Published in Towards Data Science

Prakhar Mishra
Dec 14, 2021 · 5 min read ★ · Listen

Text Data Augmentation Using the GPT-2 Language Model

Generating synthetic data in NLP using Pre-trained Language Model GPT-2

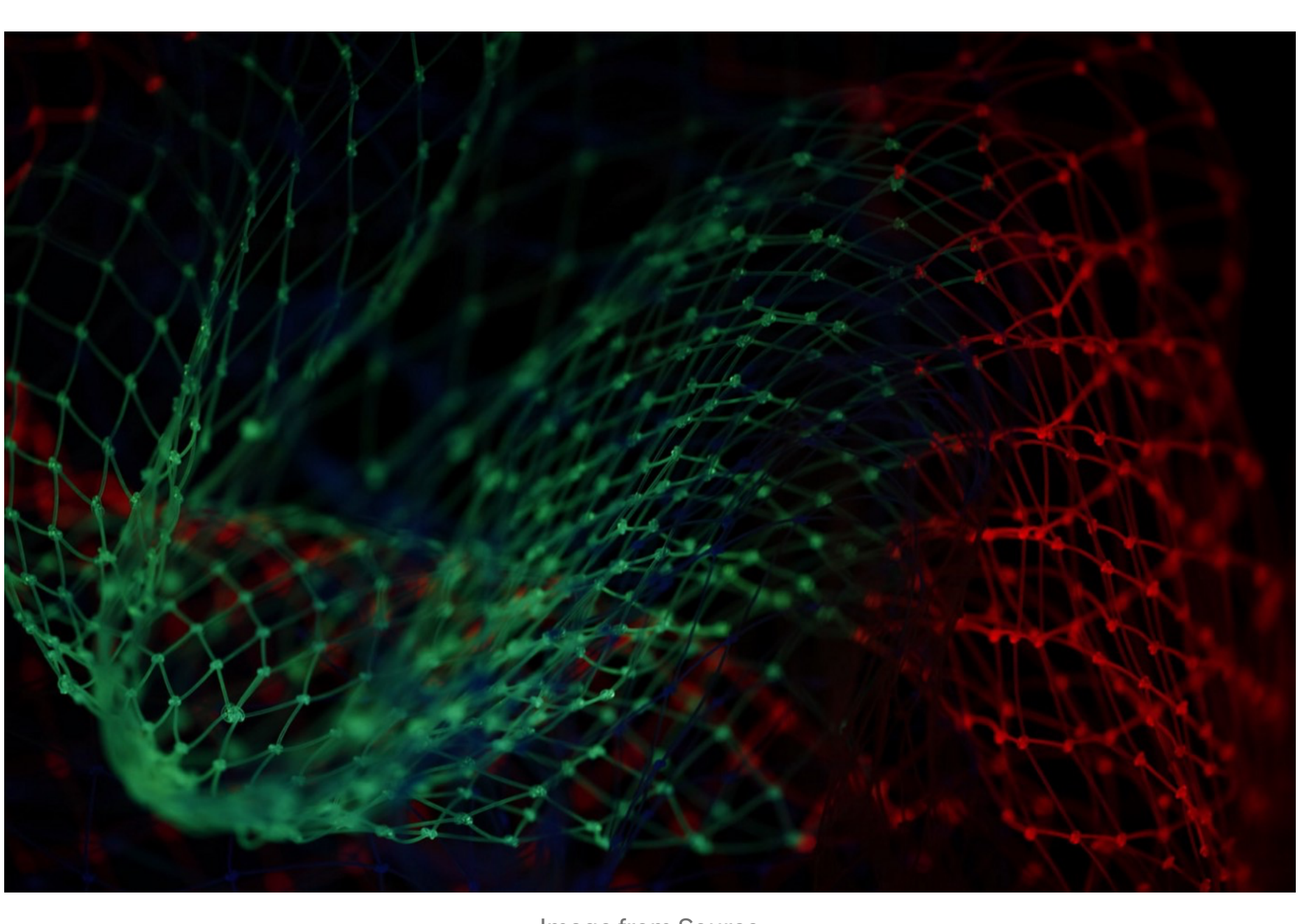


Image from Source

Data Augmentation is a technique that is heavily used by deep learning practitioners to add diversity and size in their training dataset for designing robust machine learning systems. Every engineer wants their model not to just perform well on the training set but also generalize well to the unseen scenarios. So apart from overfitting and regularization, one of the other important factors that determine the generalization of a model is the amount and variety of related data it sees during the training time. As of today, there are a lot of tested transformations available on Images for doing augmentation that works amazingly well under low-resource settings. But it's not that easy when it comes to Text data. Simply, because Natural Language encapsulates various levels of syntactic and semantic information.

Some of the existing Approaches and Problems

Past work, in this domain, focused on **synonym replacement** of certain special types of words in grammar using WordNet, Word2Vec, etc approaches. Such approaches act as a good starting point but do not add much value to our models in terms of providing variability. Also, these systems are very brittle in nature. For example, WordNet has a fixed set of words and will often result in Out-of-Vocabulary, whereas, treating the nearest neighbor from pre-trained Word2Vec as a synonym would not always give desired results in practice. For example — The closest neighbor of the word "amazing" is "spider". And this is also correct in some sense considering "Amazing Spider-Man" is a movie.

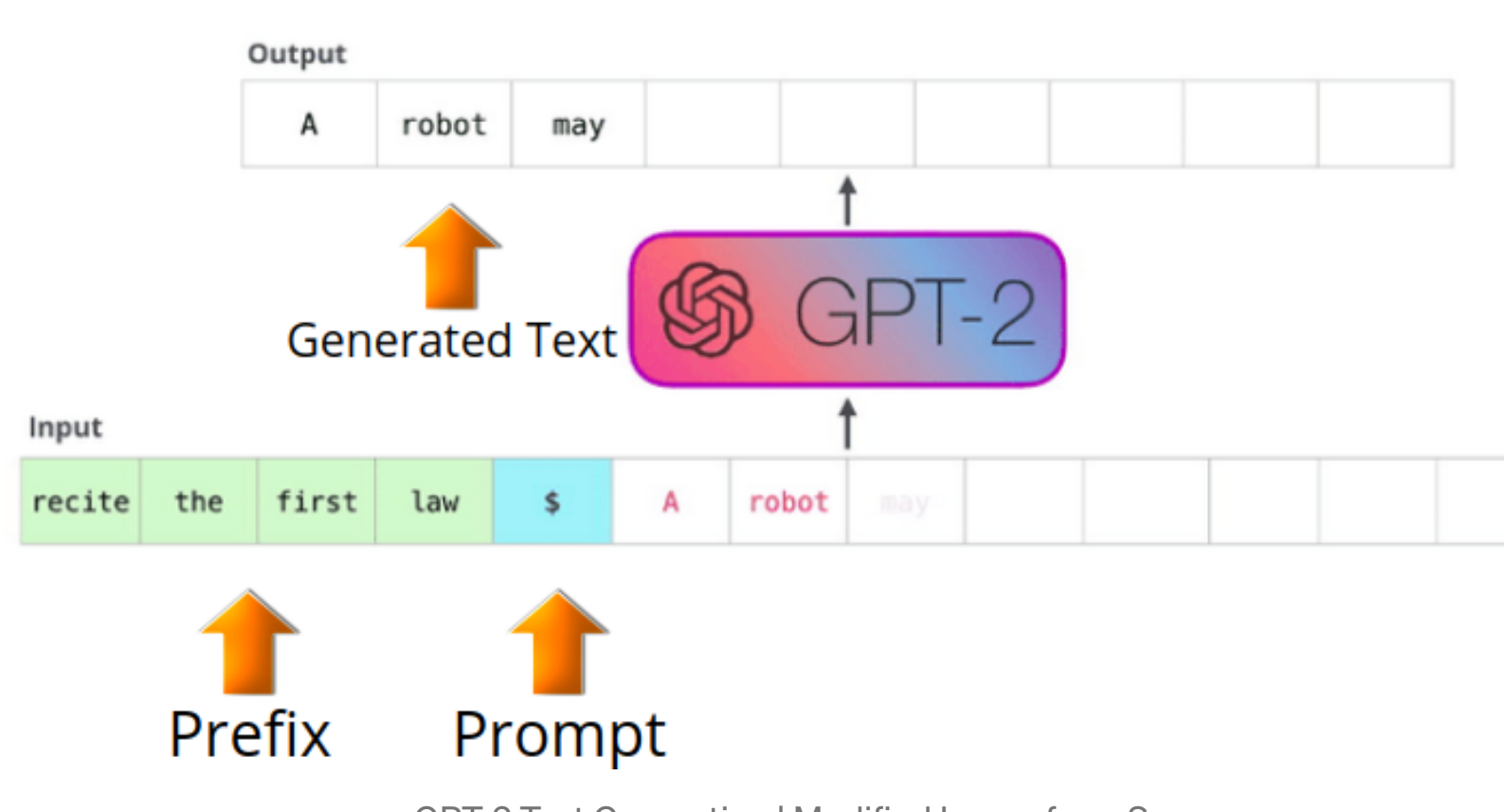
One of the interesting papers in this domain is EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. The author's in this paper talk about 3 ways for augmenting text data that have proven to improve results for the task of text classification. *Please feel free to follow the tagged post for more in-depth details regarding the same.*

> *Today, in this blog we will see how we can use **GPT2 for doing high-quality text augmentation**.*

Before we jump to code and methodology, I would like to take some time to explain GPT-2 in a paragraph or two. GPT-2 deserves a separate blog of its own, *for which you can follow* The Illustrated GPT-2 (Visualizing Transformer Language Models)

GPT-2 Language Model

GPT-2 is essentially a sophisticated Language Model at heart that is based on **Transformer Architecture** and is trained on 40GBs of WebText. It's a stack of multiple decoder units on top of each other enabled with some advanced learning concepts like Masked Self Attention, Multiple Heads, Residual Connections, Layer Normalization, etc. GPT-2 language model tries to optimize is to essentially predict the next word in the given sequence having seen past words. The below figure i.e. borrowed from The Illustrated GPT-2 (Visualizing Transformer Language Models) gives a clear picture visually —



GPT-2 Text Generation | Modified Image from Source

Here the green-colored text acts as a prefix, post which once the prompt ($) is seen, the model starts generating one word at a time in an autoregressive fashion till the end of the token is reached. The Autocomplete feature on smartphones, Autocompose in Gmail is essentially built on similar concepts.

Training such huge models would require quite a lot of GPU days and a lot of data. To our luck, versions of this model were open-sourced saving us from training these models from scratch. We can now directly fine-tune such models on our tasks by giving them decent-sized domain-specific data. Let's now see how can we do it in practice.

Let's move forward and fine-tune the GPT-2 model on our classification dataset and see it generate realistic examples for a given class.

Dataset

We will be using the Email Spam Dataset for the purpose of our experiments. . We have ~5500 samples in our dataset. The below figure shows the snippet of the same -

```
ham     Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...
ham     Ok lar... Joking wif u oni...
spam    Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt
ham     U dun say so early hor... U c already then say...
ham     Nah I don't think he goes to usf, he lives around here though
spam    FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it still? Tb ok! XxX s
ham     Even my brother is not like to speak with me. They treat me like aids patent.
ham     As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your callertune for all Callers
```

Email Spam Dataset data snippet | Image by Author

Approach

We would be using GPT-2 model for **learning the word distribution, semantics, syntactic constructs around both the classes Ham and Spam**. And once the training is done, we would be generating synthetic samples for each of these classes which you will see are good enough to be added back to our dataset for training a classification model with even larger data.

The pre-trained GPT-2 model becomes a suitable candidate for this task as we have very few samples for each of the Spam/Ham classes with us in the dataset. We would ideally want a model that already knows a lot about **syntactic, grammatical, semantic structures** of natural language. As a part of the implementation, we will fine-tune GPT2 on our Spam/Ham dataset and expect it to learn the word usage, language structure of such emails.

We start by creating input samples by concatenating labels and text to each other and passing this to the GPT-2 model for **learning the word-word and label-text dependency relations**. Adding the label to the actual sample would help us later in guiding the model to sort of control the generation of text and keep it specific to a given label/topic.

Training GPT-2 is straight forward as training any other language model, in which we pass one word at a time and predict the next on the other end and then loop the generated word back to the input and so on and at every step, we calculate the cross-entropy loss. *You can access the training code here.*

Once the model is trained and dumped, we are ready to generate samples. You can find the code for generating samples using the trained model at finetune_gpt_generate. We employ the **Top-k, Top-p sampling strategy** as our choice of text decoding technique.

Results

Here are some of the novel samples generated by our trained model. As promised, we can see that the generated samples look very real with enough variability and should hopefully help our model in pushing the accuracy numbers and generalize better.

```
SPAM: You have 2 new messages. Please call 08719121161 now. £3.50. Limited time offer. Call 090516284580.
SPAM: Want to buy a car or just a drink? This week only 800p/text betta...<|endoftext|>
SPAM: FREE Call Todays top players, the No1 players and their opponents and get their opinions on www.toda
SPAM: you have been awarded a £2000 cash prize. call 090663644177 or call 090530663647<|endoftext|>

HAM: Do you remember me?<|endoftext|>
HAM: I don't think so. You got anything else?<|endoftext|>
HAM: Ugh I don't want to go to school.. Cuz I can't go to exam..<|endoftext|>
HAM: K.,k:)where is my laptop?<|endoftext|>
```

Generated Samples | Image by Author

All of the code and pre-trained spam/ham email generator model can be found at —

GitHub - prakhar21/TextAugmentation-GPT2: Fine-tuned pre-trained GPT2 for custom topic specific...
Fine-tuned pre-trained GPT2 for topic-specific text generation. Such a system can be used for Text Augmentation. git...
github.com

If you'd like you can also check out some recent research in NLP data augmentation space at this.

I hope you enjoyed reading this. If you'd like to support me as a writer, consider signing up to become a Medium member. It's just $5 a month and you get unlimited access to Medium.

References

1. The dataset used is publicly available and can be accessed at https://www.kaggle.com/venky73/spam-mails-dataset

Thank you for your time!