Controllable Language Generation for Movie Script Generation

Pradyumna Tambwekar ptambwekar3@gatech.edu
Haard Shah haard@gatech.edu
Somnath Sarkar ssarkar70@gatech.edu

Abstract

This project deals with genre-based controllable movie script generation. We fine-tune two language models - GPT-2 and BART on the IMSDB movie script dataset using special genre tags to delineate the styles of the script. By learning the embeddings for these genre-tokens we generate novel scripts for unique combinations of genres during inference by using these genre tokens as the input. To evaluate our model we fine-tune a BERT classifier, to identify the most likely genre(s) for a given script. We also explore other unsupervised methods, inspired from evaluating style transfer techniques, to verify whether our controllable model is actually generating scripts within the specified genre. Qualitatively our proposed method proved to be better than our baselines, even though quantitative results didn't completely support this. However the evaluation techniques we propose could be beneficial for future avenues of research

1 Introduction

The goal of this project is controllable movie script generation, where the control we are aiming to incorporate refers to the specific genres of the movies. We would like to generate novel scripts in a genre completely different from a given text prompt. Our approach follows a similar methodology to that proposed in [Keskar et al., 2019, Dathathri et al., 2019], where we fine-tune a GPT-2 model with special genre tokens at the start of each input, i.e. <Comedy>, <Horror>, <Action>etc. Then during inference, we provide the start of a script from our testing set as an input along with a new genre. For example, we could take the start of a "Star Wars" script as an input and specify the genre as horror to generate a novel script based on the input in a genre you wouldn't expect.

Our main contributions include applying CTRL [Keskar et al., 2019] method on a novel dataset and implementing evaluation metrics to quantitatively assess the quality of generations. First, testing CTRL approach on a novel IMSDB movies scripts dataset is a valuable contribution because the style information encapsulated in the various genres of movies is fine-grained and difficult for models learn biases on. Such an approach might also help in data-augmentation for future tasks within this domain. The IMSDB dataset is relatively small, therefore further research in this field could benefit from such a system.

[Keskar et al., 2019] trained their approach on a 140GB dataset including Wikipedia, Project Gutenberg and OpenWebText, which have significant data for individual categories. The IMSDB movie scripts dataset on the other hand has much less data per category (genre). To make sure we can fit our dataset, we use a large GPT-2 [Radford et al., 2019] language model pretrained on 40GB of internet text. We also test the CTRL approach with BART language model [Lewis et al., 2019], which is pretrained on CNN news dataset and also uses a bidirectional encoder to better capture context. Furthermore, subtle nature of the interplay between various movie genres, like Action and Thriller or Fantasy and Sci-Fi or Drama and Romance, calls for a more extensive evaluation metric to assess how controlled the generations are. To do this, we borrow from literature in text style transfer [Syed et al., 2019]. As our second contribution, we translate methods described in [Syed et al., 2019] to quantify stylistic alignment of generated text with target style we wish to generate with.

2 Related Work

Controllable language generation (CLG) has been an ongoing area of research in NLP for the past few years. Prior to the emergence of transformers, controllable language generation models primarily

revolved around reinforcement learning methods like policy gradients. Such methods used a reward signal direct models for multi-turn language generation tasks like dialogue generation [Li et al., 2016, Dhingra et al., 2016] or narrative plot generation [Tambwekar et al., 2019]. However reward functions can be sparse and often require a high-degree of hand-authoring. Research based on Variational Auto-Encoders have proven to be a successful approach to CLG [Hu et al., 2017], but VAE's are used less frequently for this task as they are dependent on effectively learning disentangled representations corresponding to specific aspects of the samples.

In this project, we employ an approach motivated by the CTRL model [Keskar et al., 2019]. They propose a transformer architecture enhanced with control codes to elicit authorial control of the language generated. We apply a similar methodology using other existing transformer architectures for the task of generated genre-focused movie scripts. We build on this model by presenting quantitative evaluation to measure how accurate the language generated represented the given genre. In [Keskar et al., 2019], they validate their results only through qualitative analysis. We use methods inspired by style-transfer evaluation to measure the effect of the genre code on generated scripts [Syed et al., 2019]. Furthermore, we also explored evaluating generated results using another multi-label classifier.

3 Method

In this section we implement three baselines for generating movie scripts. The first two baselines are GPT-2 and BART models which are fine-tuned with control codes on the IMDB dataset. The third baseline is an n-gram language model trained on the IMDB dataset.

3.1 Data

We used movie scripts collected from IMSDB (https://www.imsdb.com) to train our model. Our movie script dataset consists of 1147 documents. Each movie script may have one or more genres, out of a set of 24 unique labels. The scripts have an average document length of 22,521 words. The distribution of movie scripts across genres can be seen in table 1.

Examples of document labels are the scripts for "Titanic (1997)" which is labelled with the "Drama" and "Romance" genres, and "Joker (2019)", which is labelled with "Crime","Drama" and "Thriller".

Scripts uploaded to IMSDB are user-generated, and as a result there is significant noise in the script text and the labels. The first iteration of data cleaning has mainly involved clearing spurious HTML tags from the scripts. This was followed by a pre-processing step to remove infrequently occurring genres. We chose to include only scripts from genres with atleast 50 movies. This left us with 11 genres to use for training, comprising of 1049 movies.

Genre	Number of Movies
Action	307
Action.Thriller	1
Adventure	186
Animation	43
Biography	3
Comedy	378
Crime	216
Drama	626
Family	44
Fantasy	119
Film-Noir	4
History	3
Horror	157
Horror.Mystery	1
Music	5
Musical	25
Mystery	110
Romance	200
Sci-Fi	169
Short	3
Sport	2
Thriller	384
War	28
Western	14

Table 1: Distribution of movies in each genre

3.2 Model/Analysis

In our approach we extended the work presented in [Keskar et al., 2019], by applying their methodology on the GPT-2 model. We modified the embedding layer of GPT-2 to include the new special genre tokens, and pre-pend them to the each input during training (see figure 1).

For the purpose of evaluating the quality of generated scripts, we used a pretrained BERT model and fine-tuned it for the task of multi-label sequence classification, where the labels correspond

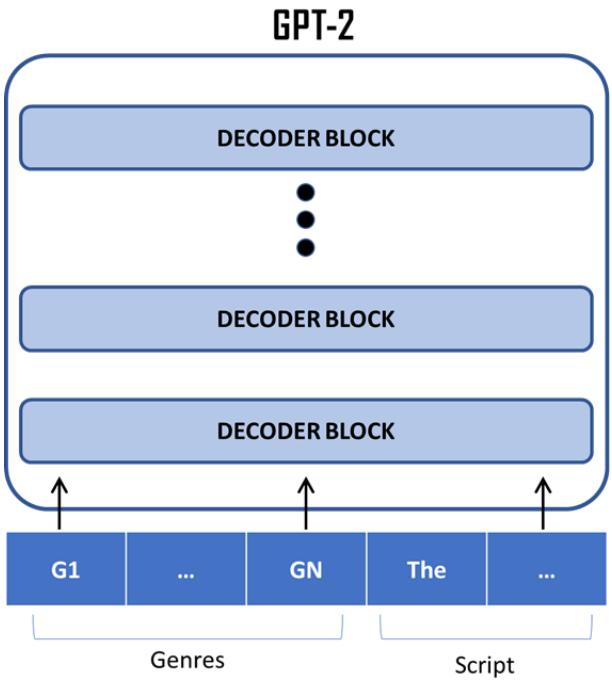

Figure 1: Diagram depicting our method to generate genre-stylized scripts

to the genre of the scripts queried. We chose the *BERT-Base-Cased* pretrained model, which has 110M parameters [Devlin et al., 2018]. As a result of it's bidirectionality, BERT is shown to achieve SOTA performance in text classification task.

We recognize that this model is a fairly simple implementation, but this is not the only model we attempted to implement for our project. We detail all our additional experiments and what we could have done better in the limitations/future work section.

3.3 Baseline Models

3.3.1 BART

BART is a bidirectional encoding model, similiar to BERT. However while BERT is typically unsuited to text generation tasks, BART has been found to perform well on such problems. Of particular interest to us was the fact that the model has been used to achieve state of the art results on text summarization data sets, and we framed the controllable language generation task as a modified version of the summarization problem for this model. We used a Large BART model with an added language modelling head that was pretrained on a large text corpus and fine-tuned on our movie script data set. The fine-tuning procedure consisted of prepending a genre token to a input sequence of maximum length 1024, and attempting to predict the next block of 56 tokens in the data set.

3.3.2 N-Gram

We implement various n-gram language models to serve as another baseline to juxtapose with our approach. We train n-gram models for n = 2, 3, and 4. We use the standard MLE method to training and convert infrequent words in vocabulary to unk. In order to allow n-grams to generate text in a controlled manner across various genres, we simply train distinct models for each genre. Since we want to exclude less frequent genres, with less than 50 movie scripts, we train 11 distinct n-gram models for each n. We train 33 different n-gram models in total. IMSDB dataset contains many movie scripts with more than one movie scripts. We decided not to extend the n-grams to model content of multiple categories to minimize complexity and ensure that each n-gram has enough data to learn from.

4 Results

In this section we explain our experiments relating to qualitatively and quantitatively evaluating movie scripts generated by our approach.

4.1 Experiment Setup

We used the huggingface library to implement our language models [Wolf et al., 2019]. Specifically we used huggingface's pretrained implementation of GPT-2 (Small) [Radford et al., 2019] as our base model. To process our data, we first tokenized scripts using a Byte-Pair encoding (BPE) [Sennrich et al., 2015] tokenizer to create sub-word tokens. We then transformed each script into batches with sequences of length 250 with 20 tokens overlapping in each sequence. Even though overlapping sequences could lead to some redundancy, we chose to include it to give the transformer a better chance of maintaining coherence across the entire length of the script. The average size of each script was approximately 15000 tokens, therefore it would have been impossible to use the entire script as a single input. In future experiments we would like to see how a model with the capability to handle larger input sequences would fare in such a scenario. We discuss this further in section 5.

We fine-tuned our model for 5 epochs using the AdamW optimizer and a linear decay schedule (with warmup) to train our model with a learning rate of 0.003. The entire training process took $\sim$ 80 hours on a Nvidia-Quadro 16 GB GPU.

The classifier was implemented using FastBERT library built using HuggingFace library [Trivedi, 2020]. We used LAMB optimizer proposed by [You et al., 2020] to speed up training. The model was trained on Google Colaboratory for three epochs which took approximately nine hours. The code and hyper-parameter details are available on our GitHub repository linked at the end of this report.

We also implemented a linguistically motivated method introduced by [Syed et al., 2019] to quantify stylistic alignment for the task of style transfer.

They suggest three different methods lexical, syntactic and surface. The syntactic and surface methods try to capture the sentence level and punctuation information such as length of sentence and commas in a sentence. The lexical elements try to capture the word-level information. We attempted to emulate the lexical elements approach to our movie scripts dataset. The authors of [Syed et al., 2019] use a set of seed words for lexical categories such as: subjective, objective, abstract, literary, etc.

In our case we defined lexical categories as the distinct movie genres. First we tried using similar words for our genre categories from Word2Vec, but we ran into out of vocabulary errors when using pretrained Word2Vec models. Then we shifted to manually selecting a set of seed words through a combination of Word2Vec and $relatedwords.org$. We collected 10 words per category with a total of 110 seed words.

We first created a representation of the generated output by averaging the word embedding of all non-stopwords in the outputs. Next we applied the K-nearest neighbors algorithm, using the embeddings of the 110 seed words of each category to calculate a style vector S, where each element s_i corresponds to the number of neighbors out of k that belong to style i. Thus giving us a distinct 11 dimensional style encoding which signifies the probabilities of generated outputs belonging to each style. We set k to 10 for our experiments.

4.2 Result Comparison

4.2.1 Qualitative comparison

Figures 2 and 3 showcase examples of scripts generated using our approach. Each script was generated using the same text prompt with a different genre attached to it. After collectively analyzing scripts generated by our models we noticed a few trends with respect to each baseline. For qualitative analysis we used 20 prompts for 7 styles and generated 140 scripts for each baseline. We then individually examined the results files and discussed our findings to come up with final conclusions regarding each model.

Genre	GPT-2
Comedy	The door opens. SHERIFF Don't you think you're gonna be surprised at this little school? BOY I'm not beach in the morning. I thought I'd be in town. SHERIFF You want to name your son, then you go to your room. BOY Okay, dumbass. SHERIFF You son of a bitch. What did I say? BOY I didn't say that I was a fool. Not around anymore. SHERIFF That's right. You're a fool
Action	The door opens. SHERIFF Welcome to the Army. I'm Jeff Simpson. BUD Jeff Simpson.Jefferson looks like he'd stumble on pop quiz. JEFF I know, I mean...He shoots a look through the eyeholes. BUD nods. BUD What's the score?Jefferson goes for the end of the score. KICKS it up. Tumbling end. JEFF sees it's a nineball game. JEFF What's the score?He throws the ball at the net. It bounces off the wall, pours down the field, and crosses the open concrete.
Horror	The blade flashes, BANG! Then he's gone. CUT TO BLACKDERMEXT. THE STREET IN MIDTOWN PHILADELPHIA – NIGHT The CAMERA looks down on the street. A car has pulled up in front of the apartment building.A GIRL walks down the street, one hand on her hip. DREW is thirteen years old. DREW is a handsome, d Popular character in his show. THE CAMERA moves slowly towards the street. DREW Not so fast! It's practically dead. DREW draws a knife and approaches the girl. DREW (CONT'D) What the hell are you doing in there?
Romance	The shirt is down. Missy frantically tries to adjust criss-cross the buttons. Skeeter's eyes are frozen in confusion. Missy frantically tries to adjust the buttons. She suddenly pulls the button again. Missy frantically presses the button. Skeeter's eyes drift from the control panel to her eyes. The screen goes black.

Figure 2: Examples of scripts generated with the prompt "He hits a button"

With our n-gram models we simply train a generator using separate datasets. Therefore each model is trained on less data than the BART and GPT-2 models, making the generated text less cohesive. With no additional signal for genre, most scripts generated by this method were just abstract, non-specific scripts. It was expected for n-grams generations to be less cohesive than large transformer based models. However, given that each n-gram model only saw data from one genre, we did expected genre specific vocabulary in its generations.

The absence of this suggests the granularity of style information encapsulated in various genres.

Scripts generated through BART did show a tendency attend to the genre specified for specific genres, i.e. Romance and Horror. One such example is demonstrated using the input "She puts her finger to her lips and tells him to hide". If the style is chosen as romance the output is,

> "I'm not going to let you get close to me", she says to him. "You're going to have to hide from me," he replies. "Yes," she replies, "I am going to hide"

Whereas if the style is chosen as horror the output becomes,

> She then looks at him in horror and says, "I'm going to kill you." He looks at her in horror and she says, "I'm going to kill you" She then puts her hand to her mouth and says she wants to kill him.

However there are several limitations of the BART model that impact it's qualitative performance on this task. Sequence generation tasks handled by BART involve copying from the input sequence, after some manipulation[Lewis et al., 2019]. Thus, the model struggles to generate novel text given a prompt. The model's initial weights - pretrained on text summarization data sets - tend to cause text generated by this model to restate the input sequence several times over.

The genre-augmented GPT-2 generates the most coherent scripts out of all three models, which is to be expected given the difference in parameters between models. Even though the genre isn't obvious, there was a perceptible difference between scripts generated with the specified genre token. For instance in Figure 2, the scripts generated with the Action and Horror style tokens are indicative of the genre specified. The action script starts off with a man being welcomed into the army, while the horror script starts with someone slashing a blade, and leads to the characters finding a dead body.

However we acknowledge that there is a slight discordance between our results and actual scripts, which makes the results of GPT-2 hard to qualitatively evaluate. Our results are generated using chunks of a script with a sequence length of 300. Within a script with approximately 15000 words, it is plausible that a significant number of these chunks could just be filler text which is not at all related to the genre of a movie. Therefore the embeddings learnt for genres may be diluted by such examples, which impacts the generated samples.

Further examples of scripts generated from each of our baselines are available in our github repository.

4.2.2 Quantitative Analysis

We attempted to use two methods to quantitatively assess the generated movie scripts. First method was a multi-label classifier implemented by fine-tuning pre-trained BERT classifier. The model had an accuracy of 81.6%, however the F1 score was 0.54. When we used the model to predict genres of generated scripts, the classifier predicted our target genre about 14% of the times, which is no better than chance (see figure: 4).

We trained various configurations of models for 3 to 5 epochs while fine-tuning various hyperparameters such as regularization term, learning rate decay, batch size, lr scheduler, optimizer, etc. Our dataset had an 80:20 training testing split. Regardless of our parameter settings, our model was always overfitting. We suspect two things could have contributed to this. First, we didn't remove stop words when preprocessing the scripts. This could have added unnecessary noise that led the model to overfit to training data. By the time we realized this, it was too late to train another model. Second, as mentioned in the qualitative section, the movie scripts are very long, and we had to process them in chunks. We tried chunks of 300 and 512 and various batch sizes of 8, 16, and 32 on Google Colaboratory. Bigger chunks and batch sizes helped but memory contraints restricted us from using larger input sizes. This may have contributed even more noise.

For computing our clustering results we used 7 styles which we believed to be easily differentiable. Similar to that in our qualitative section, using 20 seed text samples, we generated 140 examples of generated scripts for each baseline. To each of these methods we applied the lexical clustering methodology described in the experimental setup section to obtain style tokens for each generated text sample. The results from this method for our GPT-2 baseline can be seen in figure 5, where each row refers to the average style vectors for all samples with the specified style. The purpose of these experiments was to evaluate the likelihood of the generated output to correspond any particular style as well as compare generated outputs with

Genre	3-gram MLE	BART
Comedy	(righteously) I don't know . But the destruction is even more excited to meet him for a year made me so many things but you get this shit . Like``Oh, yes I want to show Frank: FRANK What? Deacon is talking to this, Marie . DICK Mitch Ryder and the fast heel toe work . 117 THE PIT . As he backs into the base of the eight tentacles snaps off with his 26-year-old chemist named ELLIOT . FOUCHET Please help me out the window . MAGGIE Ha-ha . How much longer do any of this mineral deposit . Jacob is lying on the telephone) Yes! PAMELA Those aren't you vent?	"He hits a button. He hits the button," the character says. "He's like a robot," he adds. "It's like he's trying to kill himself" "He hits the buttons. He hit the buttons," he says, "and he kills himself." "That's what happens when you hit a button."
Action	JACK You...Stanley develops an odd look on in New York .' Bethlehem eyes The Postman who returns a deadpan response . He has a box from Gin's voice trails off as if nothing was beyond suspicion . I'm fine...I just did . These two guys, looking for Joanna Zadek's house . He collapses to his arctic birds . Waiting . He smiles at him expectantly . A They begin to heal . He begins chewing his gum that POPS on cue, Joe and scream as loud as possible and get some rest . ROSE (O.S .) Bruce, if it takes whatever it is that . Not particularly, but if we sit here and Noblesville.	He hits a button. He hits the button. He hits abutton. He hit a button, and he hit the button again, and again, until he was done. He then hit the next button. The button went off, and the camera followed him.
Horror	With some people are forcing the creature . 155 ARTHUR 155 stares at the sound of KIRSTY's shoulder at the sight of the Alps . It's a possibility that she'll let you know that was many years . When you see a WIDE SHOT - MELANIE turning from the earth and where I was going to make him comfortable . You want to get you some of these days --' BOWMAN What's the Law! I am of my face like a fist . Behind her, twelve years ago . ROVERINI Do you know what the hell out! CLARICE You do know she's leaving, but everything's all . CHILDS What's that way .	"He hits a button. He hits abutton. He hit a button," says a character in the film. "It's like a horror movie," says another character. "He hits the button. It's like something out of a horror film"
Romance	CAMERA TILTS DOWN ONTO THEIR FEET go pounding through the intricate``wiring diagrams ."Does that upset you . BARON If we are sending him reeling into the distance . Porthos urges him on the screen . SAMANTHA Hey, hey, hey, what does that make it work, your car in her laughter increasingly out of his EXERCISE ROUTINE . PAN TO a booth . MEREDITH Mr. Pappas, for once without having the alfalfa sprouts and, feigning hurt . I just got it out . Warden, continue . BELLA I needed something...Anything? GOWER No...(he loses his epee when fighting the war, I mean, two other men around shout similar advice .	He hits a button. He hits abutton. He hits another button. He hit a button again. Hehit a button once. Once again, he hit another button, another time, another way. HeHit a button twice. He Hit a Button Twice. He Hits a button Twice.

Figure 3: Example scripts generated by our baselines with the same seed text as that of GPT-2, “He hits a button”

	top-1	top-3
GPT-2	0.135714	0.378571
BART	0.028571	0.228571
2-Gram	0.235714	0.75
3-Gram	0.314286	0.728571

Figure 4: Accuracies for each baseline using the BERT classifier

	Action	Comedy	Thriller	Horror	Romance	Sci-Fi	Fantasy
Action	.186	.309	.237	.239	.173	.154	.131
Comedy	.154	.336	.250	.247	.185	.139	.117
Thriller	.176	.306	.243	.242	.178	.154	.130
Horror	.168	.298	.254	.248	.173	.156	.131
Romance	.162	.321	.253	.249	.171	.146	.126
Sci-Fi	.174	.333	.248	.244	.156	.149	.125
Fantasy	.181	.296	.241	.237	.180	.147	.146

Figure 5: Lexical clustering results using styles as clusters

regards to their perceived style. However our results similar to that of the BERT experiments. We were unable to get conclusive results showing a positive correlation between the values within the style vector and the actual style. The probability values were relatively uniform across the 7 styles for the generated samples. We’ve conducted these experiments for all our baselines and we also experimented with using word2vec embeddings for kNN instead, and the results of all these experiments can be found in our github repository.

We acknowledge that for this method to be a effective evaluation methodology, it requires a much more extensive set of seed words. The dilemma here is we couldn’t tell whether the GPT-2 was generating uniform scripts or whether the kNN evaluator has efficiently learnt clusters for each style in the dataset. Our goal with these experiments was to showcase a possible evaluation that could be useful in gauging how well a CLG model is actually providing lexical control to generated samples. To make this method viable in future experiments we would need a much more extensive list of words along with an empirical methodology for generating this set.

4.3 Work Division

1. Pradyumna: Trained style-specific GPT-2 model and created dataloader for the IMSDB dataset. Assisted Haard in setting up evaluation experiments. Conducted additional experiments with discriminator augmented GPT model as well as longformer model. Con-

ducted literature review for CLG techniques. Oversaw final report writing.

2. Somnath: Trained BART for text generation fine-tuned on the movie script data sets

3. Haard: Fine-tuned BERT multi-label classifier. Implemented n-gram models, trained with MLE, and used it to generate text. Calcualted tf-idf vectors and visualized vocabulary across genre using tSNE plots. Implemented evaluation metric for lexical style analysis.

5 Additional Experiments/Future Work

We want to use this section to explain the other approaches we tried and what we would do in future research in this direction. After getting our original model to work, we tried to augment the performance of the stylized scripts generated by the GPT-2 model using a discriminator. We wanted to use our trained BERT classifier to obtain the likelihood of a generated script being from the genres specified, and use that information as an added regularization term to the default GPT-2 cross entropy loss. The additional loss term can be mathematically represented as follows,

$$L_d = \frac{1}{N}\sum_{i}^{N}\sum_{j}^{G_i}(1 - P_{ij}) \tag{1}$$

where N is the number of samples in a batch, G_i are all the genres associated with sample i, and P_{ij} is the probability outputted by the pretrained BERT classification model of generated sample i being classified as a script of genre j. We wanted to further expand this approach by treating this system as a GAN rather than just a regularization term, and further training the discriminator along with the generative GPT-2 model. Unfortunately due to compute constraints we were unable to run these experiments. While completing this project we only had access to google colab credits and limited usage of a single GPU machine with a 16 GB GPU. When we ran this model using either of these two setups the process kept getting killed after $\sim$100 iterations. Our implementation of this can be found in the github repository linked in this project, under the filename `train_scripts_discriminator.py`.

We also attempted to incorporate the recently released longformer [Beltagy et al., 2020] model. The longformer was a transformer whose attention mechanism scales linearly with sequence length rather than quadratically, therefore making it more suitable for large documents sizes. However the code for this work was just released 2 weeks ago and we did not have enough time to debug and train our approach using the longformer as the base model. The use of this model would be a very interesting avenue of future work, since this model can process and generate entire scripts.

6 Conclusion

In this paper we presented a method of controllable script-generation using a methodology which trains genre embeddings for GPT-2 which can then be used to control language generated during inference. We showcase the first method to generate unstructured movie scripts using the IMBDd dataset while also providing the ability to control the genre of the script generated. We show examples generated by our methodology to prove that our proposed method is qualitatively better than our baselines.

Furthermore, we tested two methods to quantitatively evaluate the generations. One was a transformer based multi-label classifier and the other used the concept of innate style that can be captured using lexical clues. The evaluation methods weren't successful, but they both are methods have which have shown success in previous literature and are worth researching further to be used to assess the quality of controlled text generation.

7 Code Repository

Our code for training GPT-2 on the IMDBd dataset can be found here, https://github.gatech.edu/ptambwekar3/Script-Generation

References

Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: a simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. Towards end-to-end reinforcement learning of dialogue agents for information access. *arXiv preprint arXiv:1609.00777*, 2016.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1587–1596. JMLR. org, 2017.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*, 2019.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

Jiwei Li, Will Monroe, Alan Ritter, Michel Galley, Jianfeng Gao, and Dan Jurafsky. Deep reinforcement learning for dialogue generation. *arXiv preprint arXiv:1606.01541*, 2016.

Alec Radford, Jeffrey Wu, Dario Amodei, Daniela Amodei, Jack Clark, Miles Brundage, and Ilya Sutskever. Better language models and their implications. *OpenAI Blog https://openai. com/blog/better-language-models*, 2019.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

Bakhtiyar Syed, Gaurav Verma, Balaji Vasan Srinivasan, Vasudeva Varma, et al. Adapting language models for non-parallel author-stylized rewriting. *arXiv preprint arXiv:1909.09962*, 2019.

Pradyumna Tambwekar, Murtaza Dhuliawala, Lara J Martin, Animesh Mehta, Brent Harrison, and Mark O Riedl. Controllable neural story plot generation via reward shaping. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 5982–5988. AAAI Press, 2019.

Kaushal Trivedi. Fast-bert. https://github.com/kaushaltrivedi/fast-bert, 2020.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019.

Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, Cho-Jui Hsieh, and et al. Large batch optimization for deep learning: Training bert in 76 minutes, Jan 2020. URL https://arxiv.org/abs/1904.00962.