Training GPT-2 to represent two Romantic-era authors: challenges, evaluations and pitfalls

Piotr Sawicki[1], Marek Grześ[1], Anna Jordanous[1], Dan Brown[2], Max Peeperkorn[1]
[1] School of Computing, University of Kent, Canterbury, UK
[2] Cheriton School of Computer Science, University of Waterloo, Canada
P.Sawicki@kent.ac.uk, M.Grzes@kent.ac.uk, A.K.Jordanous@kent.ac.uk, Dan.Brown@uwaterloo.ca, mp770@kent.ac.uk

Abstract

Poetry generation within style constraints has many creative challenges, despite the recent advances in Transformer models for text generation. We study 1) how overfitting of various versions of GPT-2 models affects the quality of the generated text, and 2) which model is better at generating text in a specific style. For that purpose, we propose a novel setup for text evaluation with neural networks. Our GPT-2 models are trained on datasets of collected works of the two Romantic-era poets: Byron and Shelley. With some models, overfitting manifests by producing malformed samples, with others, the samples are always well-formed, but contain increasingly higher levels of n-grams duplicated from the original corpus. This behaviour can lead to incorrect evaluations of generated text because the plagiarised output can deceive neural network classifiers and even human judges. To determine which model is better at preserving style before it becomes overfitted, we conduct two series of experiments with BERT-based classifiers. Overall, our results provide a novel way of selecting the right models for fine-tuning on a specific dataset, while highlighting the pitfalls that come with overfitting, like reordering and replicating text, towards more credible creative text generation.

Introduction

Contemporary text-generation systems can create output whose surface features strongly resemble the source materials upon which they are trained. Such generative systems can have credibility in computational creativity research as they can be demonstrated to possess knowledge and produce novel and valuable results in a directed fashion (Ventura 2016). As described below, there is a growing body of work that has recently been emerging in this direction in Natural Language Processing (NLP) research, via the growing popularity of OpenAI's GPT (Radford et al. 2018; 2019; Brown et al. 2020). While GPT-based systems for stylistic reproduction have attracted some criticism (Falk 2021; Floridi and Chiriatti 2020), in general, their results have been impressive and deserve further attention in computational creativity research (Dale 2021; Köbis and Mossink 2021).

Soon, these systems could generate new works in the style of authors from previous eras, perhaps even inspired by current events or social movements. However, before this can become reliably possible without relying on human supervision to cherry-pick the results, we need to learn how well these new transformer-based systems can be trained, and what pitfalls exist with them. In particular, we need to know what are the differences in performance between various versions of the models, to allow for optimal selection. Here, we describe some steps toward these aims.

Ideally, we would like to conduct a large scale human evaluation of GPT-2 produced text, but such evaluations are prohibitively costly and difficult to organize for most researchers, therefore in this study we are focused almost entirely on automated evaluations. For the first objective of this study—detection of over-training of GPT-2 models—we perform a visual evaluation of the samples to watch for malformed text, and then we perform the BLEU (Papineni et al. 2002; Yu et al. 2017) evaluation of the samples to watch for excessively high levels of similarity (on the n-gram level) of the samples to the original dataset. For the second objective, which is to investigate which GPT-2 model performs best at the task of generating text in specific authors' style, we use BERT (Devlin et al. 2018), which is currently state-of-the-art in text classification, to identify texts that appear closer to the source material than to the output of GPT-2 models.

Poetry generation has long been an area of interest in computational creativity research (Lamb, Brown, and Clarke 2017; Oliveira 2009; 2017). Previous work includes the use of machine learning (Das and Gambäck 2014; Loller-Andersen and Gambäck 2018; Rahman and Manurung 2011), mining of corpora or other source material as inspiring examples for stylistic reproduction (Gervás 2011; Lamb and Brown 2019; Rahman and Manurung 2011; Toivanen et al. 2014) as well as approaches such as expert-based distributed systems (Corneli et al. 2015; Misztal and Indurkhya 2014), the use of constraints (Rashel and Manurung 2014; Toivanen et al. 2013), evolutionary approaches (Manurung 2004) and knowledge-based/linguistic models (Hämäläinen 2018; Oliveira and Cardoso 2015; Veale 2013). Style imitation systems have also attracted attention in text generation (Alvarez Cos, Perez y Perez, and Aliseda 2007) and other domains (Ens and Pasquier 2018; Pachet and Roy 2014), though we note that attention has also been paid to the creativity required to deviate from a

given style (Elgammal et al. 2017).

The growing attention being paid to GPT-based approaches in NLP research is beginning to get replicated in computational creativity. Here, we could mention a few notable examples where GPT-2 or BERT were applied to poetry generation: (Liao et al. 2019) have fine-tuned GPT-2 to generate Chinese classical poetry, (Köbis and Mossink 2021) have conducted an extensive human evaluation of GPT-2 generated English poetry, (Li et al. 2020) have experimented with applying rigid constraints in generation of both Chinese and English poetry, (Wöckener et al. 2021) have analysed the problems with maintaining rigid stylistic constraints in poetry generation while using RNN and GPT-2, and (Nikolov et al. 2020) and (Oliveira 2021) have explored a transformative, BERT-based approach to lyrics generation. Lyrics were also generated from GPT-2 and evaluated using BERT in (Wesek 2019).

The scope of this study is focused on poetry generation in a general language style of a specific author, learning from a corpus of poetry by two highly-regarded English poets: Lord Byron (1788-1824) and his contemporary Percy Bysshe Shelley (1792-1822). Here, we tackle poetry generation by fine-tuning GPT-2 models on the collected works of both authors.

GPT-2 and BERT models

The GPT and BERT models are derived from the Transformer architecture (Radford et al. 2018; Devlin et al. 2018), which is a form of an Encoder-Decoder model, where RNN or LSTM networks have been replaced by multiple layers of attention mechanisms (Bahdanau, Cho, and Bengio 2014; Vaswani et al. 2017), thus allowing all input to be processed simultaneously by dispensing with sequentiality. The original Transformer model followed the Encoder-Decoder architecture because it was intended for machine translation, where we first have to encode the source language sequence, and then decode it into the target language sequence (Sutskever, Vinyals, and Le 2014). Most other NLP tasks, however, do not require this kind of setup, and subsequently, the Encoder and Decoder blocks started to be used separately. The Decoder block was first developed by OpenAI into the Generative Pre-trained Transformer (GPT), and soon later the Encoder block was developed by Google into the Bidirectional Encoder Representations from Transformer (BERT). The first editions of GPT were released in two versions: small and medium, and have managed to advance the benchmarks on many NLP tasks (Radford et al. 2018). BERT was also released in two versions, which roughly matched the size of the small and medium GPT models to facilitate comparison. BERT has proven superior to the matching GPT models in Natural Language Understanding, while GPT excelled in Natural Language Generation (Devlin et al. 2018). This was expected because of the specific differences between the architectures of the Encoder and Decoder transformer blocks.

While the original Transformer (i.e. the translation machine) required separate training for each language pair, both GPT and BERT follow the transfer learning paradigm (Radford et al. 2018; Devlin et al. 2018). Both of them are first pre-trained on the large corpora of text (ranging from 5GB to 800GB, depending on the version). This initial training is very demanding in terms of time and the required hardware. The model can then be used "out of the box", or can be fine-tuned for a specific task and that is where transfer learning actually comes to play. The fine-tuning process is much faster and requires much less powerful hardware. After fine-tuning, the model can be used for a destined downstream task, using additional layers, which are referred to as "heads", that accomplish those tasks (Radford et al. 2018; Devlin et al. 2018).

The consecutive GPT versions use the same general architecture, but with much larger numbers of layers and "attention heads" (attention mechanism working in parallel). They are also trained on increasingly large datasets. Currently, the latest version of OpenAI's GPT is GPT-3; however, it can only be used through OpenAI's API. While the use of the base GPT-3 models through an API is free, fine-tuning of the model (at the time of writing this paper) has to be paid for. There exist other large language models, already exceeding the size of GPT-3, for example: Gopher (Rae et al. 2021), Megatron-Turing NLG (Smith et al. 2022), and Jurrasic-1 (Lieber et al. 2021), which, if available to the public at all, can only be used via their APIs. The hardware requirements of these models are way above of what researchers at most universities can access. Here, we should also mention the models released by EleutherAI: GPT-J-6B (Wang and Komatsuzaki 2021) and GPT-NeoX-20B (Black et al. 2022). However, their hardware requirements, while smaller than those of the models mentioned above, still exceed the hardware we can access.

This study was therefore carried out using GPT-2, which we could fine tune on our hardware. Applying GPT-3 and other large-scale models to poetry generation is left for future work. It is not unreasonable to expect that the results we obtained using GPT-2 will translate to larger language models when they become more accessible.

At present, there are two different applications of GPT-2 available. The original edition of GPT-2 is by OpenAI (Radford et al. 2019; OpenAI 2021). The source code of the interface for this version was later propagated with some changes by (Shepperd 2021), and as such it was used in research, for example (Lee 2019; Lee and Hsiang 2020b; 2020a). The second application is in the Transformers library (Transformers Documentation 2019; Wolf et al. 2019). This edition introduced significant changes to the code of the interface, making the process of fine-tuning and generation easier. We are using both applications in our experiments.

The GPT-2 models from the Transformers library that are used for text generation are referred to in their library as "Language Modelling Head" models (Transformers Documentation 2019), therefore in order to distinguish them from OpenAI models, in this paper, we refer to them as "LMH" models, while the OpenAI versions are referred to as "Regular". These two implementations differ significantly. The LMH models have a standard learning structure of training epochs, where each epoch has a specific number of steps depending on the size of the dataset. The Regular models do not use epochs in training. Additionally, training of the

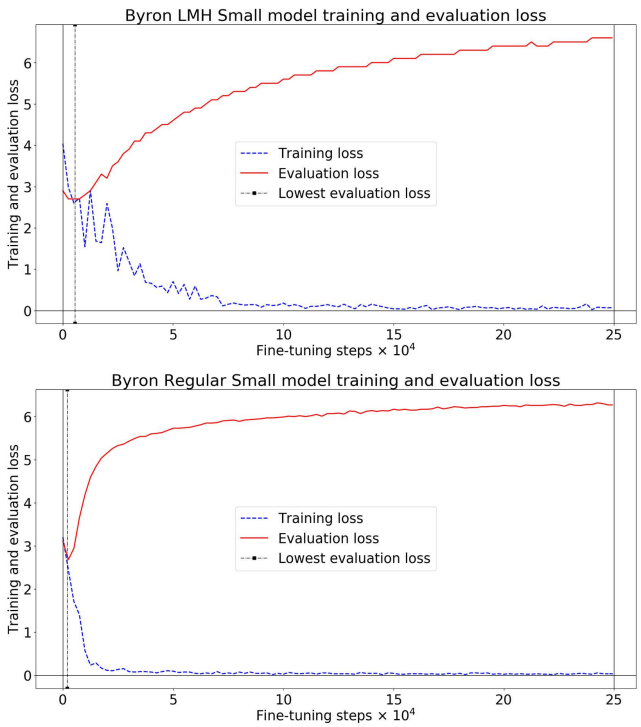

Figure 1: Training and evaluation loss for the LMH Small model (Top) and the Regular Small model (Bottom) fine-tuned for 250K steps on the dataset of Collected Works of Lord Byron.

LMH models is much faster per training step as compared to the Regular models. The Regular models are released in four versions: Small (124M parameters), Medium (345M parameters), Large (774M parameters) and XLarge (1558M parameters). The LMH models, at the time of writing this paper, were released only with Small, Medium and Large versions, with the same number of parameters as the respective Regular models. Due to hardware limitations, we only fine-tune the Small and Medium models, and as a result, we use four models in total for each task: LMH Small, LMH Medium, Regular Small and Regular Medium. Experiments with the Large and XLarge models are left for future research. It has to be noted that, at this point, we are not experimenting with adjusting hyperparameters, neither during fine-tuning nor during sample generation. The default *top_k* for both models is 50, default *temperature* is 1, default *top_p* is 1 for Regular models, and 0.9 for LMH models. For consistency, we have set *top_p* value for LMH models to 1. The train/test split of the datasets is 70/30.

Figure 1 shows the training loss and evaluation loss for the Regular Small and LMH Small models fine-tuned on the Byron dataset for 250K steps (results for medium models and for models fine-tuned on the Shelley dataset are very similar, and therefore are not presented here). We can see that the lowest evaluation loss is achieved very early in the fine-tuning process: for LMH models, this occurs around 5000 fine-tuning steps, and for the Regular models even sooner, around 1700 steps. We believe this is because the datasets

Author	Org size	Length	Final size	Length
Byron	7.2 MB	183643	2.4 MB	62947
Shelley	2.3 MB	59207	0.98 MB	29151

Table 1: Details of the datasets including original size, original length (lines), pre-processed size, pre-processed length (lines).

used for fine-tuning are relatively small, and the models become overfitted fairly early. In this study, we investigate whether the point of the lowest evaluation loss is optimal for early stopping of the fine-tuning process, and to get deeper insights into the behaviour of the GPT-2 models, we evaluate the actual quality of the generated samples. To that end, we conduct a number of evaluations of samples generated at specific checkpoints. This will be further described in the later sections.

Data preparation

Original datasets

Our main interest is generation and evaluation of poetry in the style of a specific author. For that purpose we have chosen two Romantic-era poets: Lord Byron and Percy Bysshe Shelley. The datasets for both of them were created from their collected works, downloaded from Gutenberg.org (Project Gutenberg 2020), by removing all introductions, forewords, footnotes, generic Gutenberg text at the end of the file, replacing all extended Latin characters with a closest matching ASCII character (for example“ Ã” is replaced with “A”, “á” with “a”, etc.). Sequences of multiple blank lines in the original text were replaced by a single blank line. We have removed all metadata, i.e., page numbers and the end of line verse numbers. We have left the poems’ titles and chapter numbers, since they contribute to author’s ”style”, but also to preserve the separation of individual poems. Being aware of the destructive impact that artefacts of poor pre-processing of data can have on the output of GPT-2, we have paid particular attention to the task of data preparation. Both the original and the pre-processed datasets can be found on our online repository[1].

Additionally, we have removed all plays, thus leaving only poetic works. The purpose of this pre-processing was to leave only the poetic text written by the authors themselves.

Setup 1 for visual and BLEU evaluations

We fine-tune all four GPT-2 models used in this study (Regular Small, LMH Small, Regular Medium, LMH Medium) on both datasets (Byron and Shelley). For the visual and BLEU evaluation in Experiments 1 and 2, we fine-tune the GPT-2 models for 250K steps, generating 100 samples at each 10K steps interval. The samples are generated with a length of 1000 tokens (the maximum sample length for these models is 1024 tokens). We generate only 100 samples at

[1]Our datasets and example outputs generated by GPT-2 are available at:
`https://github.com/PeterS111/GPT-2-for-Byron-and-Shelley`

each checkpoint because of the time it takes to generate full-length samples (for example the LMH Medium model running on Nvidia P100 GPU takes around 2 minutes per sample). Thus we obtain 8 sets of 2500 samples, four for each author.

Setup 2 for BERT evaluations

Datasets for the visual and BLEU evaluations created in Setup 1 have an insufficient number of samples per checkpoint (only 100) to be used for training the BERT-based classifiers. For this reason, we create a separate set of 8 datasets by fine-tuning all of our GPT-2 models on both datasets for 10K steps and generate 1K samples at each 1K steps checkpoint. We have chosen this span of checkpoints because it covers the sweet spot where the evaluation error is at the lowest, and we can observe the quality of the samples immediately before and after that point. Thus we obtain 8 datasets of 10K samples. The samples are limited to 600 tokens, since as we explain later, we use only the first 20 lines of each sample in Experiments 3 and 4.

Part 1—Evaluation of the overfitted models

In this section, we analyse the impact of overfitting on the quality of the produced samples. This is to emphasize the importance of early stopping of the fine-tuning process and to explore how the existing quantitative metrics (such as BLEU) correlate with the overfitting of GPT-2.

Experiment 1—Visual evaluation of text quality

During this research, we have observed that while generating text with the full length possible, many samples come with significant errors. We have decided to establish whether there is any regularity in the production of the malformed samples. For this purpose, we analyse the datasets from Setup 1. From every 100 samples generated at a specific checkpoint, 10 samples are selected at random and evaluated manually by the authors using the following procedure: when looking at the sample, we check at which point within the sample the text becomes unintelligible or contains obvious errors. We take note of the line number where this happens, and we take note of the total number of lines in the sample (including the blank lines at the end). Then we calculate the percentage of the correctly produced lines. After that, we calculate the average value for those 10 samples. We repeat this for all 25 checkpoints. The results for both datasets are shown in Figure 2. Examples of correct and malformed samples generated in our experiments are available on our repository.

We can see that the Regular models score almost 100% across the whole range of fine-tuning checkpoints. For the LMH models, the percentage of correctly generated text within a sample is at its best at 10K and 20K checkpoints, and after that, it rapidly decreases to around 35% for the remaining checkpoints.

We have observed that once the errors start appearing in the samples generated by the LMH models, the remainder of the sample is almost always malformed. In contrast, the Regular models occasionally produce a few malformed lines

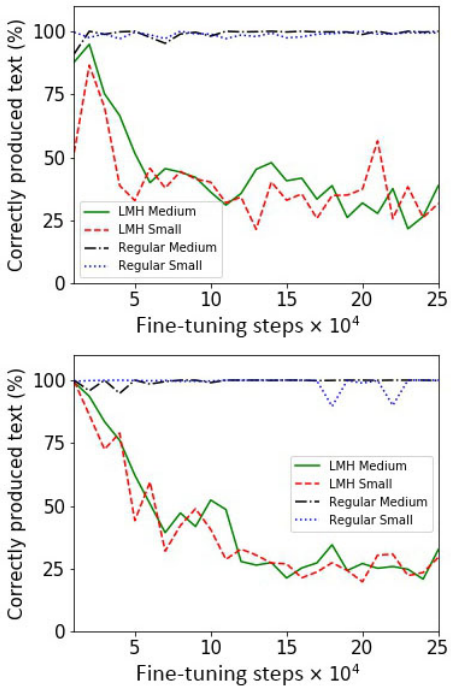

Figure 2: Results of the visual evaluation of text quality for samples generated from the Byron (top) and Shelley (bottom) datasets.

in the middle of the sample, but the subsequent text is consistent again. The LMH models' output does not have this "self-repairing" property. We are aware that these results could be different if much larger or much smaller datasets were used for fine-tuning. The reason we chose datasets of this size is because of our objective of style preservation of an individual poet.

A well-formed sample with a Byron-style text of 1000 tokens usually spans around 45 to 80 lines in a text file. However, the malformed samples from the LMH models could sometimes exceed 180 lines, of which often only around 30 lines at the top of the sample are of good quality. With the Shelley-style samples, the malformed samples can exceed 250 lines, with more or less the same proportion of correctly produced text. This is because the number of tokens per line plummets, and many blank lines are inserted into the sample. This "stretching" phenomenon was not observed in the samples produced by the Regular models. One could raise a question whether these results were caused by poor data pre-processing. This, however, is unlikely since our data preprocessing was rigorous and comprehensive, as described in the section on Data Preparation. Given the quality and fineness of data used for fine tuning, it is clear that the errors in the samples must be caused by the properties of the models themselves.

Because of the malformed text in long samples, in the subsequent experiments with the BERT-based classifiers, we will limit the sample length to the first 20 lines of text from each sample. This is because our goal in the second part of this study is to evaluate only well-formed outputs, instead of

learning to spot obvious errors, like repeated lines, garbled and unintelligible text, etc.

The results of the visual evaluation in Figure 2 show a deficiency of the LMH models after 10K-20K training steps with these specific datasets. When these results are contrasted with Figure 1, one can notice a clear correlation between overfitting—quantified by a high evaluation loss in Figure 1–and the ratio of malformed lines in Figure 2. Such a correlation in not present in the results of the Regular models, however, and their results in Figure 2 do not detect any malformed text according to the visual evaluation. For this reason, we perform a BLEU evaluation in the next experiment to see if the effect of overfitting can be uncovered in the samples from the Regular models.

Experiment 2—What does the BLEU evaluation tell us?

The visual evaluation has informed us that samples from the Regular models appear to be correctly produced across all the checkpoints from 10K to 250K fine-tuning steps, regardless of the increasing evaluation loss between those checkpoints. But, are there any noticeable and measurable changes in text quality between those checkpoints? In order to establish that, we perform a BLEU evaluation of those samples against the original dataset.

Bilingual Evaluation Understudy (BLEU) was originally designed to evaluate machine translation (Papineni et al. 2002), where we have the master translation of the sentence in the source language, and a candidate translation produced by the system. BLEU compares and counts the matching n-grams between the master and the candidate, resulting in a score between 0 and 1. The closer the score is to 1, the better for the translation task because this indicates that the candidate translation is more similar to the master translation. While not designed for that, BLEU is sometimes used to evaluate text quality (Yu et al. 2017), but when used for this purpose, it suffers from several deficiencies. Our objective, however, is to measure only the n-gram based similarity between the samples and the original text. We therefore expect that BLEU is an appropriate algorithm for our application because we have two types of text to compare, albeit we interpret the scores differently. Unlike in the translation tasks, in our research, we are aiming at a lower score, which indicates that fewer n-grams in the sample were copied from the original dataset, thus demonstrating a higher level of originality. In other words, we treat the BLEU score as a plagiarism detector on the n-gram level, which would be quantified by a high score. We use it to explore if there are any trends in the averaged BLEU scores between consecutive checkpoints. In our application of BLEU, we score each sample against the original dataset, and then average the results, similarly to what was proposed by (Yu et al. 2017) and implemented by (Wesek 2019). The implementation of BLEU used in our code was from the NLTK library[2].

Like in Experiment 1, we follow Setup 1, and we use 100 samples for each checkpoint from 10K to 250K. Samples

[2]Natural Language Toolkit Documentation: `https://www.nltk.org/`

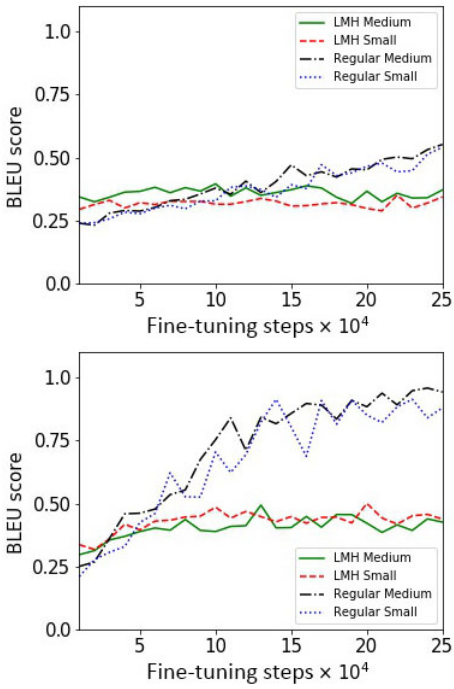

Figure 3: BLEU scores for Byron (top) and Shelley (bottom) calculated for samples with 1000 tokens length.

have the length of 1000 tokens. We compute BLEU for all the 100 samples at each specific checkpoint, and we take the mean of those values to obtain a single value per checkpoint. Figure 3 shows that the BLEU score consistently increases with the model fine-tuning steps for both Regular models. This indicates an increasing similarity of the samples compared to the original dataset, when the models are fine-tuned for longer. This increasing BLEU score basically means that GPT-2 plagiarizes n-grams from the original dataset. On the other hand, the BLEU scores for samples from the LMH models do not increase in the same way. This is because of the increasingly high amount of the malformed text, which prevents the BLEU score from rising. When we evaluated the samples truncated to 200 tokens (which discarded all the malformed text in the samples from the LMH models), the increase of the BLEU scores for both types of models was very similar. In other words, they consistently rose with the number of fine-tuning steps.

Altogether, we have observed that overfitting in the LMH models is easy to spot because the samples are malformed in an obvious way, but we must remain cautious about overfitting the Regular models, where the results of overfitting are not noticeable by our visual evaluation (Figure 2). Furthermore, the samples from the Regular models appear to be well-formed across all the checkpoints, while containing increasingly higher levels of plagiarized text or n-grams. This means that both automated and human evaluators could be misled by such outputs, since we cannot expect them to have memorized the original text. As a result, it is advisable to stop the fine-tuning process when the evaluation loss is

minimised or soon after when the samples start to be well produced after the initial learning of the model. Excessive training can lead to plagiarised n-grams, with even entire blocks of text repeated in GPT-2's output for the numbers of fine-tuning steps above 100K on our datasets.

A base requirement for GPT-2 to be creative is that it creates semantically and syntactically correct output. However, one way for it to do so is to just copy the source material, so high BLEU scores (or another measure for plagiarism detection) can indicate that the system is moving away from the novelty, which is also fundamentally necessary for creativity. As such, using a system like BLEU can be helpful in experiments with GPT-2-based poetry.

Part 2—BERT evaluation of correctly produced samples

In this section, we perform two experiments with a BERT-based binary classifier to establish which of the four GPT-2 models used is best at replicating the authors' style.

Experiment 3—Can fine-tuned GPT-2 outwit BERT?

In this experiment, we aim to verify if GPT-2's outputs can be of a sufficiently high quality to confuse a BERT-based classifier trained to distinguish the generated text from the original work of a poet. The experiments in the previous section have warned us about the plagiarized text (n-grams) in the samples from the Regular models. This behaviour, however, starts to become prominent late into the fine-tuning process, which, in our case, is well after 10K fine-tuning steps. For this reason, the samples we evaluate in this experiment are produced from checkpoints at 1K to 10K fine-tuning steps, which is before the plagiarism starts having significant impact.

Our previous experiments with the visual and BLEU evaluation have informed us that in the samples from the LMH models, for most checkpoints, only the first part is of high quality, and therefore, in this experiment, we use only the first 20 lines of each sample. This is because our intention is to apply BERT to the text that is correctly produced, and not learn to spot obvious mistakes, like in the malformed samples. The original text of the author is also split into 20-line fragments before it is fed into BERT, both for learning the classifier and for prediction. For each BERT-classifier, we prepared a dataset of 1K samples for each label, giving 2K samples in total, where label 0 is for samples from the original dataset, and label 1 for the samples generated by GPT-2. The train/test/validation split is 70/25/5. We use "bert-base-uncased" from the Transformers library, which is trained for 20 epochs, with the Adam optimizer, and a learning rate of 2e-5. The average classification accuracy on test data is taken as a final result of classification. Since our classification problem is balanced, i.e. the input contains the same amount of samples for both labels, we do not need to calculate Precision, Recall and F1 scores, and we can rely solely on accuracy. As described in the section on data preparation, we train the GPT-2 models for 10K steps and generate

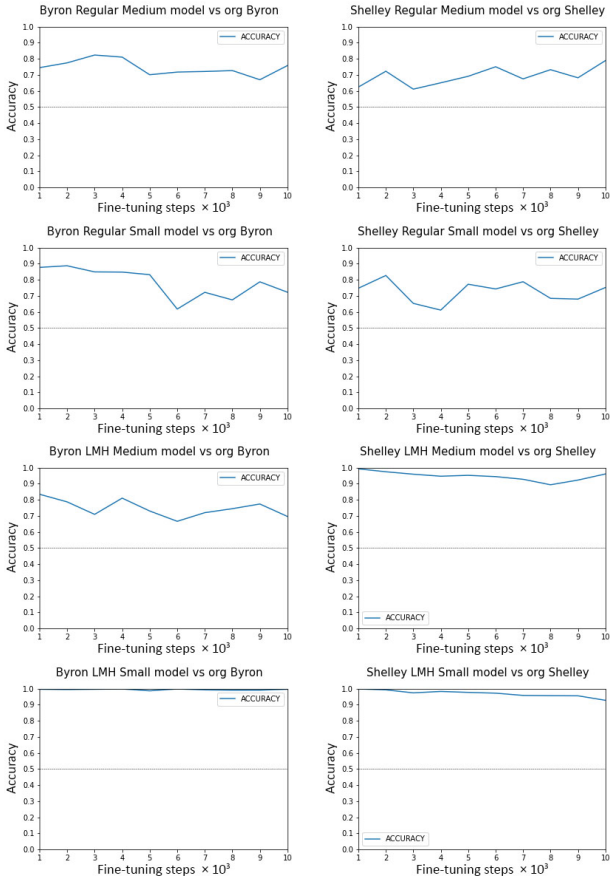

Figure 4: Results of the BERT evaluation of the Byron (left) and Shelley (right) checkpoint samples produced by the four different GPT-2 models. Row 1 (top): Regular Medium models, Row 2: Regular Small, Row 3: LMH Medium, Row 4 (bottom): LMH Small.

1K samples at each 1K steps interval. This scope of checkpoints encompasses the best evaluation loss for both types of models (Figure 1). This allows us to observe the changes in classification results for 10 checkpoints, with a separate dataset of 2×1K samples for each checkpoint. Thus we train ten BERT-classifiers for each dataset/GPT-2 model pair.

It is important to note how we interpret the results. In most applications, the closer the values of accuracy are to 1, the more desirable the performance of the classifier is. In this experiment, however, the GPT-2's output will be deemed to be indistinguishable from the original text, when the BERT classifier behaves like a random classifier. We know that a random classifier has an expected accuracy of 0.5 in a two-class problem. On the other end, an accuracy of 0 would mean the model still distinguishes between classes. For these reasons, the accuracy of 0.5 is our desirable target that can be seen as the evidence of GPT-2's high performance in generating high quality text. We can think of this as a sort of a "Turing test", which is successful when the BERT classifier behaves like a random classifier. This adversarial evaluation approach has been used before in NLP with both human (Köbis and Mossink 2021) and automated evalua-

tions (Bowman et al. 2015).

Figure 4 shows the classification results for Byron and Shelley's datasets generated from all four GPT-2 models used in this study. The results show that the Regular models perform well on all checkpoints on both datasets, but interestingly the Regular Small model required 6K steps to reach its optimal performance, while its lowest evaluation loss is at 1700 steps (Figure 1). This indicates that we cannot rely on the evaluation loss alone, but instead, we may want to analyse the models' output to establish the optimal early stopping time. The LMH Medium model performs well on Byron, but very poorly on Shelley. The LMH Small models have the lowest scores on both datasets. Thus, the Regular models appear to be a more reliable choice.

All models appear to have similar, stable performance across all 10 checkpoints (Figure 4), and thus these results do not correlate with the evaluation loss. This is because they are for the numbers of fine-tuning steps when no strong overfitting is observed (Figure 1) and the BLEU scores did not increase significantly yet (Figure 3).

In the next experiment, we use BERT and Setup 2 again to compare the GPT-2 implementations, but using a different experimental design.

Experiment 4—Which GPT-2 is better at replicating the author?

In the previous experiment, we were classifying samples from the original dataset (label 0) against samples from a specific GPT-2 model (label 1). This gave us some indication as to which model is better at replicating the authors' style.

Here, we propose a novel setup for text evaluation with the BERT-based classifier. This time we take samples from two different GPT-2 models, which we assign labels 0 and 1, and we classify against them only the samples from the original author's writing. The accuracy is averaged, and it indicates to which models' output the samples from the original are closer.

To train the classifier, we use the dataset of 1K samples generated from two different GPT-2 models in a given pair after 10K steps of training. We selected this number of fine-tuning step because (according to our previous experiments) both the evaluation loss (Figure 1) and the BLEU scores (Figure 3) show that we are not comparing overfitted models that plagiarize the original works. As explained before, only the first 20 lines of each sample are used in this experiment (see Setup 2). Just like in the previous experiment, we use "bert-base-uncased" from the Transformers library, which is trained for 20 epochs, with the Adam optimizer, and a learning rate of 2e-5. Every classifier is tested on an additional test dataset of 1K samples randomly selected from the original authors' corpus, each sample 20 lines in length. The results are averaged, giving a single value of accuracy. This value indicates which label the original dataset is closer to, i.e., which GPT-2 generates text more similar to the original work. Since we have four different models, we can create six possible pairs (Table 2 and Table 3) for each dataset.

Since the class labels are 0 and 1, Tables 2 and 3 can be interpreted in the following manner: when the score is smaller than 0.5, then the model listed in the left column wins, and conversely, when the score is greater than 0.5, then the model in the right column is the winner.

Tables 2 and 3 show that the Regular Medium model wins on both datasets. On the Byron dataset, the Regular Medium model is clearly the best, Regular Small and LMH Medium are both second best and appear to have very similar performance, while LMH Small scores the lowest. This is consistent with the findings from the previous experiment in which the Regular models led to better results. Regarding the Shelley dataset, the Regular Medium model again performs the best, but the other three models have similar performance. This could indicate that the LMH Small model performs better on the Shelley dataset because it is much smaller than the Byron dataset.

Label 0	Label 1	Score
Regular Medium	LMH Medium	0.32
Regular Small	LMH Small	0.08
Regular Medium	Regular Small	0.32
LMH Medium	LMH Small	0.09
Regular Small	LMH Medium	0.51
Regular Medium	LMH Small	0.08

Table 2: Results of Experiment 4. Byron's work is classified using BERT models trained on two types of GPT-2 generated data.

Label 0	Label 1	Score
Regular Medium	LMH Medium	0.31
Regular Small	LMH Small	0.49
Regular Medium	Regular Small	0.28
LMH Medium	LMH Small	0.54
Regular Small	LMH Medium	0.49
Regular Medium	LMH Small	0.31

Table 3: Results of Experiment 4. Shelley's work is classified using BERT models trained on two types of GPT-2 generated data.

To conclude this section, both evaluations with the BERT classifier—the first being a sort of a "Turing test", and the second being our novel setup—show that the Regular (OpenAI original release) models perform better in general. While we have to watch out for Regular models' tendency to plagiarize text, they could be a preferred choice, especially if we want to generate text with the full sample length of 1024 tokens.

Discussion

This pilot study represents initial explorations into investigating GPT-2 from a computational creativity perspective. The question of whether GPT-2 can generate high-quality poetry (Lamb, Brown, and Clarke 2016), or creative poetry (not necessarily the same goal, as reflected in (Jordanous 2018)), is much larger than the scope of this pilot study; here we focus on the initial steps of model selection for this domain and avoiding problems caused by and analysing consequences of overfitting. One critique of a system based

on generating the style of a known poet (Gervás 2011) is how a poet's style can diverge during their career. This criticism deserves focused attention in our future work; it is not a straightforward question to address and will benefit much from collaboration with experts in English literature. A quick attempt to solve this problem might involve training the model by tagging the dataset with indicators of which period of the author's writing the specific parts come from, and then applying those tags during generation of poems.

A key question here is: is GPT-2 creative? Our work above does not answer that question but gives us some material to consider, which we structure according to the 'Four Ps' of creativity: Producer, Product, Process, Press (Jordanous 2016). GPT-2 can produce Products that might be deemed creative, and it learns from a controlled Press (environment) in the form of the input corpus (though it is not able to interact more widely with its Press). The Process is (very) arguably creative as an example of Boden's exploratory creativity (Boden 2004). Does GPT-2 possess attributes of a creative Person though? This is hard to claim; GPT was developed as a language model, not an AI system, and behaves as a tool to assist a user. That being said: we see such potential to enhance future GPT-x systems through computational creativity research, to make GPT more creative in its own right.

A related question is: is the core task of generating new poems in an existing author's style a valid computational creativity task. Brown and Jordanous consider exactly this question in a new paper (Brown and Jordanous 2022), and give an overall fairly positive answer; in particular, the questions we are addressing in this paper (in particular around avoiding plagiarism and around ensuring high-quality outputs) provide some evidence that the task we are addressing is non-trivial in important ways, and hence more likely to require proper computational creativity effort.

Conclusion

In this study, we analysed the GPT-2 models' outputs with a twofold objective: 1) to observe how overfitting affects the output of GPT-2, and 2) to compare the quality of the output from various versions of GPT-2.

While working with deep neural networks, we are normally looking for the point of the lowest evaluation loss because this is known to lead to the best generalisation (Shalev-Shwartz and Ben-David 2014), though we also know (Domingos 1998) that a bit of overtraining or more complexity can lead to better results on a specific test dataset. The lowest evaluation loss in our results happens very early in the fine-tuning process, that is, before 10K steps in each case. We have trained our models for much longer (up to 250K steps) in order to observe how overfitting affects the quality of the generated samples. In the case of the LMH models, overfitting manifests by production of malformed samples. In the case of the Regular models, the samples are almost always well formed, even for 250K training steps. However, using the BLEU evaluation, we have discovered that, with overfitting, the BLEU score of the samples evaluated against the original dataset is continuously rising, which means that samples contain higher and higher levels of n-grams plagiarized from the original corpus. Effectively, the samples are becoming a kind of collage or pastiche of the original, instead of being fluently created. Such samples could easily mislead both human and automated judges and make them believe that the samples are "good", while they simply contain text plagiarized from the original sources. We should add that with extreme overfitting observed after around 100K training steps, our GPT-2 models plagiarize even long blocks of text (e.g. 50 lines of the original poem can be reproduced by a GPT-2 in many runs; see our supplementary repository for specific examples). Overall, given that we know that machine learning researchers recommend more complex models (Domingos 1998), we advice to stop the fine-tuning process as soon as the samples start looking "good" after the initial learning of the model, and to always check for plagiarism, which can mislead metrics and evaluation that cannot flag plagiarism.

With regards to the second objective of automated evaluation of samples to determine which GPT-2 model would be preferred for poetry generation, we have used two different setups of the BERT-based binary classifier. The first experiment with BERT showed that Regular models are a more reliable choice.

The second BERT experiment, which is our novel approach, in which we classify the samples from the original dataset by the classifier trained on samples from two different GPT-2 models, shows a clear advantage of the Regular Medium model on both datasets. These results are consistent with those of the first setup, and confirm the above findings that the Regular models appear to perform better than LMH models on style preservation tasks.

This study stresses the importance of applying various methods of text evaluation. As of yet, we do not have a single method that would tell us which text is "better". Quantitative methods, like BLEU, can tell us about the repeated n-grams, but they do not inform us about the creative quality of the text. Deep neural network classifiers offer a viable solution, but they can be misled by plagiarised outputs that would be indistinguishable from the original data. Based on our findings, we advice to always use multiple methods of evaluation.

The evaluation setups that we investigated require further research. Will they still provide valid insights when applied to Large and XLarge GPT-2 models, or even the larger models like GPT-3 or EleutherAI's GPT models? Will they be different when applied to much larger or much smaller fine-tuning datasets? If different versions of BERT were used, would they produce different evaluations? This is ongoing research and we hope this study has offered useful insights into the practicalities of evaluating GPT-2-produced text.

The overall contribution of this paper could be seen as both to the AI tools for computational creativity and to the methodologies, which seem to be quite intricate given that the machine learning models lose their generalization capability when overfitted, and can, therefore, plagiarise easily.

Contributions Experimental design: PS with MG, AJ, DB, MP; experimental implementation: PS; writing: PS with MG, AJ, DB, MP, editing: MG, AJ, DB, MP.

References

Alvarez Cos, M.; Perez y Perez, R.; and Aliseda, A. 2007. A generative grammar for pre-hispanic production: The case of El Tajin style. In *Proceedings of the 4th International Joint Workshop on Computational Creativity*, 39–46.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Black, S.; Biderman, S.; Hallahan, E.; Anthony, Q.; Gao, L.; Golding, L.; He, H.; Leahy, C.; McDonell, K.; Phang, J.; et al. 2022. Gpt-neox-20b: An open-source autoregressive language model. *arXiv preprint arXiv:2204.06745*.

Boden, M. A. 2004. *The creative mind: Myths and mechanisms*. Routledge.

Bowman, S. R.; Vilnis, L.; Vinyals, O.; Dai, A. M.; Jozefowicz, R.; and Bengio, S. 2015. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*.

Brown, D., and Jordanous, A. 2022. Is style reproduction a computational creativity task? In *Proceedings of the 13th International Conference on Computational Creativity*.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Corneli, J.; Jordanous, A.; Shepperd, R.; Llano, M. T.; Misztal, J.; Colton, S.; and Guckelsberger, C. 2015. Computational poetry workshop: Making sense of work in progress.

Dale, R. 2021. GPT-3: What's it good for? *Natural Language Engineering* 27(1):113–118.

Das, A., and Gambäck, B. 2014. Poetic machine: Computational creativity for automatic poetry generation in Bengali. In *ICCC*, 230–238.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Domingos, P. 1998. Occam's two razors: the sharp and the blunt. In *KDD*, 37–43.

Elgammal, A.; Liu, B.; Elhoseiny, M.; and Mazzone, M. 2017. CAN: Creative adversarial networks, generating "Art" by learning about styles and deviating from style norms. *arXiv preprint arXiv:1706.07068*.

Ens, J., and Pasquier, P. 2018. Caemsi: A cross-domain analytic evaluation methodology for style imitation. In *ICCC*, 64–71.

Falk, M. 2021. Artificial stupidity. *Interdisciplinary Science Reviews* 46(1-2):36–52.

Floridi, L., and Chiriatti, M. 2020. GPT-3: Its nature, scope, limits, and consequences. *Minds and Machines* 30(4):681–694.

Gervás, P. 2011. Dynamic inspiring sets for sustained novelty in poetry generation. In *ICCC*, 111–116.

Hämäläinen, M. 2018. Harnessing NLG to create Finnish poetry automatically. In *Proceedings of the ninth international conference on computational creativity*. Association for Computational Creativity (ACC).

Jordanous, A. 2016. Four pppperspectives on computational creativity in theory and in practice. *Connection Science* 28(2):194–216.

Jordanous, A. 2018. Creativity vs quality: why the distinction matters when evaluating computational creativity systems. AISB.

Köbis, N., and Mossink, L. D. 2021. Artificial intelligence versus Maya Angelou: Experimental evidence that people cannot differentiate AI-generated from human-written poetry. *Computers in human behavior* 114:106553.

Lamb, C., and Brown, D. G. 2019. Twitsong 3.0: Towards semantic revisions in computational poetry. In *ICCC*, 212–219.

Lamb, C.; Brown, D. G.; and Clarke, C. 2016. Evaluating digital poetry: Insights from the CAT. In *Proceedings of the seventh international conference on computational creativity*.

Lamb, C.; Brown, D. G.; and Clarke, C. L. 2017. A taxonomy of generative poetry techniques. *Journal of Mathematics and the Arts* 11(3):159–179.

Lee, J.-S., and Hsiang, J. 2020a. Patent claim generation by fine-tuning openai GPT-2. *World Patent Information* 62:101983.

Lee, J.-S., and Hsiang, J. 2020b. PatentTransformer-2: Controlling patent text generation by structural metadata. *arXiv preprint arXiv:2001.03708*.

Lee, J.-S. 2019. Personalized patent claim generation and measurement. *arXiv preprint arXiv:1912.03502*.

Li, P.; Zhang, H.; Liu, X.; and Shi, S. 2020. Songnet: Rigid formats controlled text generation. *arXiv preprint arXiv:2004.08022*.

Liao, Y.; Wang, Y.; Liu, Q.; and Jiang, X. 2019. GPT-based generation for classical chinese poetry. *arXiv preprint arXiv:1907.00151*.

Lieber, O.; Sharir, O.; Lenz, B.; and Shoham, Y. 2021. Jurassic-1: Technical details and evaluation. *White Paper. AI21 Labs*.

Loller-Andersen, M., and Gambäck, B. 2018. Deep learning-based poetry generation given visual input. In *ICCC*, 240–247.

Manurung, H. 2004. An evolutionary algorithm approach to poetry generation.

Misztal, J., and Indurkhya, B. 2014. Poetry generation system with an emotional personality. In *ICCC*, 72–81.

Nikolov, N. I.; Malmi, E.; Northcutt, C. G.; and Parisi, L. 2020. Rapformer: Conditional rap lyrics generation with denoising autoencoders. *arXiv preprint arXiv:2004.03965*.

Oliveira, H. G., and Cardoso, A. 2015. Poetry generation with PoeTryMe. In *Computational Creativity Research: Towards Creative Machines*. Springer. 243–266.

Oliveira, H. 2009. Automatic generation of poetry: an overview. *Universidade de Coimbra*.

Oliveira, H. G. 2017. A survey on intelligent poetry generation: Languages, features, techniques, reutilisation and eval-

uation. In *Proceedings of the 10th international conference on natural language generation*, 11–20.

Oliveira, H. G. 2021. Exploring a masked language model for creative text transformation. 62–71.

OpenAI. 2021. openai/gpt-2: Code for the paper "language models are unsupervised multitask learners".

Pachet, F., and Roy, P. 2014. Non-conformant harmonization: the real book in the style of take 6. In *ICCC*, 100–107.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, 311–318.

Project Gutenberg. 2020. `http://gutenberg.org/`.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving language understanding by generative pre-training.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I.; et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1(8):9.

Rae, J. W.; Borgeaud, S.; Cai, T.; Millican, K.; Hoffmann, J.; Song, F.; Aslanides, J.; Henderson, S.; Ring, R.; Young, S.; et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

Rahman, F., and Manurung, R. 2011. Multiobjective optimization for meaningful metrical poetry. In *ICCC*, 4–9.

Rashel, F., and Manurung, R. 2014. Pemuisi: a constraint satisfaction-based generator of topical Indonesian poetry. In *ICCC*, 82–90.

Shalev-Shwartz, S., and Ben-David, S. 2014. *Understanding machine learning: From theory to algorithms*. Cambridge University Press.

Shepperd, N. 2021. nshepperd/gpt-2: Code for the paper "Language Models are Unsupervised Multitask Learners".

Smith, S.; Patwary, M.; Norick, B.; LeGresley, P.; Rajbhandari, S.; Casper, J.; Liu, Z.; Prabhumoye, S.; Zerveas, G.; Korthikanti, V.; et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, 3104–3112.

Toivanen, J.; Järvisalo, M.; Toivonen, H.; et al. 2013. Harnessing constraint programming for poetry composition. In *The Fourth International Conference on Computational Creativity*. The University of Sydney.

Toivanen, J.; Gross, O.; Toivonen, H.; et al. 2014. "The Officer Is Taller Than You, Who Race Yourself!": Using Document Specific Word Associations in Poetry Generation. In *Proceedings of the Fifth International Conference on Computational Creativity*. Jožef Stefan Institute.

Transformers Documentation. 2019. OpenAI GPT2 — transformers 4.5.0.dev0 documentation.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Veale, T. 2013. Less rhyme, more reason: Knowledge-based poetry generation with feeling, insight and wit. In *ICCC*, 152–159.

Ventura, D. 2016. Mere generation: Essential barometer or dated concept. In *Proceedings of the Seventh International Conference on Computational Creativity*, 17–24. Sony CSL, Paris.

Wang, B., and Komatsuzaki, A. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

Wesek, A. 2019. A comprehensive study of state-of-the-art word embedding algorithms for natural language generation. `https://www.cs.kent.ac.uk/people/staff/mg483/documents/wesek19lyrics.pdf`. University of Kent, Unpublished MSc Thesis.

Wöckener, J.; Haider, T.; Miller, T.; Nguyen, T. T. L.; Pham, M. V.; Belouadi, J.; Eger, S.; et al. 2021. End-to-end style-conditioned poetry generation: What does it take to learn from examples alone? In *Proceedings of the 5th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, 57–66.

Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Yu, L.; Zhang, W.; Wang, J.; and Yu, Y. 2017. SeqGAN: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31.