Towards Adversarial Genetic Text Generation

Deniz Kavi

Text generation is the task of generating natural language, and producing outputs similar to or better than human texts. Due to deep learning's recent success in the field of natural language processing, computer generated text has come closer to becoming indistinguishable to human writing. Genetic Algorithms have not been as popular in the field of text generation. We propose a genetic algorithm combined with text classification and clustering models which automatically grade the texts generated by the genetic algorithm. The genetic algorithm is given poorly generated texts from a Markov chain, these texts are then graded by a text classifier and a text clustering model. We then apply crossover to pairs of texts, with emphasis on those that received higher grades. The approach described in this paper was designed to be as modular as possible and as such, changes to the grading system and further improvements to the genetic algorithm are to be the focus of future research.

1 Introduction

Text generation can be described as a "next word prediction" problem. This method of approaching text generating can be explained as, given a string of words, predict what the next word will be. Originally, text generation algorithms used a small part of the input text. For example, an algorithm might attempt to predict a word using just the previous two words of the sentence. However, since the information the algorithm has access to is limited, its ability to generate a coherent text is also limited.

Models that use "attention" to determine what parts of the text are relevant to what will follow. When trained on a large corpus of natural language, these models are currently the state of the art in natural language processing and text generation [8][1][12]. Attention allows neural network models to "pay attention" to only the relevant parts of the previous information. Whereas a Markov chain or a frequency based model would only have knowledge of some part of a sentence, neural networks using attention are able to pick required and relevant information from previous sentences. That being the case, models using attention can "remember" much more information than their predecessors as relevant words are more important than randomly picked words.

David C. Wyld et al. (Eds): CoSIT, AIAPP, SIGL, CRIS, NLPML - 2021
pp. 87-95, 2021. CS & IT - CSCP 2021 DOI: 10.5121/csit.2021.110407

Although researchers have used genetic algorithms for text generation [6][5], genetic algorithms have received little attention relative to deep learning approaches. In this paper, we propose an adversarial approach to the problem of text generation with genetic algorithms.

2 Related Works

In this section, we will evaluate previous approaches to the two components of our approach: text grading and generation. More specifically, we will be examining how effective they are at the target task and whether they could be applied to our own research.

2.1 Markov Chain for Text Generation

Markov chains can be represented as a sequence of states, where certain states come after each other. The order of the respective states is determined by the probability of of how these states are ordered in the input text dataset. And although any type of data, ranging from financial data to weather data can be represented in the form of states, we've represented text as a series of states, each word being a state. After the input text, in our case the ASAP dataset, is processed via the Markov chain, the model will be able to predict and generate the most probable word that will follow the it was given. We used Markov chains for our initial population, we used Markov chains because they performed less well compared to their deep learning counterparts. The reason we wanted it to perform worse was so that most of the task of text generation could be left to the genetic algorithm without causing problems in the grading system. As the grading algorithms hadn't seen samples of entirely randomly generated texts their ability to grade them would be low, so we used Markov chains that the grader could understand without lowering the genetic algorithm's contribution.

2.2 Transformer Language Models

Transformers are unsupervised machine learning models trained on a large corpus of the target language to predict the word or token which will follow the input text. Unlike Markov chains, transformers aren't constrained by the number of states they can have in memory, thanks to a parameter called attention. With emphasis on attention, a transformer model is able to remember relevant details of the previous text, which would be part of the text data that it pays attention to, allowing for it to recall relevant details without the need to analyze massive amounts of information.

2.3 Adversarially Learned Neural Outlines

In Subramanian et al. [11] the authors propose the usage of a generator, first adversarially producing a sentence outline and then generating words sequentially

conditioned by both the outline and previous outputs. This is inspired by GANs [4] and Autoencoder [10] models in that there are generator and discriminator neural networks. They fit "a non-parametric kernel density estimator(KDE) on the samples produced by a GAN and then evaluating the likelihood of real examples under this KDE."

2.4 Previous Genetic Algorithm Approaches to Text Generation

In Manzoni et al. [6], the authors describe a process by which they use word embeddings instead of the words themselves as input to the genetic algorithm. They first mapped every word of sample sentences of k length to a wor2vec vector, applying mathematical operations to the vectors and decoding the modified vectors, finally interpreting them as words. The mutation and crossover parts of a genetic algorithm would be done through linear algebra operations as the words are encoded as vectors.

2.5 Automated Essay Grading

An earlier system, the Intelligent Essay Assesor(IEA) [3], uses Latent Semantic Analysis [2] to grade essays. It measures and takes the sum of individual words' "meanings" to evaluate the whole passage's meaning. IAE compares the input essay to other essays in terms of the quality of its content and its form. The drawbacks with this approach is that the grader will be entirely unable to compare and thus, grade essays that it hasn't seen examples of.

A more artificial intelligence oriented system called IntelliMetric [9] uses manually determined syntactic and semantic features to feed into machine learning algorithms. This approach is very similar to ours in that the problem is essentially framed as a text classification task, but likely with a different dataset.

3 Methods

The primary difference of our approach to the problem of text generation is the use of a "grader", which is a supervised machine learning model trained on a dataset [1] of essays and their human graded scores. We used the training set with 12977 sample essay with labels(grades). The grades above 50(which there were 2 of) were changed to 50 to fit the way the classifier processed data.

We used an XLnet [12] based classifier, though any text classification or regression method would be usable for the dataset and the task of essay grading. XLnet is originally a language model trained on a large English corpus, we replaced the last layer of its architecture to a softmax layer to fit the task of text classification, so that it returns a grade when given a text as input.

[1] https://www.kaggle.com/c/asap-aes/data

To determine if the essays had topical consistency, we implemented a text clustering algorithm, which, without seeing the texts' labels would seperate samples into "clusters" based on similarities to other texts. The algorithm used was Scikit-learn's [7] "MiniBatchKMeans" model for clustering. The inputs to the clustering model were TF-IDF vectors for The Automated Student Assessment Prize(ASAP) dataset, with the stopwords removed. Instead of predicting what topic(cluster) each text belongs to, we only need to make sure that it belongs to any topic, as the model having a high confidence in the text belonging to a topic means that the text has a consistent topic, an attribute which should be rewarded. To calculate how closely a model follows its topic we take the reciprocal of the distance between the model's prediction and the cluster center, where predictions closer to the cluster center means that the model is more confident that the text belongs to a particular topic. The sum of the grade given to the essay by the text classifier and the text clustering model was used as the fitness function of the genetic algorithm, where those with higher fitness values will have a higher probability of passing their genes on to future populations.

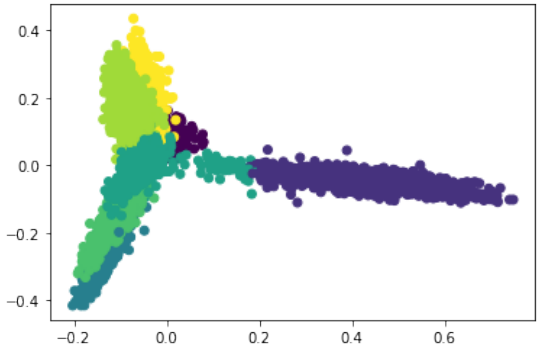

Figure 1: Clusters generated by the KMeans model, visualized in 2D

The Genetic algorithm first starts with a population of essays generated using Markov chains, which is a simpler and less successful approach simply working with the last word and picking the most probable word that would follow from its vocabulary. We used the open source markovify [2] library to implement this technique. We picked a weaker algorithm because our approach to grading these texts would not give reliable results on entirely random sequences of words or strings. Markov chains also increase the speed of the process as the children of the population are more likely to be readable if they are formed from reasonably readable parents.

This first population is then moved to the mating pool in relation to the scores they receive, for example a text that scored an 8 would be copied to the mating pool 8 times and one with a score of 1 would be copied only a single time.

[2]github.com/jsvine/markovify, accessed in August 2020

Doing so allows for a higher probability of texts with higher scores "breeding" and exchanging attributes or genes with other texts. Crossover is then applied to the population in the mating pool breed better future essays. Sentences of the two partners going through crossover are passed down to the child where the child would become a mix of the two parents. See Figure 2 as an example. In Figure 2, there is a 50 percent chance of a sentence from the first text and a 50 percent chance that the sentence is selected from the second text, in this specific example, by pure luck, sentences from the first text were selected more frequently. There also is a small probability(1/10) that individuals words in the sentences might be changed to those of the text's partner. Crossover is further explained in Algorithm 1. The actual program code was written in python.

Algorithm 1: Crossover

Data: array T_1 of sentences; array T_2 of sentences; float m, mutation rate

Result: array *child*, a combination of the two input texts

empty array *child*;
prob $\leftarrow$ random(0,1);
if $prob < 0.5$ **then**
  add sentence from T_1 to *child*;
end
else
  add sentence from T_2 to *child*;
end
if $m > random(0, 1)$ **then**
  replace a word in n sentence of *child* with corresponding word from T_1 or T_2
end

Algorithm 2: Text Generation with Genetic Algorithm

Data: integer f, minimum fitness required for the algorithm to stop; integer $maxGen$, maximum number of generation for the algorithm to stop

Result: Population of computer generated texts

```
generation ← 0;
create initial population of texts generated by Markov chains;
get fitness for each individual text in population;
while Fitness < f and generation < maxGen do
    add text as many times as its score to the mating pool;
    perform crossover;                    ▷ as described in algorithm 1
    empty population;
    add children to population;
    increase generation by 1;
end
```

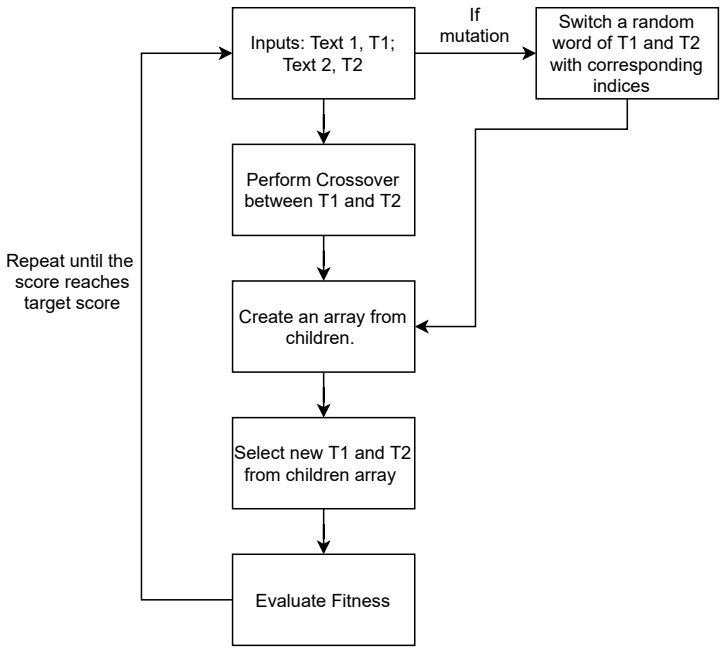

Figure 2: Flowchart of the Text Generation Process

He was always happy because he talks about what he does for me.Also, if a dirigible went down in the grass and birds chirping in trees and i knew i can save her from getting hurt from these actions. They should be things like music, tears and laughter have in common? you can contact the people there really get on her nerves.My personal opinion is that the weather can change, and it is commonly broken.Whatever doesnt @CAPS1 you makes you think why did they put it into words of how you might think people exercise computers.Must of the offensive material in libraries.He still had enough energy to turn off their computers, go exercise, be around nature and have some good healthy fun & work up a sweat.First off, people in this world so they can fully understand them.In fields, they use lead weights because it was very busy.If material were removed because someone said it was a night I will never forget it.

They then want to play out side and entered my apartment and went straight to his room and playing video games on computers begin to lose terms of reality.Also, you must have a parents permission.You have the right to ruin the movies that are for socializing, @CAPS11 messages also for People talk with one another, away.A friend of mine and she said go to some one esle and if its a laptop.Im only a @NUM1 year old sister goes on a computer all the time with vidio char, the most popular man or woman on the cover becasue one day they will be completly online.So don't take everything serious I mean it's a library they think of it.I think it is offensive to some.Censorship in libraries should not have the right to any information that they are quick and reliable, and they are great cooks and have skills for cooking.The nature of the wind and snow was terrible I had been laughing, and she ran to her room and she asked if I have the best hair.It would be hard though.This shows how thankful Narciso is to have an aircraft that low in cities.

They then want to play out side and entered my apartment and went straight to his room and playing video games on computers begin to lose terms of reality. Also, you must have a parents permission. He was always happy because he talks about what he does for me.They should be things like music, tears and laughter have in common? You have the right to ruin the movies that are for socializing, @CAPS11 messages also for People talk with one another, away.A friend of mine and she said go to some one esle and if its a laptop. you can contact the people there really get on her nerves. My personal opinion is that the weather can change, and it is commonly broken.Whatever doesnt @CAPS1 you makes you think why did they put it into words of how you might think people exercise computers.Must of the offensive material in libraries.He still had enough energy to turn off their computers, go exercise, be around nature and have some good healthy fun & work up a sweat.

Figure 3: Simplified Crossover Example

4 Discussion and Future Work

If the basic structure of our approach is kept, then there would be an initial text population, scored with an automated system and crossed-over with the other individuals within the population. The first attribute of the algorithm, the initial population, could be changed from Markov chains to entirely random sequences of words if the grader is able to work with random texts. Or any other population of texts that could be combined to generate more meaningful texts. The grading system could also be replaced with any system that would be able to grade medium length texts automatically at a reasonable speed. The implementation of crossover could also be changed, possibly with taking into account the fact that words can be represented as vectors with word embeddings. When words represented as vectors, arithmetic operations can also be applied to words to change their meanings or as a way to perform the steps of the genetic algorithm. More types of mutations could also be added to improve variety in word choice and order. In summary, the algorithm can be largely modified while most of its core properties can be kept. This should allow for further experimentation for text generation with genetic algorithms.

5 Conclusion

We demonstrate a proof-of-concept for a genetic algorithm for text generation using automated essay grading as the fitness function. The algorithm uses sample texts generated by Markov chains trained on the ASAP dataset and merges those samples with each other to produce texts with higher grades. Our system for grading is a combination of a text classifier and text clustering algorithm trained on the aforementioned dataset. The highest score achieved by the text generated by the genetic algorithm was 54/58.

Most of the components of the approach described may be changed based on differing needs. The process used to generate the initial text population may be changed from Markov chains to any process that outputs text. Future work may even create an entirely random initial population. We chose to use Markov Chains as they provided somewhat but not completely meaningful text. Experimentation with mutation rates should also provide an increase in the quality of the texts generated. The method used for evaluating the quality of the text is also replaceable as any process that gives a quantitative assessment of texts could be used as a fitness function. Other approaches to text evaluation could be added to the algorithms described in this paper, which likely would increase performance.

6 Acknowledgments

We thank Dr. David Perkins(Hamilton College) for his guidance and help throughout the development of this project.

References

[1] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165 [cs.CL].

[2] Peter Foltz. "Latent Semantic Analysis for Text-Based Research". In: *Behavior Research Methods* 28 (Feb. 1996), pp. 197–202. DOI: 10.3758/BF03204765.

[3] Peter Foltz, Darrell Laham, and T. Landauer. "The intelligent essay assessor: Applications to educational technology". In: *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning* (Apr. 1999).

[4] Ian J. Goodfellow et al. *Generative Adversarial Networks*. 2014. arXiv: 1406.2661 [stat.ML].

[5] Ruli Manurung, Graeme Ritchie, and Henry Thompson. "Using genetic algorithms to create meaningful poetic text". In: *J. Exp. Theor. Artif. Intell.* 24 (Mar. 2012), pp. 43–64. DOI: 10.1080/0952813X.2010.539029.

[6] Luca Manzoni et al. *Towards an evolutionary-based approach for natural language processing*. 2020. arXiv: 2004.13832 [cs.CL].

[7] Fabian Pedregosa et al. “Scikit-learn: Machine learning in Python”. In: *Journal of machine learning research* 12.Oct (2011), pp. 2825–2830.

[8] A. Radford et al. “Language Models are Unsupervised Multitask Learners”. In: 2019.

[9] Lawrence Rudner, Veronica Garcia, and Catherine Welch. “An Evaluation of IntelliMetric^{TM} Essay Scoring System”. In: *Journal of Technology, Learning, and Assessment* 4 (Jan. 2006).

[10] Juergen Schmidhuber. *Deep Learning in Neural Networks: An Overview*. 2014. arXiv: 1404.7828 [cs.NE].

[11] Sandeep Subramanian et al. “Towards Text Generation with Adversarially Learned Neural Outlines”. In: *NeurIPS 2018*. Dec. 2018. URL: https://www.microsoft.com/en-us/research/publication/towards-text-generation-with-adversarially-learned-neural-outlines/.

[12] Zhilin Yang et al. *XLNet: Generalized Autoregressive Pretraining for Language Understanding*. 2019. arXiv: 1906.08237 [cs.CL].