SI630 W19: Story Generation through GPT-2 Model

Arushi Jain , Sagnik Sinha Roy
arushij@umich.edu sagniksr@umich.edu

Abstract

Language or story generation has a long history of pursuit in artificial intelligence. Earlier people used to use hand-authored models but today, where computers have replaced humans in most of the tedious tasks it would not be surprising to see computers replacing humans in such creative tasks too. We present a model which produces stories of texts of varied lengths depending on the type of input. Our project is primarily based on the GPT-2 model [2] which demonstrates that language models begin to learn these tasks without any explicit supervision when trained on a new dataset of millions of webpages called WebText. We took the pre-trained embeddings, fine-tuned and build 3 different models for three types of corpora (short stories, essays and game of thrones novels). GPT-2 model has several other applications like Summarization, Question-Answering, Translation etc. but we have utilized it for text generation in this paper. Overall, we found that it was able to generate partially correct sentences (few grammatical and syntactic errors) for all 3 types of corpora which totally made sense upon reading. This means that model could infer the context from the input given and was able to produce realistic texts of varied lengths. Findings from this project suggest a promising path towards building language processing systems which learn to perform tasks from their naturally occurring demonstrations.

1 Introduction

Machine learning techniques have been consistently improving efficiency in most of the industries. Many labour tasks have become automated or "smarter" from the use of machine learning algorithms. For example, fraud detection in the banking sector has greatly reduced the speed with which banks can take action in resolving identity theft complaints. Natural language processing in particular, has seen a huge amount of improvement in the last decade, with the advent of openly available datasets, and powerful machines. Tasks like named entity recognition, sentiment analysis, etc have taken over the industry by storm, and are used in a wide variety of applications.

Through this project, we aimed to discover the potential of machines in the writing industry. While machines are adept at performing procedural tasks, can they truly simulate the human creativity? This is the question we aimed to answer satisfactorily by developing a story completion algorithm. An algorithm that can complete stories close to human writing would be revolutionary, and it would be very interesting to see the pieces of text that machines could potentially come up with.

This is a very recent field in Natural language processing. Although, Google has championed in this field by using Neural Networks to help you write emails but doing something creative like generating second half of the story based on the first half of the story inputted is still an untouched territory. Recurrent neural networks (RNNs) have been the most popular way to solve the problem of text generation, or text completion. In the recent years, Transformer Based Networks have replaced Long Short Term Memory (LSTMs), and have been increasingly used to solve the same problem.

Google AI team has launched a model in Nov18 called BERT (Bidirectional Encoder Representations from Transformers)[4] which predicts whether two sentences inputted are related or not. But OpenAI is still working on the sentence generation model called GPT-2 (Generative Pre-trained Transformer). It generates text samples in response to the model being primed with an arbitrary input. We tried to implement this technique where we trained models on a pre-trained model named GPT-2 which utilizes Transformer Based Network primarily to perform this task.

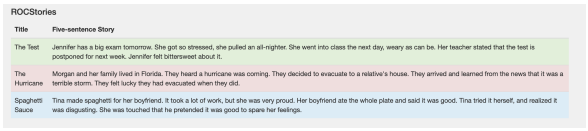
We saw that it was producing partially correct sentences since there were few grammatical and syntactic errors for all 3 types of corpora (Short Stories, Essays, and Game of Thrones). We evaluated our model on three different metrics namely Logistic Classifier, Average Perplexity and Cosine Similarity. For cosine similarity, we compared our predicted output with the actual output by calculating the cosine similarities of their average vectors and achieved more than 45% cosine similarity for all 3 corpora, highest for Short Stories (53.93%). Average perplexity is lowest for Game of Thrones corpus (312.3814) as compared to Short Stories (807.0305) and Essays (606.3497) which means that out of three models, Game of Thrones model is predicting best for unseen data. As per Logistic Classifier, both Short Stories and Game of Thrones model are performing well giving 49% accuracy, since in this case best performing model will have 50% accuracy (Refer Evaluation and Results for details).

Lastly, we think that this project would be interesting to the novel writers, tv script writers, poets who can leverage it for predicting the stories which might appeal to audiences given that they have trained on such a dataset. This might also be helpful in publishing sector, as with a good model, there are applications like book review, grammar check, etc. that could be automated, resulting in a more efficient industry.

2 Problem Definition and Data

We built a model which generates the ending for a given pieces of text. Our input was different pieces of texts with varied lengths(short sentences, essays, book chapter). We built training models separately for each type of text depending on its length. For example, essays will have a separate training data and model from book chapters. Our output is an ending for the given piece of text depending on input's length. Our data is varied lengths of text which includes short sentence, essays and book chapters to predict the ending for each of them.

Ideally, we would want our model to predict the outputs which resemble to the actual output in terms of storyline, plot of the character and type of language used (sarcasm, romantic etc). Apart from that other elements of any language model like grammar and fluency are very important and we would want the model to perform well on these


ROCStories

Title	Five-sentence Story
The Test	Jennifer has a big exam tomorrow. She got so stressed, she pulled an all-nighter. She went into class the next day, weary as can be. Her teacher stated that the test is postponed for next week. Jennifer felt bittersweet about it.
The Hurricane	Morgan and her family lived in Florida. They heard a hurricane was coming. They decided to evacuate to a relative's house. They arrived and learned from the news that it was a terrible storm. They felt lucky they had evacuated when they did.
Spaghetti Sauce	Tina made spaghetti for her boyfriend. It took a lot of work, but she was very proud. Her boyfriend ate the whole plate and said it was good. Tina tried it herself, and realized it was disgusting. She was touched that he pretended it was good to spare her feelings.

Figure 1: Sample of ROCStories Dataset

basic criteria too. We have used cosine similarities to judge whether generated output resembles to the actual output. We have used average perplexity scores to see if the output generated is fluent in english or not. We have further employed technique whose underlying idea comes from Generative Adversarial Networks (GANs)[12]. We built a discriminator (logistic classifier) to see whether it is able to discriminate between real and machine-generated outputs.

Data Sources:

Short Stories: Story Cloze Test and ROCStories Corpora[8]

This data (shown in Figure 1) consists of ROCStories which has a title and 5-sentences for the story. Since, there are approx 90,000 stories, we trained on 80% of them and further tested and validated on 20% (using logistic classifier, refer to the Evaluation and Results section for details).

Essays: A concatenated corpus of highly insightful essays from Paul Graham[13]

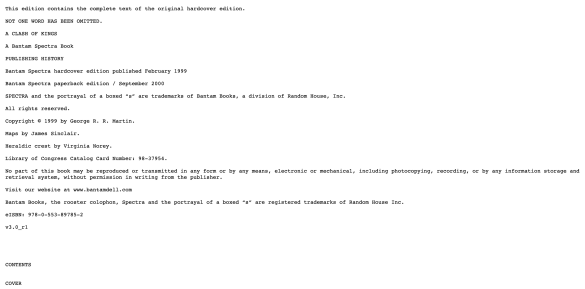
We trained our model on this data (shown in Figure 3) which consists of Essays by Paul Graham stored in a single txt file. For testing, we purposely used another essays dataset to see whether one writer's writing style can be compared to other writers' writing style. We tested on the dataset called The Hewlett Foundation: Automated Essay Scoring[1] where all essays were written by students ranging in grade levels from Grade 7 to Grade 10.

Books:Word Vectors Game of Thrones-LIVE[7]

This dataset (Figure 4) contains the first five books of The Song of Ice and Fire series by George R. R. Martin. We went a bit ambitious here and used first five novels as our training set to predict stories for sixth novel and compared them with the samples available online.

3 Related Work

As mentioned above, Recurrent neural networks (RNNs) has been the most popular way to solve the problem of text generation, or text completion. But since the inception of Transformer based



This edition contains the complete text of the original hardcover edition.
NOT ONE WORD HAS BEEN OMITTED.
A CLASH OF KINGS
A Bantam Spectra Book
PUBLISHING HISTORY
Bantam Spectra hardcover edition published February 1999
Bantam Spectra paperback edition / September 2000
SPECTRA and the portrayal of a boxed "s" are trademarks of Bantam Books, a division of Random House, Inc.
All rights reserved.
Copyright © 1999 by George R. R. Martin.
Maps by James Sinclair.
Heraldic crest by Virginia Norey.
Library of Congress Catalog Card Number: 98-37954.
No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage and retrieval system, without permission in writing from the publisher.
Visit our website at www.bantamdell.com
Bantam Books, the rooster colophon, Spectra and the portrayal of a boxed "s" are registered trademarks of Random House Inc.
eISBN: 978-0-553-89785-2
v3.0_r1
CONTENTS
COVER
TITLE PAGE

Figure 2: Sample of Game of Thrones Book Dataset

Networks, people have started experimenting this technique too for text generation. Not surprisingly, it is producing far better results than LSTMs. There are many papers which have utilised RNNs to perform story completion.

The Effect of Different Writing Tasks on Linguistic Style: A Case Study of the ROC Story Cloze Tasks[10] used logistic regression after generation to categorize an ending, either as right vs. wrong or as original vs. new. Writing Stories with Help from Recurrent Neural Networks[9] uses the evaluation functionality of Creative Help to compare RNNs to alternative approaches such as the described case-based reasoning method. Incorporating Structured Commonsense Knowledge in Story Completion[5] uses three types of information: narrative sequence, sentiment evolution and commonsense knowledge for story completion.

But Google launched one paper called Attention Is All You Need [4] in 2018 where they propose Transformer Networks based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Basis of BERT and GPT-2 both is actually attention-based mechanisms. Before the release of BERT there was GPT model(previous version of GPT-2) by OpenAI which trains a left-to right Transformer LM on a large text corpus. Infact, many of the design decisions in BERT were intentionally chosen to be as close to GPT as possible so that the two methods could be minimally compared. In Feb 2019, OpenAI published another paper called Language Models are Unsupervised Multitask Learners [3] introducing GPT-2 which is a 1.5B parameter Transformer that achieves state of the art results on 7 out of 8 tested language modeling datasets in a zero-shot setting. Samples from the model reflect improvements and contain coherent paragraphs of text which was also true in our case when we tried to predict paragraphs of text using Essays and Game of Thrones corpora. On reading comprehension the performance of GPT-2 is competitive with supervised baselines in a zero-shot setting. However, on other tasks such as summarization, while it is qualitatively performing the task, its performance is still only rudimentary according to quantitative metrics.

For evaluation, we referred Story Cloze Evaluator: Vector Space Representation Evaluation by Predicting What Happens Next [6] paper which gives unique methodology for evaluating the correct ending at paragraph and sentence level. The evaluator works as follows: given the vector representations of the two alternative endings and the four-sentence context as a whole, it rewards the embedding model if the contexts embedding is closer to the right ending embedding than the wrong ending. The closeness can be measured via cosine similarity of the embeddings. We utilized the same technique for evaluation in our paper. We have also used perplexity scores on the generated output and a logistic regression classifier, which is discussed in greater depth in the evaluation and discussion sections.

4 Methodology

Since our project predicts stories for three different types of datasets, we trained 3 different models for each of them. Below we will go into details for each of them separately but there are few things in common for all the three models we trained. Firstly, due to time constraints, we performed all the training and testing on Colab notebooks on Google Drive. Secondly, we didn't have to do any pre-processing like removing stopwords, tokenization etc because that would have led to loss of information while inputting the data into GPT-2 model and also the input representation of GPT-2 is Byte Pair Encoding (BPE) which doesn't require pre-processing. Lastly, the most important part, we have used the pre-trained embeddings of smaller (117M parameter) GPT-2 model released by OpenAI in Feb2018. Of course, it is less proficient than larger GPT-2 model (1.5B parameter) but still it was sufficient to produce results for our project. Also, since it was trained on many texts with biases and factual inaccuracies, and thus it is likely to be biased and inaccurate as well.

GPT-2 model is based on self-attention architectures like the Transformer Networks [4] which was first introduced by Google. Basically, it uses

encoder-decoder structure where encoder maps an input sequence of symbol representations (x1, ..., xn) to a sequence of continuous representations z. Then decoder generates an output sequence (y1, ..., ym). The difference is that at each step the model takes the previously generated symbols as additional input when generating the next. Both encoder and decoder comprised of several layers which work on self-attention mechanism. 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.

The results produced by GPT-2 model are state-of-the art results primarily due to the training set utilized in building the model. It was prepared by scraping web pages which have been curated/filtered by humans and created a dataset called WebText that contains the text subset of these 45 million links. Secondly, the input representation used was Byte Pair Encoding (BPE) which effectively interpolates between word level inputs for frequent symbol sequences and character level inputs for infrequent symbol sequences. Since this approach can assign a probability to any Unicode string, this allows to evaluate LMs on any dataset regardless of pre-processing, tokenization, or vocab size.

We fine-tuned this 117M model for three different datasets by utilizing the training code provided by nshepperd [11] on the GPT-2 repository on Github. Below are some details which are specific to each dataset:

Short Stories Corpus: For this dataset, we first combined short stories datasets from 2016 and 2017 to increase the size of the training dataset. Further, we concatenated all the five sentence fragments which were given separately to make the coherent 5-sentence story. Next, we divided the aggregated dataset into training(80%) and testing(20%). We trained the model on this dataset for 1139 epochs.

Essays Corpus: For this dataset, no pre-processing was required like Short Stories corpus. We trained our model on one dataset which consists of Essays by Paul Graham and tested it on another essays dataset on purpose to see whether we can replicate the results on different author's writing style. We tested on the dataset called The Hewlett Foundation: Automated Essay Scoring [1] where all essays were written by students ranging in grade levels from Grade 7 to Grade 10. The model was trained on this dataset for 1002 epochs.

Game of Thrones Corpus: For this dataset, we removed the first few pages of the novels like Appendix, Preface etc. Then trained our model on first five novels (for 1092 epochs) and predicted the stories for the sixth novel. We tested on the samples of three chapters (1 on Theon and 2 on Arianne) from the sixth novel called Winds of Winter.

All the materials utilized and code developed in this project can be found on this Github Repository: https://github.com/scarescrow/gpt-2

5 Evaluation and Results

It was difficult to come up with evaluation metrics for our models, given the level of problem it is trying to solve. Listed below are different techniques that we used for evaluation. Since we have trained different models for all the three corpora, we had to train different evaluation models (details listed below) for all 3 of them.

Logistic Classifier: The logic behind building such a classifier to test the accuracy of the model comes from Generative Adversarial Networks (GANs) [12]. Logistic classifier will work as the discriminator, which evaluates them for authenticity; i.e. the discriminator decides whether each instance of data that it reviews belongs to the actual training dataset or not. Main idea is to build such a classifier that will not be able to discriminate between actual and generated output which means that model is performing best when the machine-generated output starts behaving like the real one.

Corpora	Training	Testing
Short Stories	80% (test data)[1]	20% (test data)
Essays	80% (other data)[2]	20% (other data)
GOT	80% (3 chapters)[3]	20% of (3 chapters)

Table 1: Logistic Classifier

Similar to GANs, we input 50% real (ground truth) and 50% fake (machine-generated) outputs

[1]Here test data for short stories comes from the original samples which were divided into 80-20 ratio while training. We are using 80% of that 20% test data to train the logistic classifier

[2]Here, we have taken 80% of other data to train the logistic classifier because we wanted to test on another data

[3]Here, we have taken 80% of three chapters (1 on Theon and 2 on Arianne from the sixth novel called Winds of Winter) to train the logistic classifier because we wanted to test on another data

together (for the same input) to train our logistic classifier. Using this trained logistic classifier (one for each type of corpora), we further tested and reported test accuracy and F1-score.

Table 1, shows the details of training and testing data for building the Logistic classifier.

Perplexity: Perplexity is a measurement of how well a probability model predicts a test data. Minimizing perplexity is the same as maximizing probability. Therefore, low perplexity is good and high perplexity is bad which means the perfect model would when test data has perplexity of 1. So, we took out the perplexities for each sentence generated by our model and then averaged it to report the final scores. Here we would also like to highlight that perplexity reported for original GPT-2 model(developed by OpenAI) in the paper is 35.13.

Cosine Similarity: Apart, from the above approaches, we tried to replicate the evaluation from one of the papers (Story Cloze Evaluator: Vector Space Representation Evaluation by Predicting What Happens Next [6]). They basically calculate the average vector representation for the paragraphs/sentences and match with the context based on cosine similarity. On similar lines, we first generated the average vector of the stories predicted by our model, second generate the average vector of the actual story and compute the cosine similarity of the two.

Baselines: For the perplexity scores and cosine similarities, we took 2 baselines. The first one is a random sentence which was common for all the three corpora and for second baseline, we randomly picked one output sample generated from our model which is different for all three corpora.
First Baseline = "The quick brown fox jumped over the lazy dog"
Second Baseline:
Short Stories Baseline = "The new employee was nice and young."
Essay Baseline = "Most floors in single story buildings began with a series of blocks bolted to the floor."
GOT Baseline = "Neither were eating well enough unless he joined in their merry custom."

For the logistic regression classifier, we again took two baselines. In the first one, we randomly assigned a predicted class for each test data point. For the second baseline, we selected a class of 1 (which signifies that the text is machine generated) for all of the data points. Table 2 summarizes the results from the logistic classifier described above for all the three corpora with the 2 baseline results. As described above, ideally we would want a logistic classifier that will not be able to discriminate between actual and generated output which means that the accuracy will be 50% for the best performing model. We got 49% accuracy in both Short Stories and Game of Thrones corpus which means that our model is performing really well for both them but slightly poor on Essay corpus which has 66% accuracy.

Corpora	F-1 Score	Accuracy
Baseline 1(SS)	0.4278	43.06%
Baseline 2(SS)	0.6666	50%
ROC Short Stories	**0.4999**	**49.01%**
Baseline 1(E)	0.5024	49%
Baseline 2(E)	0.6666	50%
Essays	**0.6382**	**66.66%**
Baseline 1(GOT)	0.5148	51.48%
Baseline 2(GOT)	0.6666	50%
Game of Thrones	**0.5517**	**49.01%**

Table 2: Logistic Classifier Results

Below are the results from the perplexity and cosine similarity described above for all the three corpora with the 2 baseline results:

Corpora	Perplexity	Cosine Similarity
Baseline 1(SS)	2835.1219	48.47%
Baseline 2(SS)	376.9086	46.53%
Short Stories	**807.0305**	**53.93%**
Baseline 1(E)	2605.5371	39.89%
Baseline 2(E)	1389.9412	35.26%
Essays	**606.3497**	**47.60%**
Baseline 1(GOT)	308.1078	27.11%
Baseline 2(GOT)	378.3495	33.43%
GOT Corpus	**312.3814**	**47.87%**

Table 3: Perplexity and Cosine Similarity Results

Results show that perplexities are better than baselines in all the cases except Baseline2 in Short Stories. This is because the chosen baseline itself was the one generated by the model trained on short stories, and since we're only looking at one sentence, it gets the advantage of not being averaged over several generated sentences because when we evaluate it over the whole corpus, there are many sentences with different perplexities which tend to skew the overall perplexity score.

Figure 3: Sample Outputs generated for Short Stories and Essays

Figure 4: Sample Outputs generated for Game of Thrones

6 Discussion

The performance of the 117M model of GPT-2 seems to be pretty stellar. We have included some images which show the sample outputs generated by each of our models.

For the essay corpus, we tried to train on the essays of Paul Graham, while we generated text for essays written by students in the grades between 7 and 10. The evaluation metrics clearly show a drop in this particular use case as compared to the normal ones. It is interesting that a classifier can pick up on these subtle differences. The percentage of classification accuracy increases by almost 16% for the Essays corpus, which means that the distinction between machine generated and human written text is more evident in this case. However, we don't see a difference in the perplexity and cosine similarity scores. This shows that the fine tuned model is good enough to identify the contexts and generate words relative to them to get decent scores in these metrics.

In the case of Game of Thrones and Short Stories, it is interesting to note that the accuracy and F-1 are both at around 50%, which shows that the discriminative classifier is struggling to differentiate between the human written text and the model generated one. This has scary implications, because if the model is ever used for nefarious purposes (like generating fake news), it would make it even harder for algorithms to identify the real articles from the fake ones.

The fine tuned model for Game of Thrones is the best of the three in terms of perplexity, which shows that a limited corpus can greatly increase the coherence of the generated text. This is because the vocabulary of the Game of Thrones corpus is limited, as we only had five books to train the unigram model, on which the perplexity scores are generated. The short stories corpus have the highest perplexity, which could be because the short stories corpus uses short sentences and simple words, while the fine tuned model has some remnants of its training on the web pages dataset.

The cosine similarity scores show that only in the case of short stories the model is able to generate words that are strictly in the context of the actual continuation. It should be noted though that the similarities reported here were only for the generation of one sentence, and comparing that to the actual sentence. When we calculated the similarities for longer pieces of text (like a paragraph of generated text vs the next paragraph that was actually present in the corpus), we saw much higher similarities in all three models (between 60-65%).

One fascinating observation made by us was that the length and format of the text varied greatly based on the fine tuning. For example, for the short stories dataset, length of generated sentences were about four to five words, and the story was switched in three or four sentences, which is exactly like the corpus. Meanwhile, for the Game of Thrones corpus, we saw a dynamic shift in the length of the sentences, with many fantasy elements into it. The context is identified accurately, and we saw cases where we gave a seed with a character name, and found the same character being talked about, even in the second paragraph of the generated text.

Figure 3 and Figure 4 are captures from the poster we presented at the UMSI Spring Expo. They show the samples generated by each of the three models. The first section (Try your luck) in

Figure 3 were the short stories generated by the model, where the first sentence was the input seed. The Essays section of Figure 3 shows the text generated by the essay model (the input seed was the text in gold). Figure 4 shows the text generated by the game of thrones model, where the text in gold was the input seed.

For short stories and essays it was hard to identify that they were generated by machines on a quick look. However, if we inspect the generated text in the game of thrones section closely, we noticed that there is loss of coherence in the paragraphs, and even though it uses words from the books, the generated text itself is incomprehensible. On the flip side, for the generated short stories, it is able to logically continue the input seed and satisfactorily complete the story.

Given the success of fine-tuning GPT-2 model, we plan to further investigate fine-tuning by tweaking certain hyperparameters especially since it is unclear whether the additional training data and capacity of GPT-2 is sufficient to overcome the inefficiencies demonstrated by BERT.

7 Conclusion

We have seen that using the 117M model of GPT-2, we can generate well written text within small number of training epochs for short stories and essays which totally made sense. The model can replicate a small corpus of text that it has been trained on quiet accurately, and it is hard for a discriminator (logistic classifier) to separate generated text from authentic one.

However, the model is not efficient when it comes to mixing styles, that is if it is trained on a particular corpus, it is not that good at generating texts for a different but similar corpus.

It is indeed surprising to see how a small model with 117 million parameters can be so proficient at generating text specific to a corpus within a small amount of training epochs. In future, it would be enthralling to see the performance of the larger model with 1.5 Billion parameters, if ever OpenAI releases it.

8 What You Would Have Done Differently or Next

GPT-2 model involves lot of hyper-parameters which can be experimented with. So, if we had to start the project over we will definitely change certain parameters and evaluate the performance. In the current models, we haven't used the "endoftext" separator while concatenating the different novels of Game of Thrones and essays. Currently, we have not trained in batches, so we could not accumulate gradients as we have trained it only once. Also, we will experiment with batch size and learning rate. Apart from that, we will try to input different types of corpora like Music Lyrics, TV scripts etc. to see how GPT-2 model performs on them.

9 Acknowledgments

We would like to sincerely thank Prof. Jurgens for his valuable suggestions. When we started, he contributed primarily by suggesting how we should look for pre-trained models and not train RNN from scratch which led us to GPT-2 model. When we were stuck with evaluation, he suggested many ways to successfully evaluate our model.

References

1. Hewlett, W. a. (2012). The Hewlett Foundation: Automated Essay Scoring. Retrieved from Kaggle: https://www.kaggle.com/c/asap-aes/data
2. OpenAI(2019). https://github.com/openai/gpt-2.Retrieved from Github: https://github.com/openai/gpt-2
3.Alec Radford, J. W. (2019). Language Models are Unsupervised Multitask Learners. OpenAI.
4. Ashish Vaswani, N. S. (2017). Attention Is All You Need. Advances in Neural Information Processing Systems 30 (NIPS 2017).
5. Jiaao Chen, J. C. (2018). Incorporating Structured Commonsense Knowledge in Story Completion.
6. Nasrin Mostafazadeh, L. V.-t. (2016). Story Cloze Evaluator: Vector Space Representation Evaluation by Predicting What Happens Next. Proceedings of the 1st Workshop on Evaluating Vector Space Representations for NLP.
7. Raval, S. (2016). word-vectors-game-of-thrones-LIVE. Retrieved from Github: https://github.com/llSourcell/word-vectors-game-of-thrones-LIVE
8. Rochester, U. o. (2016). Story Cloze Test and ROCStories Corpora. Retrieved from Department of Computer Science, University of Rochester: http://cs.rochester.edu/nlp/rocstories/
9. Roemmele, M. (2016). Writing Stories with Help from Recurrent Neural Networks. Proceedings of the Thirtieth AAAI Conference on

Artificial Intelligence (AAAI-16).
10. Roy Schwartz, M. S. (2017). The Effect of Different Writing Tasks on Linguistic Style: A Case Study of the ROC Story Cloze Task. Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017), 1525.
11. Shepperd, N. (2019). Code for the paper ”Language Models are Unsupervised Multitask Learners”. Retrieved from Github: https://github.com/nshepperd/gpt-2
12. Skynet,A Beginner’s Guide to Generative Adversarial Networks (GANs). Retrieved from Skymind:https://skymind.ai/wiki/generative-adversarial-network-gan
13. Soni, N.(2018). Paul Graham Essays. Retrieved from Kaggle: https://www.kaggle.com/krsoninikhil/pual-graham-essays