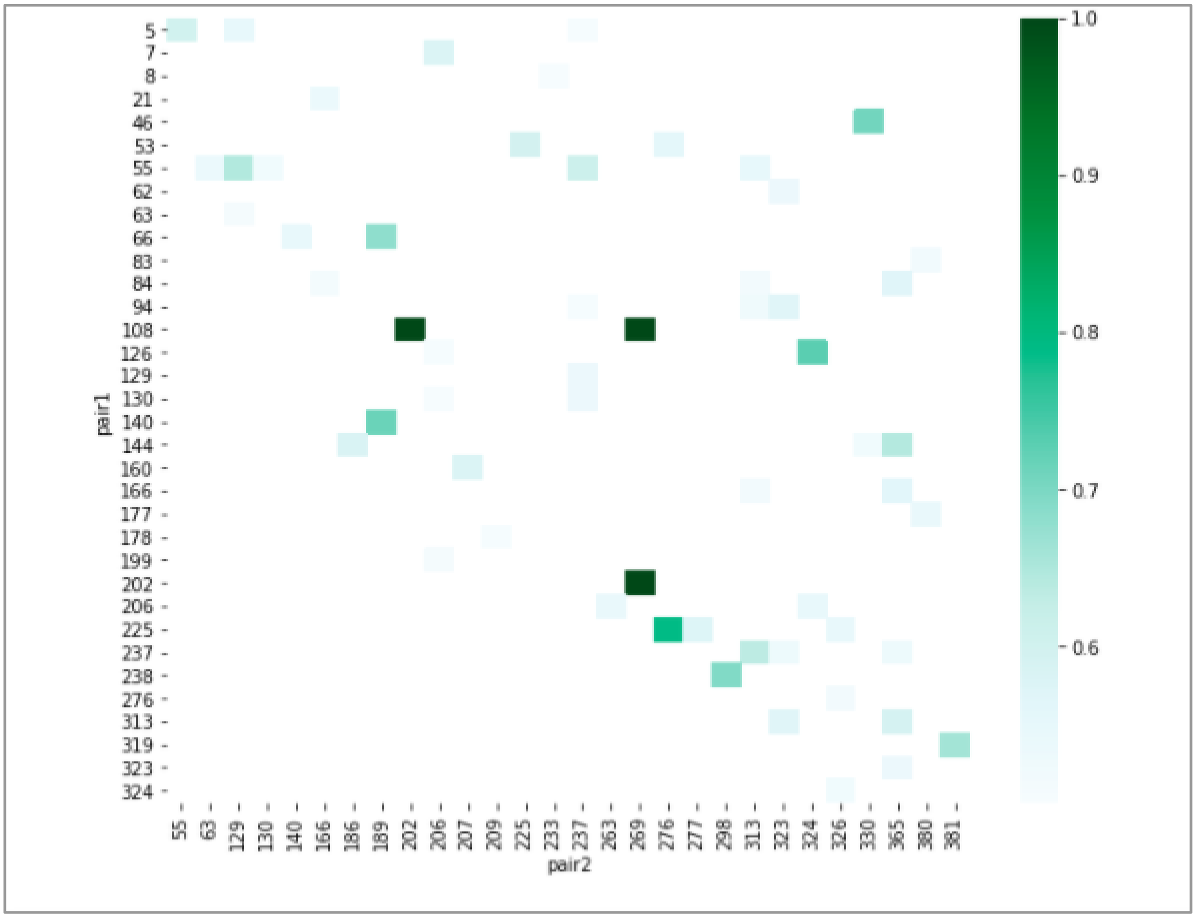


Fig 20

From the heatmap above in Fig 20, we see that comments from subreddit 108, 202 and 269 are extremely similar to each other as they have a similarity score ~ 1. Comments from subreddit 276 and 225 also are similar with a similarity score of 0.75. Out of all the combinations of over 330 different posts, fewer than 10 posts have a similarity score of 0.75. This shows that reddit is already doing a good job at handling redundancies.

Extension of Cosine Similarity:

Although in this project, we have only determined if two posts are similar to each other, there are multitude of opportunities to implement cosine similarity in subreddits to drive traffic.

1) We can use cosine similarity to recommend other subreddits to a current user. For example, if the user regularly reads subreddits related to hiking, we can refer him to relatable subreddits such as camping, outdoors.
2) We can use cosine similarity to club certain subreddits together, eg: club askScience and science subreddits together.
3) Use similarity modeling to detect redundant posts and merge them along with their votes

4.3.3 Visualization of word embeddings using PCA

(Code in **(gpt2_transformer.ipynb))**

We furthered our exploration of our curated dataset by examining the word embeddings for the set of tokens created from our dataset. This was made easy by using tensorboard in combination with the GPT2 model checkpoint produced by the GPT2 simple finetuning script. We created a tab separated text file to link word id's to corresponding tokens, and then visualized our results for the dataset used to train the news chatbot [Figure 21]. Unfortunately, it was difficult to make heads or tails of the data given the exceedingly high number of tokens (~50,000).

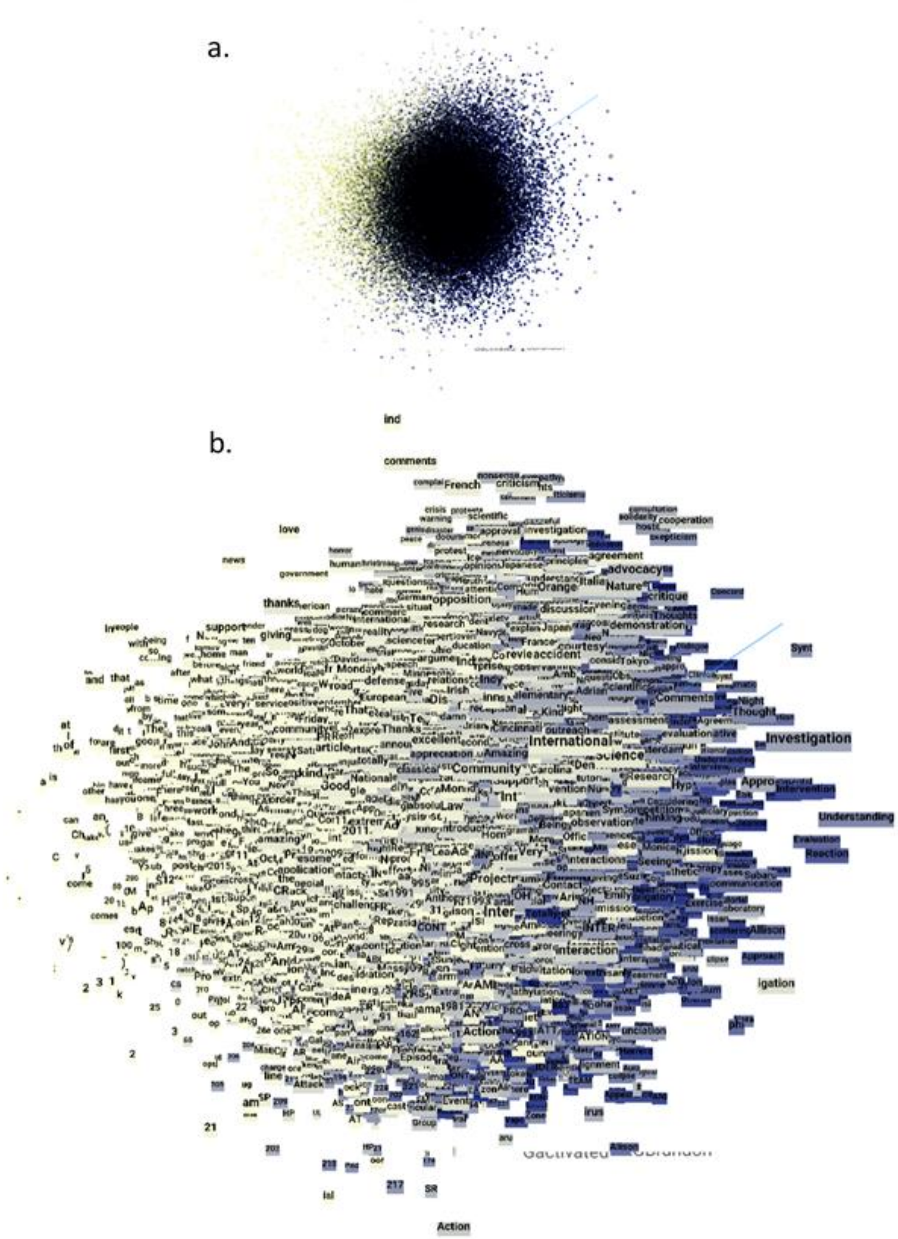


Figure 21: PCA word embedding visualization. Blue coloring indicates high frequency tokens, yellow coloring indicates low frequency tokens. 3d token labels are visualized in b).

We were able to further analyze the word embedding by honing in on certain regions using the box selecting (supplied by the tensorboard API). In figure 21a, we selected high frequency words, and identified “Confederation”, “PTSD”, “suspicions”, and “distrust” as closely clustered, high frequency words. We also tried selecting outliers, visualized in figure 21b, but it was still difficult to identify trends given the sheer number of datapoints.

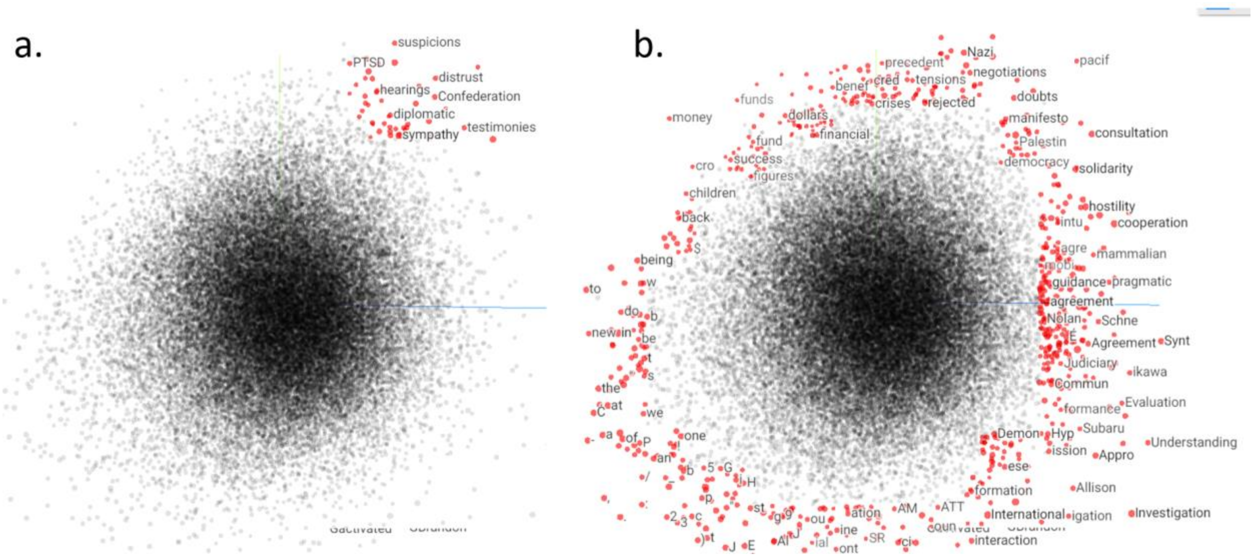


Figure 22: PCA word embedding visualization with region selection. In a) we selected a region with higher relative frequencies of word tokens observed, in b) we selected outliers in the current 2-dimensional view.

5. Conclusion:

While the chatbots created using the transformer model with the multi head attention mechanism and the GPT2 model with transfer learning performed well, we did recognize some strong differences in the quality of results when tested with the same utterances as the input. The vanilla chatbot performed reasonably well with conversational utterances but was not capable of generating coherent results for test inputs related to the news or science subreddits. We have attributed this behaviour to the size of data and the amount of training conducted on the model.

The GPT2 model performed much more strongly than the vanilla chatbot model when tested with inputs related to the news or science subreddits. While the GPT2 model outperformed the vanilla chatbot in these topics, it surprisingly did not produce concise and

coherent responses on conversational queries such as “How are you doing?. We attribute this to the fine-tuning pipeline, which involved training initially on the conversation dataset from the cornell movie dialog corpus and then on the respective subreddit datasets.
In summary, we’d like to defer to the output to our summarization transformer model. After passing the motivation section from this final report as input to the news summarization transformer model, we produced the following (cherrypicked) abstractive summaries:

(News)
“For the first time, AI-powered robot learning could help people better understand the meaning of their lives”
(Science)
“Transformer: a machine learning model of human attention”

6. References:

https://towardsdatascience.com/transformer-attention-is-all-you-need-1e455701fdd9
https://github.com/nshepperd/gpt-2.
https://arxiv.org/pdf/2004.08900.pdf, https://syncedreview.com/2019/06/27/the-staggering-cost-of-training-sota-ai-models/
https://arxiv.org/abs/2006.01997
https://towardsdatascience.com/understanding-cosine-similarity-and-its-application-fd42f585296a
https://en.wikipedia.org/wiki/Transformer_(machine_learning_model)
https://arxiv.org/pdf/2004.08900.pdf, https://syncedreview.com/2019/06/27/the-staggering-cost-of-training-sota-ai-models
https://arxiv.org/abs/2006.01997
https://d4mucfpksywv.cloudfront.net/better-language-models/language_models_are_unsupervised_multitask_learners.pdf
https://d4mucfpksywv.cloudfront.net/better-language-models/language_models_are_unsupervised_multitask_learners.pdf