Large Language Model for Rap Generation

Bartosz Dzionek **Jamie Welsh** **Emmy Zhou**

University of Edinburgh

April 4, 2023

Abstract

Composing rap lyrics requires a blend of rhythmical skills and lyrical prowess, with intricate rhyme patterns serving as a fundamental component of good lyricism. In this paper, we present DistilGPT2-rap, a rap lyrics generation model that is a Large Language Model (LLM), trained to capture the stylistic elements of rap. To develop our model, we adopt a transfer learning approach by fine-tuning the pre-trained DistilGPT2 with a custom dataset of rap lyrics scraped from the internet. We evaluate our model against several baselines, including the more complex GPT2-Large, using perplexity and task-specific metrics. We also propose new metrics for rap lyrics evaluation. Our results reveal that DistilGPT2-rap achieves state-of-the-art perplexity and rhythmic performance. Furthermore, we also present a qualitative evaluation of the generated lyrics. Our work has potential applications in the music industry, such as assisting artists with the creation of new songs.

1. Introduction

Rap is a widely popular music genre that features rhyming lyrics spoken rhythmically over a musical beat. Short for rhythm and poetry, the style is characterised by its complex structures of rhyme, rhythm, and semantics, making it a fascinating domain within the intersection of computational creativity and natural language processing. Typically, rap lyrics consist of rhyming couplets, with each line containing a specific rhythm and a set number of syllables (Janzer, 2012). The lyrics often incorporate wordplay, metaphors, and homonyms, requiring a profound understanding of language to replicate. Since rap is deeply rooted in African American culture, its lyrics also reflect the social, political, and economic realities of the community (McCoy, 2017). Consequently, one of the significant challenges in rap lyrics generation is to produce lyrics that are both artistically and culturally relevant.

While recent literature shows an abundance of high-quality poem generation models, rap presents a distinct challenge as it requires the generated text to not only rhyme but also have a rhythmic flow. Moreover, the types of rhyme schemes used in rap differ from those found in traditional poetry, where the use of multisyllabic rhymes is much more extensive. Colloquially known as “multis”, multisyllabic rhymes are rhymes which contain two or more syllables (e.g., “random luck”, “vans and trucks”). Such rhymes do not typically consist of words which fully rhyme but rather assonances, i.e., the rhyming of two or more stressed vowels such as “baby” and “crazy”. Multisyllabic rhymes are often used by rappers to demonstrate their technical skill and lyrical prowess and are considered a hallmark of complex and advanced rapping (Edwards, 2009). For example, the rapper Eminem cleverly combines multisyllabic rhymes with storytelling in his song "Lose Yourself":

"His palms are sweaty, knees weak, arms are heavy
There’s vomit on his sweater already, mom’s spaghetti"

In this verse, Eminem creates a series of multisyllabic rhymes with the repeated -a sounds in “palms”, “arms”, and “mom’s” as well as the -e sounds in “sweaty”, “knees”, “weak”, “heavy”, and “spaghetti”. Our goal with this project is to develop a model capable of recreating such intricate rhyme schemes.

Previous approaches to rap lyric generation have ranged from predicting the next line from a bank of existing lines (Malmi et al., 2015) to training an LSTM (Potash et al., 2015) to create lyrics in the style of a given rapper. In this work, we tackle the challenge of rap lyrics generation by adapting a transfer learning approach. Our objective is to fine-tune a pre-trained language model on a corpus of rap songs. Through experiments, we want to verify how good the lyrics generated by the model are. Our work is driven by the increasing demand for systems that can interact with humans in a creative and non-mechanical way. Moreover, we are interested in studying the formal structure of rap and developing a model that can produce artistic work that is both creative yet compliant with the style of the genre.

Our work makes the following contributions:

- We create a custom dataset of rap lyrics by scraping song lyrics from songs written by a collection of well-known rappers.
- We fine-tune a large language model on the dataset and outperform much larger models in perplexity.
- As far as we know, this is the first large language model fine-tuned for English rap lyrics generation.

- We propose two new metrics for rap generation evaluation. In particular, our model performs the best in syllable count difference.

- To facilitate future research on rap generation, we make both our code[1] and model[2] publicly available.

2. Related work

The first computational approaches to rap lyric generation framed the task as machine translation (Wu et al., 2013; Wu & Addanki, 2015) where the proposed models learn associations between words such as rhymes and phrases. In Wu et al. (2013), a *freestyle* recursive neural network (RNN) model is trained which aims to generate an appropriate response to a hip hop lyric, akin to a rap battle where participants must continuously improvise responses on the spot. This model is compared to a baseline statistical machine translation model and found to outperform this baseline in terms of both fluency and rhyming.

Malmi et al. (2015) build on this work in *DopeLearning*, where a computational approach based on information retrieval is proposed. Rhyme and structural features are extracted and combined with a deep neural network which learns vectorised representations of the lyrics to model the semantics. This model is trained to suggest the next lyric of rap based on only the previous line. However, the model does not generate new lyrics, it simply selects its preferred lyric from a bank of existing lines in rap songs. The trained model predicts the next lyric correctly with an accuracy of 17% when faced with a choice of 300 possible next lines (including the correct one). The notion of rhyme density is presented in this study and can be thought of as the average number of rhyming pairs per line in a song/rap. Their model *DeepBeat* outperforms all top human rappers in this rhyme density metric highlighting the ability of the model to create effective rhyming lyrics, even if they are semantically meaningless.

Instead of choosing the next lyric from a bank of existing lyrics, Potash et al. (2015) train a Long Short Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997) language model to create new lyrics in the style of a given rapper and hence the model is aptly named *Ghostwriter*. This LSTM model generates lyrics with a far lower rhyme density than in Malmi et al. (2015) however the LSTM architecture proves to fare better than a baseline n-gram model in terms of rhyme density and similarity to the target artist.

In Liang et al. (2018), a new way to learn useful representations of rap lyrics called AttAE-RL2 is proposed. This method uses an attention-based autoencoder that focuses on capturing the specific features of rap lyrics, such as their rhyme, rhythm, and meaning, to create better representations. Large language models often make use of similar attention structures to capture dependencies between words. The authors evaluated AttAE-RL2 on different datasets and showed that it outperforms existing methods such as Malmi et al. (2015) in terms of next-line prediction. It also outperforms various benchmarks in representation quality and downstream task performance, such as identifying the artist or genre of a song.

Previous approaches such as Malmi et al. (2015) and Potash et al. (2015) focused on the unconditional generation of rap lyrics, whereby lyrics are generated without any focus on the surrounding context. Generating lyrics based on context may help lyrics to exhibit more of a narrative story. Training sequence-to-sequence models would allow for the creation of rap lyrics while implicitly learning the context surrounding rap verses, with the hope that the output lyrics tell more of a coherent story (Vaswani et al., 2017). A novel approach to generating a rap verse based on another piece of text is presented in Nikolov et al. (2020), this is called *conditional generation*. A transformer-based autoencoder is trained to take existing text and reconstruct it in a particular style. Important content words are extracted from the text and lyrics are constructed from these content words. A further step uses the large language model BERT (Devlin et al., 2018) to paraphrase lyrics in order to increase the rhyme density. It results in an increase of more than 10%.

While studies like Malmi et al. (2015), Potash et al. (2015) and Nikolov et al. (2020) focus on the rhyming side of rap lyrics with little regard for the rhythm of the lyrics, Xue et al. (2021) aims to also account for the rhythmic aspect. Transformer-based neural architectures are the modern state-of-the-art for language modelling (Devlin et al., 2018; Radford & Narasimhan, 2018). With that in mind, a transformer-based system that models both rhymes and rhythms is trained to generate rap lyrics in Chinese. A beat symbol is inserted into the lyrics in order to model rhythms. The model is evaluated with the metrics of rhyme accuracy and rhyme density. The rhyme density achieved is 1.65, outperforming earlier lyric generation methods (Malmi et al., 2015; Potash et al., 2015). The models trained in these studies were able to often achieve rhyming on the last token of each line but they fared worse at incorporating rhymes within lines. Xue et al. (2021) was the first study to fine-tune an LLM for the task of rap lyric generation. However, their model only generated songs in Chinese, while our model works in English.

3. Data set

As far as the authors know, there does not exist any publicly available dataset with rap lyrics. This is due to copyright infringements that could arise when publishing such datasets. Instead, the most common approach in the literature is building custom datasets by scraping lyrics websites (Wu et al., 2013; Malmi et al., 2015; Xue et al., 2021).

To build the dataset, the first step was obtaining a list of English-rapping artists. Combining a ranking of the most popular rappers on a popular TV channel with two different community-based rankings, yielded a set of 226 rappers.

[1] https://github.com/dzionek/llm-rap-generation
[2] https://huggingface.co/dzionek/distilgpt2-rap

Dataset	Artists	Songs	Lines	Words
Potash et al. (2015)	1	-	-	40K
Malmi et al. (2015)	104	11K	580K	-
ours	226	21K	1.6M	12M
Wu et al. (2013)	-	52K	2.7M	22M
Manjavacas et al. (2019)	28K	65K	-	37M

Table 1. Approximate dataset counts compared against literature.

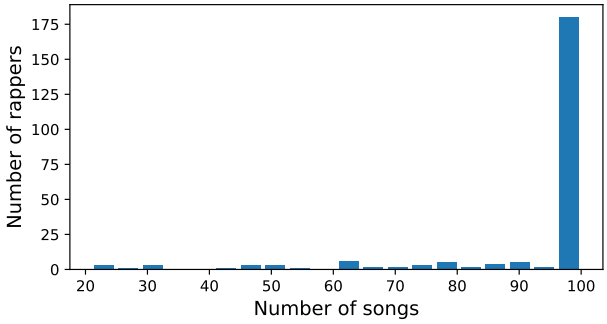

Figure 1. The histogram of songs per rapper in our dataset.

Then, we scraped a popular lyrics website for lyrics of the most popular songs of each rapper. We capped the number of songs per rapper at 100 to better balance the dataset and potentially avoid lyrics of lower quality.

We manually validated the dataset to guarantee the robustness of the scraping pipeline. Then, we pre-processed the data for training in order to remove some of the extraneous text. This was done by splitting each song into its lines and removing any special characters in the text. Furthermore, the decision was taken to remove any bracketed expressions from each line since these were often ad-libs that interfered with the rhymes at the end of the lyrics. Finally, for simplification, all text was made lowercase. Additionally, lyrics were encoded by a tokenizer associated with each large language model. Since the models can only use a limited number of tokens, each song was truncated to first the 1024 tokens.

Table 1 presents statistics about our dataset with regard to others in the literature. Our dataset has three times lower number of words than the largest dataset by Manjavacas et al. (2019), but we achieve this with 100 times fewer artists. We hypothesize that having a large number of words coming from a small number of artists can make our model more coherent. This is due to potentially big differences between rap styles and quality when having a large set of artists. Moreover, our dataset might be more balanced than in other papers as we managed to get 100 songs for the vast majority of rappers. This is clearly visible on the histogram of songs as presented in Figure 1. The authors are not aware of such distributions from the other papers.

We will split our dataset into a training set comprising of 80% of the songs chosen randomly. For evaluation, we will use the remaining 20% as the validation set.

4. Task

The research question that we would like to answer in this paper is: *Can one fine-tune a large language model with a task-specific dataset in order to generate high-quality rap lyrics*?

Given a sequence of tokens $\langle X_1, \ldots, X_T \rangle$, language modeling is a task of estimating the distribution $\mathrm{P}(X_1, \ldots, X_T)$ (Eisenstein, 2019). By the chain rule of probability, the joint probability decomposes into a product of conditionals:

$$\mathrm{P}(X_1, \ldots, X_T) = \prod_{t=1}^{T} \mathrm{P}(X_t \mid X_1, \ldots, X_{t-1}) = \prod_{t=1}^{T} \mathrm{P}(X_t \mid X_{<t}).$$

A large language model is a machine learning model that has been trained on immense amounts of textual data. Large language models are usually based on deep neural network architectures, and learn patterns and relationships in the data during the training process. In a similar fashion to the approach of Xue et al. (2021), a large language model is fine-tuned on a dataset of rap lyrics. The language model Distil-GPT2 (Sanh et al., 2019) is fine-tuned for a small number of epochs on the dataset created in section 3. This will expose the model to the stylistic and semantic properties of the genre. A simplified visual representation of the fine-tuning process is observed in Figure 2.

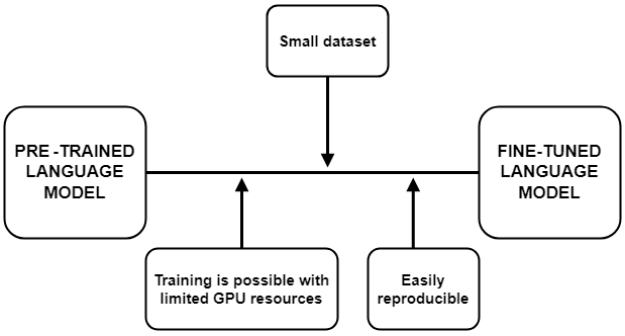

Figure 2. A simplification of the process of fine-tuning a pre-trained language model.

4.1. Evaluation metrics

As is common practice in the literature, we evaluate our language models both intrinsically and extrinsically on the held-out validation set.

In terms of a task-neutral metric for language models, we focus on **perplexity (Ppl)**. Given the log-likelihood $\ell = \sum_{t=1}^{T} \log \mathrm{P}(X_t \mid X_{<t})$, perplexity is defined as $2^{-\ell/T}$ (Eisenstein, 2019). Lower perplexity indicates the text is more likely to be generated by the language model.

In terms of task-specific metrics, four metrics are used:

Rhyme Density (RD) Following the approach of Malmi et al. (2015), we employ rhyme density as an evaluation metric to measure our model's ability in producing multisyllabic assonance rhymes. This measure calculates the average length of matching vowel sequences within a fixed

window of words such that a higher rhyme density corresponds to rhyming pairs occurring more often in a set of lyrics. To compute the rhyme density, we perform the following steps:

1. Transcribe the lyrics to phonemes using an English to International Phonetic Alphabet (IPA) converter (The CMU Pronouncing Dictionary, 2023) and remove all non-vowel phonemes.
2. Iterate over the lyrics word by word, searching for the longest vowel sequence within a fixed window of neighbouring words. By default, we set the window size to 15 but this can be adjusted.
3. Ignore matching sequences that originate from repeated phrases such as *"Walk it like I talk it, Walk it like I talk it"* (from Migos "Walk It Like I Talk It").
4. Compute the average length of the longest vowel sequence for all words.

Our computation approach varies slightly from (Malmi et al., 2015) in that we do not map similar-sounding vowels to each other. While this yields lower rhyme density scores, it also rewards better-quality rhymes that are more impactful.

Syllable Count Difference (SCD)

Rap frequently adheres to a particular rhythmic structure called a "flow," which typically involves fitting a specific number of syllables into each bar i.e., a line of rap. Though the syllable count is not always fixed, many rappers maintain a consistent syllable count throughout their verses, especially in more structured forms of rap such as freestyle battles or acapella (Edwards, 2009).

Since our dataset only consists of rap lyrics and no information about the music, we have resorted to using syllable counts as a proxy measure to evaluate the rhythmical quality of the generated lyrics. As such, we introduce a new metric, syllable count difference, which measures the average difference in the number of syllables for pairs of consecutive bars across the whole song. Considering an example from Eminem's "Rap God", the consecutive bars *"I'm beginnin' to feel like a Rap God, Rap God"* and *"All my people from the front to the back nod, back nod"*, would get a desirable score of 0 whereas the randomly chosen pair *"But for me to rap like a computer, it must be in my genes"* and *"Made a livin' and a killin' off it"* would receive a poor score of 7.

Longest Rhyme (LR)

Another new metric we introduce is longest rhyme. The metric corresponds to the length of the longest multisyllabic assonance rhyme found in a set of lyrics. It can be computed together with rhyme density.

Vocabulary Size (VS)

As proposed by Daniels (2019) in their ranking of famous rappers, the number of unique words used by a rapper can be taken as a quantitative measure of lyricism. In evaluating the lyrical diversity of our model, we are interested in the percentage of unique words within a song. The vocabulary size metric is computed by dividing the number of unique tokens by the total number of tokens. As far as the authors know, this is the first time vocabulary size has been used to evaluate machine-generated rap lyrics.

Split	RD ↑	SCD ↓	LR ↑	VS ↑
Train	0.72 ± 0.14	1.76 ± 3.4	5.15 ± 1.2	0.44 ± 0.1
Valid	0.72 ± 0.14	1.74 ± 2.28	5.11 ± 1.2	0.44 ± 0.1

Table 2. Rap metrics on the dataset splits (mean and standard deviation).

For reference, we present the task-specific metrics on our training and validation sets in Table 2. As expected, there is a very small discrepancy between the splits.

5. Methodology

The rap lyrics in the dataset likely differ in style compared to most of the text pre-trained language models have been trained on. However, a vast amount of data is required to train a language model from scratch and there may not be enough rap lyrics in existence to construct a dataset of sufficient size to train a large language model. Furthermore, pre-training large language models like BERT (Devlin et al., 2018) can take up to 100 days even with multiple GPUs. Instead of training a language model from scratch, an existing pre-trained language model can be fine-tuned to the task of rap lyric generation.

The dataset of existing rap lyrics obtained via the web scraping method described in section 3 will be used to fine-tune DistilGPT2 (Sanh et al., 2019), a smaller version of the pre-trained language model GPT-2 (Radford & Narasimhan, 2018). Released in 2019, GPT-2, (Generative Pre-trained Transformer 2), is a large language model developed by OpenAI with 1.5 billion parameters. Observe in Figure 3, a diagram representing the architecture of GPT-2. The model consists of a stack of transformer blocks, where each block has a self-attention mechanism that allows it to capture the dependencies between all tokens in an input sequence. The GPT-2 model is trained on a large dataset of text by predicting the next word in a sentence given the sequence of preceding words. This allows the model to develop a deep understanding of the semantics as well as the structure of the sentence.

DistilGPT2 is fine-tuned instead of the full GPT-2 model as it is a faster and lighter version which better fits our computing resources. It was designed by HuggingFace following the same methodology used to create Distil-BERT (Sanh et al., 2019). It was designed to reduce the computational resources required to run GPT-2, while still maintaining high performance. This is achieved by using a technique called *knowledge distillation*, whereby knowledge is transferred from a large model (GPT-2) into a smaller one (DistilGPT2).

Model	Ppl ↓	RD ↑	SCD ↓	LR ↑	VS ↑
N-gram (ours)	943	0.55 ± 0.16	4.52 ± 2.34	**3.31 ± 0.67**	**0.8 ± 0.06**
DistilGPT2 (Sanh et al., 2019)	132	0.55 ± 0.37	5.26 ± 8.77	3.01 ± 1.44	0.59 ± 0.22
GPT2-Large (Radford et al., 2019)	72	**0.56 ± 0.32**	3.19 ± 7.19	3.04 ± 1.18	0.58 ± 0.19
DistilGPT2-rap (ours)	**41**	0.54 ± 0.25	**2.2 ± 4.57**	3.3 ± 1.13	0.51 ± 0.15

Table 3. Evaluation metrics reported on the validation set (mean and standard deviation per song).

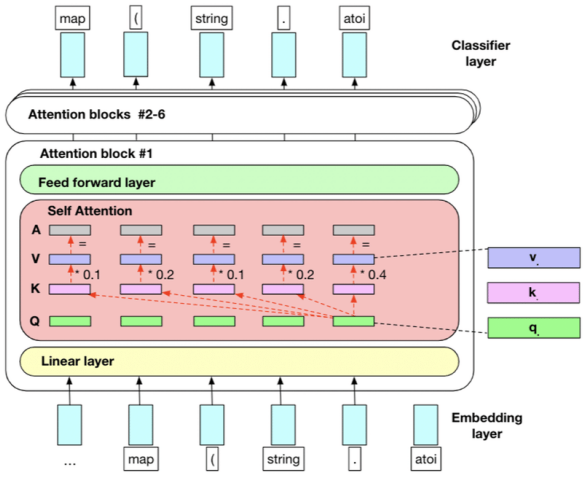

Figure 3. The attention-based architecture of GPT-2 (Aye et al., 2020).

In terms of performance, DistilGPT2 inherently achieves slightly lower scores on language tasks than the full GPT-2 model. However, the difference in performance might not big enough to have a substantial effect on the quality of lyrics generated.

Our dataset does not have to be as large as the one required to train large language models in the first place. This is because the weights of the model have already been initialised to reasonable values. The goal is to generate text in the form of rap lyrics, given some input text. Specifically, we want the model to generate a continuation of a rap song given just the first line. The language model will be fine-tuned by passing the training set (section 3). Fine-tuning assumes the pre-trained weights are already reasonable, so it will only slightly change all parameters of the model. This allows us to train for only a few epochs, to make the model exposed to the dataset comprised of solely rap lyrics.

To test perplexity, the fine-tuned model will be evaluated on the validation set (section 3). For task-specific metrics, we will provide the first line of songs from the validation set and ask the model to generate the next 100 tokens. Such generated lyrics will be compared to the original. We will also compare our models to the baselines. Our results will be fully-reproducible as we will make both our code and model publicly available.

One significant challenge with the approach outlined is that rap lyrics by their nature contain a lot of slang and colloquialisms. It is reasonable to expect the pre-trained models to have some difficulty understanding certain words and phrases. Fortunately, models like GPT-2 use sub-word tokenization techniques in order to mitigate the issue of rare words (Malkin et al., 2021). Although, rarer words tend to have less context than common words. So, the model might still perform worse on the rarely occurring words.

6. Experiments

Through all our experiments we want to assess how good the models are at generating high-quality rap lyrics.

Our first baseline is a non-neural n-gram model. In the n-gram statistical model, we assume the token X_t depends only on its previous $(n-1)$ tokens, i.e., $\mathrm{P}(X_1, \ldots, X_T) \approx \prod_{t=1}^{T} \mathrm{P}(X_t \mid X_{t-n+1}, \ldots, X_{t-1})$. One of the advantages of n-gram is simple training with maximum likelihood estimation (Jurafsky & Martin, 2008):

$$\mathrm{P}(X_t \mid X_{t-n+1}, \ldots, X_{t-1}) \approx \frac{\mathrm{count}(X_{t-n+1}, \ldots, X_{t-1}, X_t)}{\mathrm{count}(X_{t-n+1}, \ldots, X_{t-1})}.$$

By comparing different values of n, we choose 3-grams based on their perplexity. To make sure the probability is defined for tokens outside the training set, Laplace smoothing is included with $\alpha = 10^{-4}$ (Manning et al., 2008). This is especially important as rap lyrics often contain slang and made-up words.

The next baseline is GPT2-Large (Radford et al., 2019). With 774 million parameters it is prohibitively hard to train using our computational resources. However, we are still able to evaluate it. To compute perplexity, we use all songs from the validation set. To generate text and compute rap-specific metrics, we use 1000 first songs to limit our computational time. Since this is a quarter of the validation set, we do not see it as a major limitation of our study. Similarly, DistilGPT2 without fine-tuning is another baseline to make sure the fine-tuning improves rap generation.

Note that not fine-tuned versions of GPT2-Large and DistilGPT2 are general language models. The generated model might generate text that it does not consider to be a rap song. To mitigate this, when generating lyrics we start the input with the prompt: *(Below are ONLY lyrics from a rap song)* for both non-fine-tuned models.

Finally, we fine-tune our model of DistilGPT2 using our training set. It is trained for 7 epochs using AdamW optimizer (Loshchilov & Hutter, 2017) with the learning rate 10^{-4} and weight decay 10^{-2}. This model does not start with a prompt as it can learn the structure of rap lyrics.

Cut the music up
A little louder, yeah

You had a lot of crooks try to steal your heart
Never really had luck, couldn't never figure out
How to love, how to love
You had a lot of moments that didn't last forever
Now you in a corner, tryna put it together
How to love, how to love

Figure 4.1. Original song (*How to Love*, Lil Wayne)

cut the music up
what what what
got fly hoes kneel
stand your loneliness
im holding you down to a 50000
check it yo
jump from gee to gee
so im like all about bacon or sausage
the ultimate freak off

Figure 4.2. n-gram (ours) continuation

cut the music up
Just wait til the world stops
But wait till the world stops
I'm gonna tell you what's in my life,
How I made more than a hundred thousand dollars a week
I've been here before so many times
And my life is a thousand miles long
No no no, it's not like that,
I can't wait until I get out the door
Well, I'll be back

Figure 4.3. GPT2-large continuation

cut the music up
The track sounds like a dance song
How it sounds in the studio
The song sounds like an old-fashioned dance-club [...]
You start a song
Now, when your song starts,
you get closer and closer

You've got to get on it
The song sounds like two dance songs.

Figure 4.4. DistilGPT2 continuation

cut the music up
the beats is loud im sure to send them higher
dont go to sleep youre in the mind of the lord
youre not the one to see the god within within
and all this noise around my head youre the chosen
youre not the one to see me fallin victim of your power
no matter what you do you know my soul is strong
so i can do what i cannot do

Figure 4.5. DistilGPT2-rap (ours) continuation

Godfather's in the house..
Grab yo' bitches

Pimpin ain't pimpin ain't easy man
Pimpin ain't pimpin ain't easy man
Pimpin ain't pimpin ain't easy man
Pimpin ain't pimpin ain't easy man

Figure 5.1. Original song (*Pimpin' Ain't Easy*, Ice-T)

godfathers in the house
hey yo cause im on that
i know my language
but i stay on the top down riding clean
its goin down
about the fresh trim at
bitch im a savage
aint nothing wrong with the vibe aint no [...]
losing my balance
just rockin it
having you be hangin in bars naked for dollars

Figure 5.2. n-gram (ours) continuation

godfathers in the house
and the house in me (above), I say it's too cold
I don't let nothing stay to me
the house in me, I can hear them calling in my chest
if I get lost on thursday (the day of the [...]
the house in me, I'm in the mood for love
the house in me, so cold so

Figure 5.3. GPT2-large continuation

godfathers in the house
The two have a heart
the two have never talked

a true friendship and love
the two have never been lonely
And all that he cared for
A love has never been lost
A romantic home in my back yard
My mother had a heart
The two have never had that love
A home in my back yard
This year the two have two great children. [...]

Figure 5.4. DistilGPT2 continuation

godfathers in the house
my motherfuckin name is krsone
ill get to smacking her for that dick
and ima bust it all
ill leave your motherfuckin head cracked
imma stick it like a knife
and i put em up in a tree
i like the way you kick it and that nasty sound
uh uh baby baby
you aint fucking wit this dick
im a damn good motherfuckin slut

Figure 5.5. DistilGPT2-rap (ours) continuation

Importantly, to make sure large language models do not repeat generated lines, we generate their output using top-k sampling with $k = 50$ (Holtzman et al., 2020).

6.1. Quantitative results

The numerical results are presented in Table 3.

Our model achieves the lowest perplexity of all models tested. We believe perplexity is the most important metric as it is vital that our model can interpret unseen data as being similar to what the model generates itself.

DistilGPT2-rap also has the best syllable difference count. This implies our model learned that consecutive lines should be of similar length measured in syllables. While this is an oversimplified method of measuring rhythm, it shows that our model is good at handling the basic rhythmic aspect of writing lyrics in absence of any musical information. Considering the syllable difference count metric, our model is in fact very similar to values reported on the dataset (Table 2).

It is difficult to draw definitive conclusions from the rhyme density metric, as all models yield nearly identical scores. While n-gram and DistilGPT2-rap surpass the non-fine-tuned baseline models in the longest rhyme category, the rhyme density and longest rhyme scores are still significantly lower than those in the original dataset listed in Table 2.

Interestingly, all models have a higher average vocabulary size than our dataset. Especially, the n-gram model produced the most diverse output in terms of vocabulary. We suspect the large vocabulary is due to the Laplace smoothing used in the n-gram model, where low-frequency words receive extra probability mass to counteract data sparsity.

6.2. Qualitative results

To evaluate models quantitatively, we give the first line of lyrics to the model (as discussed in section 5) and generate continuation. Two samples are presented in figures 4.1-4.5 & 5.1-5.5. The symbol `[...]` represents the given line is too long to fit in this paper.

For each of the models, the continuations of both songs are very different from the original song (and each other). This is to be expected since the models are only given a short line as context to start generating the song. The output generated by both the pre-trained baselines is coherent and tells some form of a semantic story. Both of these LLMs generate very stylistically similar content as well. The lyrics however lack the rhythmic flow of rap. Our fine-tuned model on the other hand produces lyrics with subject content similar to what one comes to expect from the genre. Furthermore, the lyrics seem to have much more of a flow to them. We also did not need to truncate the lines for our model. Other models often generate a very long line, forgetting they should generate rap lyrics. This observation is backed up by results on the syllable count difference (Table 3).

Unfortunately, the continuation of the line 'godfathers in the house' by our fine-tuned model yields explicit and offensive lyrics. This is a by-product of fine-tuning it on our dataset which contains a lot of vulgar, offensive, and racist language. While such language is not acceptable from a moral point of view, it constitutes an important characteristic of rap as a genre. When generating output, it would not be too difficult to censor certain inappropriate words if desired. However, to keep the content as consistent as possible with the task-specific dataset, we decided not to do so in this paper.

The continuation of the other line for our model, 'cut the music up', has a nice rhythm to it and the story is semantically coherent. On top of this, the lyrics are not explicit.

For both lines, continuations by our model appear to be the best of all the models, the structure and story are superior to that of the baselines and in-keeping with the genre.

7. Conclusions and Future Work

Large language models are a very powerful tool for a range of natural language processing tasks. In particular, it has been empirically shown throughout this report that they can be fine-tuned on smaller datasets for specific tasks like rap lyric generation. The fine-tuned Distil-GPT2-rap model we trained achieves the best perplexity among the tested models. It also produces intelligible lyrics consistent with the style of the genre in terms of rhythm and subject material. The lyrics predicted by our model are far more stylistically similar to rap than any of the baselines. Hence, it can be concluded that fine-tuning a large language model is an effective approach to rap lyric generation.

While the lyrics generated by our model have good story, structure, and rhyme, they do not capture the rhyming component as well as they could. In future work, we should aim to increase the rhyme density of the model.

In this study, a lighter version of the language model GPT-2 is fine-tuned on the dataset of rap lyrics. The decision to choose DistilGPT2 was due to the lack of computational resources required to train larger models. In future work, one could consider fine-tuning GPT2-Large. Moreover, nowadays, there are more powerful LLMs in existence such as Megatron-Turing Natural Language Generation 530B (Shoeybi et al., 2019) and GPT4 (Radford & Narasimhan, 2018). It can be safely assumed that training a larger and more powerful model on the rap dataset would lead to reduced perplexity. Moreover, we hypothesize that the increased complexity of such a model would lead to an increased ability to implicitly model rhythm and rhymes.

Large language models are trained on vast swathes of textual data. Fine-tuning an LLM does not require nearly as much data. However as is the case with many deep learning problems, having more data is often beneficial. In this study, Distil-GPT2 is fine-tuned on a dataset containing 1.6 million lines from a total of 21,000 rap songs from 226 different artists. These artists were chosen based on three

different rankings of English-language rappers. This ensured that the dataset was comprised of lyrics from some of the most respected and popular rappers. Including lyrics from additional artists would be the easiest way to expand the dataset and potentially improve the evaluation performance of the model. However, it is important to ensure that this does not detract from the quality of the data.

Moreover, large language models are also pre-trained on non-English corpus. Future work should address rap in other languages. We also note that lyrics are only one part of rap, and the other is music. It would be interesting to see models that are able to generate both at the same time.

Last but not least, one needs to consider the ethical aspects. We showed that lyrics generated by our model can be explicit and offensive. It is necessary to create models that are not biased while at the same can express the nature of rap.

8. Acknowledgement

The project was done as part of the course *Machine Learning Practical* at the University of Edinburgh. We thank the course organizers and tutors for offering the course and providing helpful feedback.

References

Aye, Gareth, Kim, Seohyun, and Li, Hongyu. Learning autocompletion from real-world datasets. 11 2020.

Daniels, Matt. The largest vocabulary in hip hop, 2019. URL https://pudding.cool/projects/vocabulary/index.html.

Devlin, Jacob, Chang, Ming-Wei, Lee, Kenton, and Toutanova, Kristina. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

Edwards, Paul. *How to Rap*. Chicago Review Press, 2009.

Eisenstein, J. *Introduction to Natural Language Processing*. Adaptive Computation and Machine Learning series. MIT Press, 2019. ISBN 9780262042840. URL https://books.google.co.uk/books?id=72yuDwAAQBAJ.

Hochreiter, Sepp and Schmidhuber, Jürgen. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997. doi: 10.1162/neco.1997.9.8.1735.

Holtzman, Ari, Buys, Jan, Du, Li, Forbes, Maxwell, and Choi, Yejin. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rygGQyrFvH.

Janzer, M. Uncharted progress: A musical analysis of the elements and evolution of rap. 2012. URL https://soundideas.pugetsound.edu/cgi/viewcontent.cgi?article=1205&context=summer_research.

Jurafsky, Daniel and Martin, James H. Speech and language processing: An introduction to speech recognition, computational linguistics and natural language processing. *Upper Saddle River, NJ: Prentice Hall*, 2008.

Liang, Hongru, Li, Qian, Wang, Haozheng, Li, Hang, Wei, Jin-Mao, and Yang, Zhenglu. Attae-rl²: Attention based autoencoder for rap lyrics representation learning. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, pp. 7–8, Republic and Canton of Geneva, CHE, 2018. International World Wide Web Conferences Steering Committee. ISBN 9781450356404. doi: 10.1145/3184558.3186902. URL https://doi.org/10.1145/3184558.3186902.

Loshchilov, Ilya and Hutter, Frank. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101, 2017. URL http://arxiv.org/abs/1711.05101.

Malkin, Nikolay, Lanka, Sameera, Goel, Pranav, Rao, Sudha, and Jojic, Nebojsa. GPT perdetry test: Generating new meanings for new words. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5542–5553, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.439. URL https://aclanthology.org/2021.naacl-main.439.

Malmi, Eric, Takala, Pyry, Toivonen, Hannu, Raiko, Tapani, and Gionis, Aristides. Dopelearning: A computational approach to rap lyrics generation. *CoRR*, abs/1505.04771, 2015. URL http://arxiv.org/abs/1505.04771.

Manjavacas, Enrique, Kestemont, Mike, and Karsdorp, Folgert. Generation of hip-hop lyrics with hierarchical modeling and conditional templates. In *Proceedings of the 12th International Conference on Natural Language Generation*, pp. 301–310, Tokyo, Japan, October–November 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-8638. URL https://aclanthology.org/W19-8638.

Manning, Christopher D., Raghavan, Prabhakar, and Schütze, Hinrich. *Introduction to Information Retrieval*. Cambridge University Press, 2008. doi: 10.1017/CBO9780511809071.

McCoy, Austin. Rap music, 09 2017. URL https://oxfordre.com/americanhistory/view/10.1093/acrefore/9780199329175.001.0001/acrefore-9780199329175-e-287.

Nikolov, Nikola I., Malmi, Eric, Northcutt, Curtis G., and Parisi, Loreto. Conditional rap lyrics generation with denoising autoencoders. *CoRR*, abs/2004.03965, 2020. URL https://arxiv.org/abs/2004.03965.

Potash, Peter, Romanov, Alexey, and Rumshisky, Anna. GhostWriter: Using an LSTM for automatic rap lyric generation. In *Proceedings of the 2015 Conference on*

Empirical Methods in Natural Language Processing, pp. 1919–1924, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1221. URL https://aclanthology.org/D15-1221.

Radford, Alec and Narasimhan, Karthik. Improving language understanding by generative pre-training. 2018.

Radford, Alec, Wu, Jeff, Child, Rewon, Luan, David, Amodei, Dario, and Sutskever, Ilya. Language models are unsupervised multitask learners. 2019.

Sanh, Victor, Debut, Lysandre, Chaumond, Julien, and Wolf, Thomas. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *NeurIPS EMC^{2}Workshop*, 2019.

Shoeybi, Mohammad, Patwary, Mostofa, Puri, Raul, LeGresley, Patrick, Casper, Jared, and Catanzaro, Bryan. Megatron-lm: Training multi-billion parameter language models using model parallelism. *CoRR*, abs/1909.08053, 2019. URL http://arxiv.org/abs/1909.08053.

The CMU Pronouncing Dictionary. The cmu pronouncing dictionary, 2023. URL http://www.speech.cs.cmu.edu/cgi-bin/cmudict.

Vaswani, Ashish, Shazeer, Noam, Parmar, Niki, Uszkoreit, Jakob, Jones, Llion, Gomez, Aidan N, Kaiser, Lukasz, and Polosukhin, Illia. Attention is all you need. In Guyon, I., Luxburg, U. Von, Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Wu, D. and Addanki, K. Learning to rap battle with bilingual recursive neural networks. *Twenty-Fourth International Joint Conference on Artificial Intelligence.*, 2015. URL https://www.aaai.org/ocs/index.php/IJCAI/IJCAI15/paper/viewPaper/11437.

Wu, Dekai, Addanki, Karteek, Saers, Markus, and Beloucif, Meriem. Learning to freestyle: Hip hop challenge-response induction via transduction rule segmentation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 102–112, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://aclanthology.org/D13-1011.

Xue, Lanqing, Song, Kaitao, Wu, Duocai, Tan, Xu, Zhang, Nevin Lianwen, Qin, Tao, Zhang, Weiqiang, and Liu, Tie-Yan. Deeprapper: Neural rap generation with rhyme and rhythm modeling. In *Annual Meeting of the Association for Computational Linguistics*, 2021.