GPT-2 model for tweet generation

Sara Knežević[1]

Singidunum University, Danijelova 32, 11000 Belgrade, Serbia
sara.knezevic.17@singimail.rs

Abstract. The development of transformers and their role in Natural Language Processing has led to incredible implementations in various different NLP fields. From writing prompts, text adventures, chat-bots to improved text prediction and translation. GPT-2 is currently one of the best publically available generative language models which can be fine-tuned to create various types of texts. This paper covers the use of the model in order to experiment and see how the model would generate tweets (short texts up to 280 characters) based on a dataset of tweets from 10 different accounts that cover different topics in specific writing formats. The results show that about 1 in every 5 generated tweets could be considered meaningful or comedic. This is due to the fact that the dataset itself contains abbreviations, jargon, emoticons and sarcastic comments, which makes it more difficult for the model to process. Training the model on tweets from one person would produce much better results, as it would learn the nuances and features of the way the specific person writes.

Keywords: NLP · Generative models · Twitter · GPT-2 · Transformers

1 Introduction

Natural-language generation is a software process which outputs language structures. In the 2010s, with the development of deep neural networks and representational learning, natural language processing (NLP) became widespread as its performance of various natural language tasks (modeling, parsing, reading comprehension, translation, summarization, etc.) with state-of-the-art results. The primary method in training neural networks is gathering a large dataset which are used to train the system to mimic the patterns and then perform independently with high enough accuracy. Due to this nature of the systems, they are often very specialized in specific tasks but don't perform well when put in a completely new environment with a more abstract task. This lack of generalization can only be resolved with more training of the network and inputting a wide range of different tasks.

Synthetic text generation has seen a huge improvement with the development of Generative Pre-trained Transformer (GPT) models by OpenAI. In this project, GPT-2 fine-tuning was performed using a dataset of tweets from 10 different twitter accounts. GPT-2 is a successor to GPT and it was trained to predict the

next word in 40GB of Internet text. GPT-2 is a large self-supervised transformer-based language model with 1.5 billion parameters with the goal of predicting the next word, given all of the previous words within a text. [1] The inputs themselves are sequences of continuous text of a certain length and the targets are the same sequences, only shifted one token (word) to the right.
The GPT-2 architecture itself is not new, it is very similar to a decoder-only transformer, but training it on such a large dataset has significantly improved text generation. This project explores the possibility of a simpler, less trained version of GPT-2 to generate tweets.

2 Generative Pre-trained Transformers

The 1.5 billion parameter model presented in OpenAI's paper [1] was initially unpublished due to ethical concerns, however smaller versions (ranging from 117 million to 1.542 million parameters) have been released for use in the public as 4 different version of the model. This paper uses the smallest, 117 million parameter model, correspondingly named GPT-2 small.

2.1 Transformers

The transformer model consists of an encoder and a decoder. The encoder maps an input sequence of symbol representations $(x_1, ..., x_n)$ to a sequence of continuous representations $z = (z_1, ..., z_n)$ which the decoder generates into an output sequence $(y_1, ..., yn)$ one element at a time. Each step is auto-regressive, meaning at each step all of the previous steps are the input to the next.

Encoders consist of a stack of identical layers which has its own sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, position-wise fully connected feed-forward network.
Decoders consist of the same amount of stacks of identical layers, but in addition to each sub-layer, the decoder has a third sub-layer which performs multi-head attention over the output of the encoder stack. An attention function can be described as mapping a query and a set of key-value pairs to an output,where the query, keys, values, and output are all vectors. In other words, it pays attention to future blocks, as well as the previous ones. The query is a representation of the current word used to score against all the other words. Key vectors are what is matched against in the search for relevant words. Value vectors are actual word representations, once scored how relevant each word is, these are the values that are added up to represent the current word. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key. Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. With a single attention head, averaging is done instead. [2]

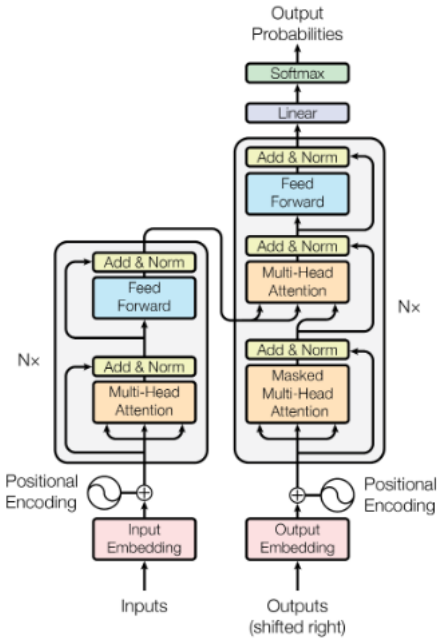

Fig. 1: Transformer model architecture

2.2 GPT-2 model

A lot of transformer-based models have recently been performing slicing on the architectures and removing either the encoder or the decoder and using a stack of transformer blocks, making them as large as possible, and feeding that stack large amounts of text. [3] GPT-2 uses this technique and keeps only the decoders and the GPT-2 dimensionality of the embeddings and hidden states is 768. The attention function in GPT-2's decoder-only architecture pays attention to specific segments of the input and masks the future tokens in order to block them from being considered in the calculation, which is later passed into a feed-forward neural network. The primary tokens used for this project are $< |startoftext| >$, $< |endoftext| >$ and $< |pad| >$, denoting the start of text, ending and filling in empty slots (padding), respectively. The model only has one input token which is processed successively through all the layers, then a vector is produced along that path. That vector can be scored against the model's vocabulary. When the top block in the model produces its output vector (the result of its own self-attention followed by its own neural network), the model multiplies that vector by the embedding matrix - individual scores for each word in the model's vocabulary. This process outputs a single word and it repeats this until the entire context is generated or the end of text token is produced.

3 Development

3.1 Gathering tweets

Fine-tuning neural networks is done by gathering new data and introducing it to a new model. For this project 25552 tweets were scraped from 10 different twitter accounts using Python, Twitter's API and the corresponding python library tweepy. The tweets themselves were stripped of URLs, annotations, tags and hashtags in order to provide text only input to the model and they are stored in a single-column csv file, which is convenient for the model as the start and end tokens are inserted around each row in the csv. The data is standardly split to 80-20% for training and testing, respectively.

"Tweets are bad for society"

- GPT-2 trained in this project, prompted with 'Tweets'

"I am on twitter because the only person who knows they are in the real world knows their circumstances, they are in fact insufferable"

- Also GPT-2, prompted with 'I am on twitter because'

During the development of the model a lot of consideration was taken into account regarding the choices of twitter accounts for data gathering, but in order to represent the true spirit of the social network itself (and a bit for comedic purposes), the data remained.[1]

3.2 Hyperparameters and optimization

Due to memory limitations, the network was trained with a batch size of 1 over 5 epochs, which made the training itself about an hour per epoch, however after several combinations of different parameters, a training loss of 0.15 and a test loss of 0.28 were produced with the parameters marked in Table 1.

Table 1: Hyperparameters

Params	Values
Epochs	5
Learning rate	5e-4
Warm-up steps	1e3
Epsilon	1e-7

AdamW optimization was used in this model, which implements stochastic gradient descent where the weight decay is performed after controlling the

[1] Generated tweets may contain problematic language and emoticons at times.

parameter-wise step size. The default decay coefficient is 1e-2 and the value wasn't changed, however epsilon was modified - a parameter which improves numerical stability of the network.
The optimizer was used with a combination of a scheduler, a method which creates a constant learning rate preceded by a warm-up period during which the learning rate increases linearly between 0 and the initial learning rate set in the optimizer itself. Warm-up steps is a parameter which is used to lower the learning rate in order to reduce the impact of model deviation from learning on sudden new data set exposure.

4 Results

The outputs of the model are not guaranteed to produce quality tweets and sensible texts in every case. The quality tweets, ones that would convince you that a human wrote them, are rare. When generating new tweets in the code, the accuracy limit was set to be 99% which improves the outputs but certain parts of the sentences may get stuck in a loop or make no sense at all.
Since each twitter account in general has an informal and very personal signature, with a lot of abbreviations, sarcasm and language finesses - it's expected that a model that uses statistical and mathematical models to produce sentences would have a harder time representing those forms of text. Training the network on one person or way of writing would definitely produce better results as the model would learn the person's nuances.

References

1. A. Radford and J. Wu and R. Child and D. Luan and D. Amodei and Ilya Sutskever, "Language Models are Unsupervised Multitask Learners', 2019.
2. A. Vaswani and N. Shazeer and N. Parmar and J. Uszkoreit and L. Jones and A. N. Gomez and L. Kaiser and I. Polosukhin, "Attention Is All You Need," *arXiv preprint arXiv:1706.03762*, 2017.
3. P. J. Liu and M. Saleh and E. Pot and B. Goodrich and R. Sepassi and L. Kaiser and N. Shazeer, "Generating Wikipedia by Summarizing Long Sequences," *arXiv preprint arXiv:1801.10198*, 2018.
4. https://pytorch.org/docs/stable/optim.html
5. https://medium.com/swlh/everything-gpt-2-2-architecture-comprehensive-57129fac417a
6. https://www.kdnuggets.com/2020/08/transformer-architecture-development-transformer-models.html
7. https://huggingface.co/gpt2