PopNet: Evaluating the Use of LSTMs and GPT-2 for Generating Pop Lyrics From Song Titles

Stanford CS224N Custom Project | Mentor: Yuyan Wang

Jason Ah Chuen
jahchuen@stanford.edu

Eunji Lee
eunjilee1@stanford.edu

Katherine Wu
kjwu00@stanford.edu

Abstract

Many artists now use lyricists to write the lyrics for their songs. We thought that it would be interesting to implement models which are able to take the place of a pop lyricist and generate song lyrics. Currently, LSTM models have been used to generate lyrics and verses, but not songs, and GPT-2 models have been shown to be effective in creative text generation problems, but have not yet been used to generate song lyrics. We implemented LSTM models and fine-tuned GPT-2 models to take in a song title then generate either 1) a line of lyrics, 2) the lyrics for a song verse, or 3) the lyrics to an entire song because we thought it would be interesting to characterize the behavior of LSTMS at generating longer pieces of text, and employ GPT-2 on a new task. Through perplexity scores, BERT scores, and human evaluation results, as well as qualitative evaluation, we are able to see that our finetuned GPT-2 and LSTM models are able to greatly outperform our baseline, the out-of-the-box pre-trained GPT-2 model in generating pop lyrics, verses, and songs. Through our human evaluation, we find that a fine-tuned GPT-2 is able to generate realistic pop lyrics and verses, and decent pop songs. The fine-tuned GPT-2 model outperformed the LSTM model in all three generation tasks, most likely due to difficulty that the LSTM cell state has in preserving upstream information and the difficulty that LSTM attention has in identifying different relevant parts of the input for different parts of the output.

1 Introduction

Pop artists often have lyricists that write the lyrics to their style of song. We believe that it would be interesting to see whether or not we could create a system that is able to take the place of a pop lyricist.

Existing methods for lyric generation include rule-based systems, which are focused on syntax and thus are unable to produce songs with consistent, coherent meaning. LSTMs have also been used to generate lyrics and verse, but not complete songs. We implement a LSTM and a fine-tuned GPT and compare their abilities to generate pop song lyrics, verses, and songs. We chose LSTM because it is a very common approach to solving sequence forecasting tasks, and has been used to in lyric and verse generation. We wanted to explore the use of LSTMs to generate songs, to see how it would perform in generating longer pieces of text. We chose to explore GPT because of its strong capacity as a language model as proven by evaluations in recent papers. Furthermore, GPT-2 is a non-linear model and has not yet been used in a lyric, verse, or song generation task before.

We find that both the fine-tuned GPT-2 model and the LSTM outperform the baseline out-of-the-box GPT-2 model across all three tasks, and that the fine-tuned GPT-2 model was able to generate realistic lyrics and verses, and decent songs.

2 Related Work

Traditional lyric generation methods are typically rule-based and based on rhymes. Barbieri et al. used a Markov process to generate song lyrics with a defined rhyme and meter scheme [1]. Watanabe et al. proposed a topic transition model to generate single lines of lyrics that follow certain rules for accent and syllable patterns [2]. These traditional methods are focused on the syntax of words, and are unable to generate lyrics with coherent meaning. Our methods focus less on the syntax of the words, and instead aim to generate lyrics, verses, and songs with coherent meaning.

More recently, deep learning methods have been shown to be useful for text generation. Manajavacas et al. implemented Recurent Neural Networks (RNN) to generate Hip-Hop lyrics, finding that RNNs are sensitive to model scaling (character or word-level) and that a hybrid form that integrates both word and character words yields improvements [3]. A number of papers have shown that the Long Short-Term Memory (LSTM) networks, a version of the RNN where each unit is gated, is useful for

lyric generation. Potash et al. implemented a rap verse generator using a LSTM without any human-defined rhyme scheme, line length, and verse length rules, and showed that it was able to generate novel lyrics that reflect the rhyming style of an artist [4]. Wu et al. used a LSTM with a hierarchical attention model which captures the context at both a sentence and document level to implement a Chinese lyric generation system which takes in one line of lyrics and generates the following line of lyrics [5]. These previous papers aim to generate shorter pieces of text: either lyrics or verses. In addition to lyrics and verses, we also generate entire songs, investigating the ability for the LSTM to generate longer creative texts.

The Transformer and more specifically the GPT-2 model have been shown to be a powerful text generator [6, 7]. The GPT-2 model has also been used to generate text for other artistic purposes other than lyric generation, such as poetry generation. Liao et al. adopt a simple GPT model without any human crafted rules or features to generate forms of Chinese poems, and was able to generate poems that were more well-formed and coherent than those for existing methods based on RNNs [8]. Bena and Kalita finetuned a pre-trained GPT-2 model to the task of dream poetry generation, finding that the generated poems are able to retell a story or event in the first person perspective well, but are syntactically simplistic and narrative [9]. To our knowledge, there has been no paper describing the application of GPT-2 to the lyric generation task. As the GPT-2 model has been relatively successful in generating poetry, we believe that it will also be successful in generating pop lyrics, verses, and songs.

3 Approach

We fine-tuned a GPT and implemented a LSTM model to generate song lyrics in the style of a particular artist. Our rationale behind using GPT is that it achieved state-of-the-art results on many language modeling tasks including those that involve text generation similar enough to our current task, such as poetry generation, and the model can be easily trained on a specific downstream task like ours. As for the LSTM approach, we believed we could leverage its strong capabilities in sequence forecasting tasks. We used LSTM in contrast to a simple RNN because remembering long-term information is crucial for our lyric generation task.

For one of our GPT models, we used the pretrained OpenAI GPT-2 model [6]. For our baseline, we did not fine-tune the model and used the model directly to generate song lyrics given a song title. When generating song lyrics, we formatted our input to the model as a sequence of tokens to the decoder: '<|startoftext|> <song title here> ??'.

As shown in the original paper [6], the pretraining is done by using a standard language modeling objective to maximize the likelihood $L_1(U) = \sum_{n=i} \log P(u_i|u_{i-k}, ..., u_{i-1}; \theta)$ where U is an unsupervised corpus of tokens.

A multilayer Transformer decoder applies multi-headed attention over the input context tokens followed by position-wise feedforward layers to produce an output distribution over target tokens (copied from original paper [6]):

$$h_0 = UW_e + W_p$$
$$h_l = \texttt{transformer-block}(h_{l-1}) \forall i \in [1, n]$$
$$P(u) = \texttt{softmax}(h_n W_e^T)$$

where $U = (u_{-k}, ..., u_{-1})$ is the context vector of tokens, n is the number of layers, W_e is the token embedding matrix, and W_p is the position embedding matrix.

We then fine-tuned the OpenAI GPT-2 model [6] on our lyric generation task. When fine-tuning, we formatted the input to the model as '<|startoftext|> <song title here> ?? <song lyrics here> <|endoftext|>'. We use the model to generate lyrics the same way we used our baseline model. We followed this tutorial (https://towardsdatascience.com/fine-tune-a-non-english-gpt-2-model-with-huggingface-9acc2dc7635b) to finetune and use the OpenAI GPT-2 model. The following objective function is maximized during fine-tuning:

$$L_2(C) = \sum_{(x,y)} \log P(y|x^1, ..., x^m)$$

where $x^1, ..., x^m$ is one instance of input tokens along with its corresponding label y, and $P(y|x^1, ..., x^m) = \texttt{softmax}(h_l^m W_y)$ where h_l^M is the final transformer block's activation and W_y are the parameters of the added linear output layer.

For our second type of model, we implemented a long short-term memory (LSTM) model [10] by adapting the Assignment 4 code. Because we are aiming to generate lyrics that correspond well to a given song title and artist, we model this as a sequence-to-sequence (Seq2Seq) problem. The source is an existing song title for a given artist, and the target is the lyric generated for that song title. See Figure 5 in Appendix A for a diagram of the overview of our Seq2Seq model, with an example output lyric for the Rihanna song "Watch N' Learn". Note that the encoder is bidirectional, while the decoder is unidirectional.

The hidden state and cell state of the encoder, as described in Assignment 4, are represented by:

$$h_i^{enc} = [h_i^{\overleftarrow{enc}}; h_i^{\overrightarrow{enc}}] \ \ where \ \ h_i^{enc} \in \mathbb{R}^{2h \times 1}, h_i^{\overleftarrow{enc}}, h_i^{\overrightarrow{enc}} \in \mathbb{R}^{h \times 1}, 1 \leq i \leq m$$

$$c_i^{enc} = [c_i^{\overleftarrow{enc}}; c_i^{\overrightarrow{enc}}] \ \ where \ \ c_i^{enc} \in \mathbb{R}^{2h \times 1}, c_i^{\overleftarrow{enc}}, {}_i^{\overrightarrow{enc}} \in \mathbb{R}^{h \times 1}, 1 \leq i \leq m$$

and the hidden state and cell state of the decoder are represented by:

$$h_t^{dec}, c_t^{dec} = Decoder(\overline{y_t}, h_{t-1}^{dec}, c_{t-1}^{dec}) \ \ where \ \ h_t^{dec} \in \mathbb{R}^{h \times 1}, c_t^{dec} \in \mathbb{R}^{h \times 1}$$

The hidden state captures what information should be passed to the next sequence, while the cell state stores information from previous intervals. Each LSTM unit (as shown in Appendix A Figure 6) consists of four gates which are able to control the flow of information, as described in the following six equations (as described in the notes for lectures 5 aand 6): $i_t = \sigma(W^{(i)}x_t + U^{(i)}h_{(t-1)})$ (input gate), $f_t = \sigma(W^{(f)}x_t + U^{(f)}h_{(t-1)})$ (forget gate), $o_t = \sigma(W^{(o)}x_t + U^{(o)}h_{(t-1)})$ (output gate), $c'_t = \tanh(W^{(c)}x_t + U^{(c)}h_{(t-1)})$ (new cell state), $c_t = f_t \circ c_{t-1} + i_t \circ c'_t$ (cell state), $h_t = o_t \circ \tanh(c_t)$ (hidden state). By using an LSTM (which has a cell state and gates) instead of an RNN (which does not), we are able to better preserve information from the beginning of the sequence and avoid disappearing and exploding gradients, which we believed would help the LSTM model will perform well on our lyric generation tasks.

When observing outputs from our LSTM model, we noticed that many of the outputs that were generated were much shorter than the outputs in the training set. In order to correct for the short length, we modified the LSTM's decoding beam function to generate longer outputs. We determined that the target length lyric, verse, and song are 300, 25, and 5, respectfully, by referencing the average lengths of these components in the training data set. When evaluating new candidate hypotheses, if we add a end of sentence token and the output is shorter than the target length, then we penalize the output by subtracting the difference between three times the target length and the current length from score for the hypothesis. We then choose the top k candidate outputs and proceed with beam search. This penalizes shorter outputs more, as the difference between the target length and the current length is larger for shorter outputs, and penalizes outputs less for ending as the outputs get longer, until the outputs reach the target length.

4 Experiments

4.1 Data

We collected our own data set by scraping lyrics using Lyrics Genius (https://github.com/johnwmillr/LyricsGenius), a Python client which uses the Genius Lyrics API (https://docs.genius.com/). We collected the title and lyrics of the top 20 songs for 75 pop artists (the entire list of pop artists is included in the appendix A). We then used this to create three data sets: one where the title of the song is associated with a most frequent lyric in the song, one where the title is associated with a unique verse from the song, and one where the title is associated with the entire song. We created these data sets with the intent to be able to use the title of a song to generate a pop lyric, verse, or song (depending on the data set).

4.2 Evaluation Metric

We use three evaluation metrics: perplexity score, BERTScore, and human evaluation.

Perplexity is a measure of how well a generated sentence follows the structures of the sentences in the training set. A language model with a higher perplexity score is less likely to generate text which would be identified as a valid text similar to those in the training set, and vice versa. Our implementation of perplexity uses the NLTK maximum likelihood estimation (MLE), and we implemented a modified version of unigram perplexity from this article [11].

For a discrete probability distribution p, the perplexity PP is defined as: $PP(p) = 2^{H(p)}$ where $H = -\Sigma_x p(x) \log p(x)$ is the entropy, measured in bits, of p. A perplexity of k, in the context of evaluating a language model, means that the model is as confused in choosing the next word as if it had to choose randomly between k words.

BERTScore is a quantitative metric for evaluating generated text. It uses pre-trained BERT embeddings to compute the cosine similarity between two texts' token embeddings. We wrote a script to automatically run the implementation from [12] using the package released by the authors of the paper.

We used the model with hyperparameters tuned to have the lowest perplexity scores and the highest F1-BERTScore to be representative of the model class, for each generation task for both LSTM and fine-tuned GPT-2.

For the preliminary human evaluation of each of our tuned models, we ask 22 independent participants to look at a title along with either four lyrics, verses, or songs: the real lyric, verse, or song along with either lyrics, verses, or songs generated by our baseline out-of-the-box GPT-2 model, the tuned fine-tuned GPT2 model, and the tuned LSTM model. For each title, the participant is asked to rank the texts from most likely to be a real piece from a song with that title (1) to least likely to be a text from a song with that title (4). This is an adapted version of the approach taken in the paper [4]. We use 15 titles each for lyric, verse, and song for human evaluation, with no repeats in titles, for a total of 45 titles. The lower the average ranking of the generated outputs, the better the model is, where scores are between 1 (the best possible score for a model) and 4 (the worst possible score for a model).

4.3 Experimental Details

4.3.1 GPT-2

We fine-tuned our GPT-2 model for 300 epochs each on their respective data sets (lyrics, verses and songs) since from initial observations 300 epochs seemed to produce the best results while fitting within our project's time constraints and our virtual machines' space constraints.

Afterwards, we used both the baseline and fine-tuned models for text generation. We set the minimum number of words produced to be some number between the mean and median of the data sets from the training data such that the decoder would avoid producing the end-of-sentence token until then. We also prevented the model from generating the "\n" token since that was used in our data sets to demarcate examples and '<newline>' was instead used to demarcate new examples. For the decoder, we used top-K sampling mechanism with K set to 50, which we found did slightly better on perplexity and BERT scores after evaluating a certain number of the output results with the outputs from using beam search and other values of K.

For the GPT2 models, we also performed a hyper-parameter search on the repetition penalty. The rationale behind this is that words in lyrics, verses and songs are often repeated to a certain extent and we wanted to find the right balance between creative word generation and repetitions. This penalized sampling is used on top of the top-K sampling mechanism in the decoding step and works by discounting the scores of previously generated tokens [13].

Finally, in order to generate longer pieces of text, we set the minimum length of the generated output to be 5, 25, and 300 for lyrics, verses, and songs respectfully.

4.3.2 LSTM

For LSTM, we modified two hyperparameters: the number of iterations per epoch and the dropout rate. For the number of iterations, we tried 200, 500, and 1000. For dropout rate, we tried 0 (no dropout), 0.1, and 0.3. We found that the number of iterations had minimal effect at the lyric level, but at the song and verse level, 1000 iterations resulted in the lowest dev/test perplexity. We also found that including dropout at a rate of 0.1 resulted in the lowest dev/test perplexity. We trained three separate models while performing this hyper-parameter search: one each for lyric, verse, and song generation.

Parameter search for LSTM was constrained by time and virtual machine credits, so we selected these two parameters due to the fact that we wanted to prevent both overfitting and underfitting as much as possible. Further research would involve choosing more hyperparameters to train and trying more values for those hyperparameters.

Furthermore, we modified the beam search for LSTM to include a penalty for brevity, as described in Section 3. We determine that the target length lyric, verse, and song are 5, 25, and 300, respectfully, by considering the average lengths of these components in the training data set. When evaluating new candidate hypotheses, if we add a end of sentence token and the output is shorter than the target length, then we penalize the output by subtracting the difference between three times the target length and the current length from score for the hypothesis. We then choose the top k candidate outputs and proceed with beam search.

4.4 Results

We used the validation set to tune our LSTM and fine-tuned GPT-2 models by calculating the perplexity score (as described in Section 4.2) of the output in order to select a single best model for lyric, verse, and song generation for LSTM, fine-tuned GPT-2, and baseline, for a total of 12 selected models.

We completed testing using the test set on our tuned baseline out-of-the-box GPT-2 model, LSTM (with the modified beam search), and fine-tuned GPT-2 model for generating lyrics, verses, and songs.

We calculated the perplexity score for each of the outputs for each of these 12 models, in addition to the gold (true) lyrics, songs, and verses, as shown in Table 1. We also calculated the BERT score for the test outputs for our 12 models (as described in section 4.2), as shown in Table 2.

	Lyric	Song	Verse
Baseline	945.73	1623.016	1223.344
LSTM	725.7	161.534	89.825
GPT2	440.54	145.016	184.83
Gold	535.448	901.05	685.81

Table 1: Perplexity Scores

	Lyric	Song	Verse
Baseline	0.551551	0.534183	0.5341
LSTM	0.8538	0.8138	0.8369
GPT2	0.80351	0.838	0.842198

Table 2: BERT Scores

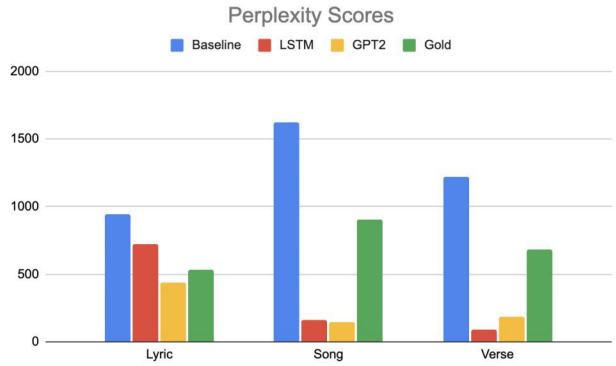

Figure 1: Graphical comparison of perplexity scores

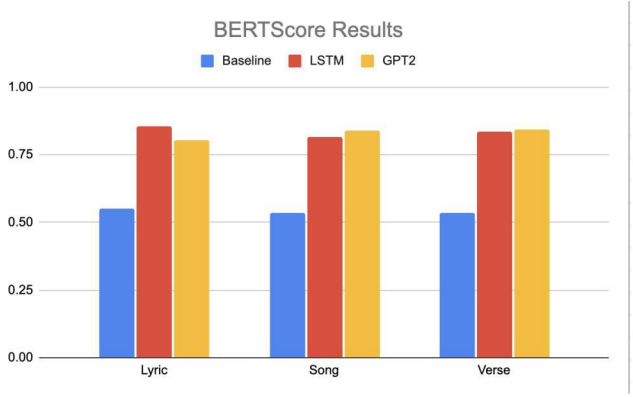

Figure 2: Graphical comparison of BERT scores

For all of our models, we recognize the limitations in using numerical evaluations to measure performance. A language model with a higher perplexity score generates text which is less likely to be identified as similar to those in the training set. While the baseline performs more poorly than our LSTM and fine-tuned GPT-2 models at the lyric, song, and verse levels in terms of perplexity, we see that both our LSTM and fine-tuned GPT-2 model have a lower perplexity than the gold songs and verses, despite the fact that the gold songs and verses are the real songs and verses and were rated the best most frequently by our human evaluators. This may be due to the fact that a model's possible outputs are more constrained, while different pop artists may vary vastly in their choice of words and song structures.

BERTScore is a measure of similarity between each output and its corresponding gold lyric, verse, or song. As expected, we see that the fine-tuned GPT-2 model and LSTM model were able to generate output lyrics that were more similar to the gold lyric than the baseline model was.

We also calculated the average lengths of the outputs for all 9 models as well as the real text as shown in Table 3, in order to see how well they would match with the real lyrics, verses and songs. Length (number of words), although fairly variable across artists and songs, is another feature which we hope to have our model mimic. We hope that our model's output does not produce output which is an atypical length. The results are pretty much as expected for our GPT-2 model, as we had given the model a minimum length and the output generated was about the minimum length (5 for lyrics, 25 for verses, and 230 for songs), when we include tokens which were not counted as words. The results were as expected for our LSTM for lyric and verse generation, as we had modified the decoder to penalize stopping before a target length (5 for lyrics, 25 for verses, and 230 for songs) and the outputs were about this length. The songs generated by the LSTM were shorter than we expected. This may have been caused by the difficulty that our LSTM encountered in learning the structure of a song, due the difficulty the cell state has in preserving information for long sequences.

From human evaluation, we scored each of the models as well as human lyricists (the real lyrics) for each type of output (lyric, verse, and song) as described in section 4.2. A model with a lower average score is able to generate better lyrics, with model average scores ranging between 1 and 4. The results are shown in Table 4.

	Real	GPT-2	LSTM	Baseline
Lyrics	6	8	11	16
Verses	38	28	26	26
Songs	390	271	163	29

Table 3: Mean text lengths

	Real	GPT-2	LSTM	Baseline
Lyrics	1.71	1.81	2.87	3.62
Verses	1.82	1.75	2.65	3.78
Songs	1.47	1.86	2.92	3.74

Table 4: Human evaluation results

Both our LSTM and our fine-tuned GPT-2 model consistently outperform our baseline out-of-the-box GPT-2 model. We further observe that the fine-tuned GPT-2 model out performs our LSTM model, and produces lyrics and verses which are realistic, as its score for lyrics and verses were lower than those for the real lyrics and verses. Our fine-tuned GPT-2 model was able to produce decent songs, but our human evaluators were still often able to discern the difference between the GPT-2 generated song and the real song.

Our human evaluation results are as or better than expected for the GPT-2 model. The GPT-2 model had been shown in the past to produce high quality text for creative text generation tasks so we expected it to score similarly or slightly worse in comparison to the real lyrics and verses. It is expected that our GPT-2 model does not perform as well on generating songs, as songs are much more complex and long, so this task is much more difficult.

Our human evaluation results were as expected for the LSTM model. We did not expect LSTM to perform as well as GPT-2 due to both the difficulty that the cell state has in preserving upstream information, and the difficulty that attention has in identifying different relevant parts of the input for different parts of the output, particularly when the input is short as the titles are.

5 Analysis

5.1 LSTM

We fine-tuned three LSTM models: one to generate pop song lyrics, one to generate pop song verses, and one to generate pop song lyrics. We find that the LSTM was able to generate decent lyrics and verses, but were unable to generate realistic songs. A few sample lyrics, verses, and songs generated by the LSTM are in Figure 3.

We noticed that at the lyric and verse level, there is less repetition, however at the song level, the same combinations of verse and chorus / pre-chorus along with a lyric was repeated over and over until the desired length of a song is reached. For the song title "Hurt Again", our LSTM model produced a plausible first verse and pre-chorus, but then repeats pre-chorus along with a specific lyric 21 more times.

This may have occurred for multiple reasons. Firstly, LSTMs have difficulty remembering past information that occurred more than a few sentences back in a sequence; this is due to the forget gate determining how much of the previous state to be preserved in the current state. Over time, the information preserved from a state from long ago is not factored in. Furthermore, attention for LSTM would lose efficacy when predicting longer texts which include repetition: pop songs are long and often have repeating sequences of lyrics and verses, and the titles which we give the model to generate text are short. As a result the attention mechanism may have had difficulty knowing which parts of the title are the most "important" and where they would be most important for the song. Hence, it could result in our model producing repeating sequences.

Furthermore, a significant characteristic of pop songs is that the title (or parts of the title) often appears somewhere in the lyrics. Our LSTM outputs often did not reflect this pattern, while GPT2 outputs had text that was related to the title more frequently, whether it included direct words from the title or the meaning of the title was reflected in the generated lyrics. This is likely because we fine-tuned the embeddings on pop song data for GPT2, while the embeddings in LSTMs are fixed. Therefore, GPT2 is able to capture the structure of pop songs, lyrics, and verses better than LSTMs, resulting in more realistic structures in outputs at the song and verse level, in addition to more relevant content to the title.

5.2 GPT2

We fine-tuned three GPT-2 models: one to generate pop song lyrics, one to generate pop song verses, and one to generate pop song lyrics. We find that the GPT-2 models are able to generate realistic lyrics and verses, and decent songs. A few samples of texts generated by these fine-tuned GPT-2 models is shown in Figure 4.

For all three of our fine-tuned GPT-2 models, we observed the model was able to generate lyrics which were highly relevant to the topic mentioned in the title. For example for the title "I Won't Give Up", the generated lyric describes not giving up a problem. For the title "I Miss You", the generated verse describes how someone wants someone they know to miss them. For the title "Galway Girl" the generated song describes a girl. We believe that the encoder for our fine-tuned GPT-2 models are able to successfully understand that the title and encode its meaning and convey it to the decoder which is able to decode the main points of the title.

Our fine-tuned GPT-2 model is also able to more successfully generate songs which take the form of a song than the LSTM model was. This can be observed in the GPT-2 generated songs "Halo" and "Galway Girl" in Figure 4. We can see that the model generated a song which is composed of a series of verses, where each verse is a series of short lyrics split by a single newline, and verses are separate by two newlines. The GPT-2 model was also able to learn the order of the verses in a song. In "Halo" we can observe that the model learned that the verses in a song which are labelled verse should be numbered in counting order. In "Galway Girl" we can see that the GPT-2 model learned that Pre-Choruses should proceed Choruses. Both songs are also

Title: What a Time
LSTM Generated Lyric: For you and I

Title: Leaving my Love Behind
LSTM Generated Lyric: I'll being love love

Title: Morph
LSTM Generated Song:
[Pre-Chorus]
I'm gonna be a little more

[Pre-Chorus]
'Cause I'm in love with you

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

[Pre-Chorus]
I don't wanna be alone

Title: Never GIve Up on the Good Times
LSTM Generated Verse:
So love you I know
I'll never ever call from me
'Cause I don't know no
'Cause I don't know no
'Cause I don't care

Title: Like a Girl
LSTM Generated Verse:
If told you down
I like my girl and I'm out
'Cause I don't need to you
'Cause I don't want to you

Title: Hurt Again
LSTM Generated Song:
[Verse 1]
You've got me in love again
And I won't let go
And I won't let go

[Pre-Chorus]
'Cause I've been wondering
And I won't let go

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

[Pre-Chorus]
If I had you in love with you

Figure 3: Lyrics, verses, and songs that were generated by LSTM models.

representative of how the fine-tuned GPT-2 model learned that songs should not start with the chorus. This confirms our initial assumptions that GPT2 would be able to learn patterns in most downstream tasks including whole song generation, especially the patterns which are common through all of the training examples.

However, our fine-tuned GPT-2 model which generated song lyrics would occasionally include sets of lyrics that were taken verbatim from the training data. This can be observed in the GPT-2 generated song examples in Figure 4. For example, we can see that in the GPT-2 generated song for the title "Galway Girl", the lyrics in pre-chorus are taken Alicia Key's "Girl on Fire", one of the songs in the training data set and in "Give Your Heart a Break" lines from the chorus are taken from Demi Lovato's "Heart Attack". We believe that this is caused by over-fitting. This most likely occurred due to multiple issues. Firstly, the GPT-2 model is extremely large with many parameters and our data set was on the smaller side of 1500 samples. Furthermore, the input to our song generation model is often very short, averaging 2.5 words, while the training output of the model is quite long, averaging 333 words. This means that the model is not given much information, but is expected to generate a lot of text. Our model likely learned simple rules, associating certain words in the tiles with verbatim lyrics from that song. In the case of "Galway Girl" which had lyrics taken from "Girl on Fire", it is likely that having "Girl" in the title became highly associated with the lyrics from "Girl on Fire".

6 Conclusion

We demonstrated the ability of a fine-tuned GPT-2 model to generate realistic lyrics and verses given a song title. We found that a GPT-2 is able to generate decent song lyrics given a song title. The fine-tuned GPT-2 model and LSTM model were both able

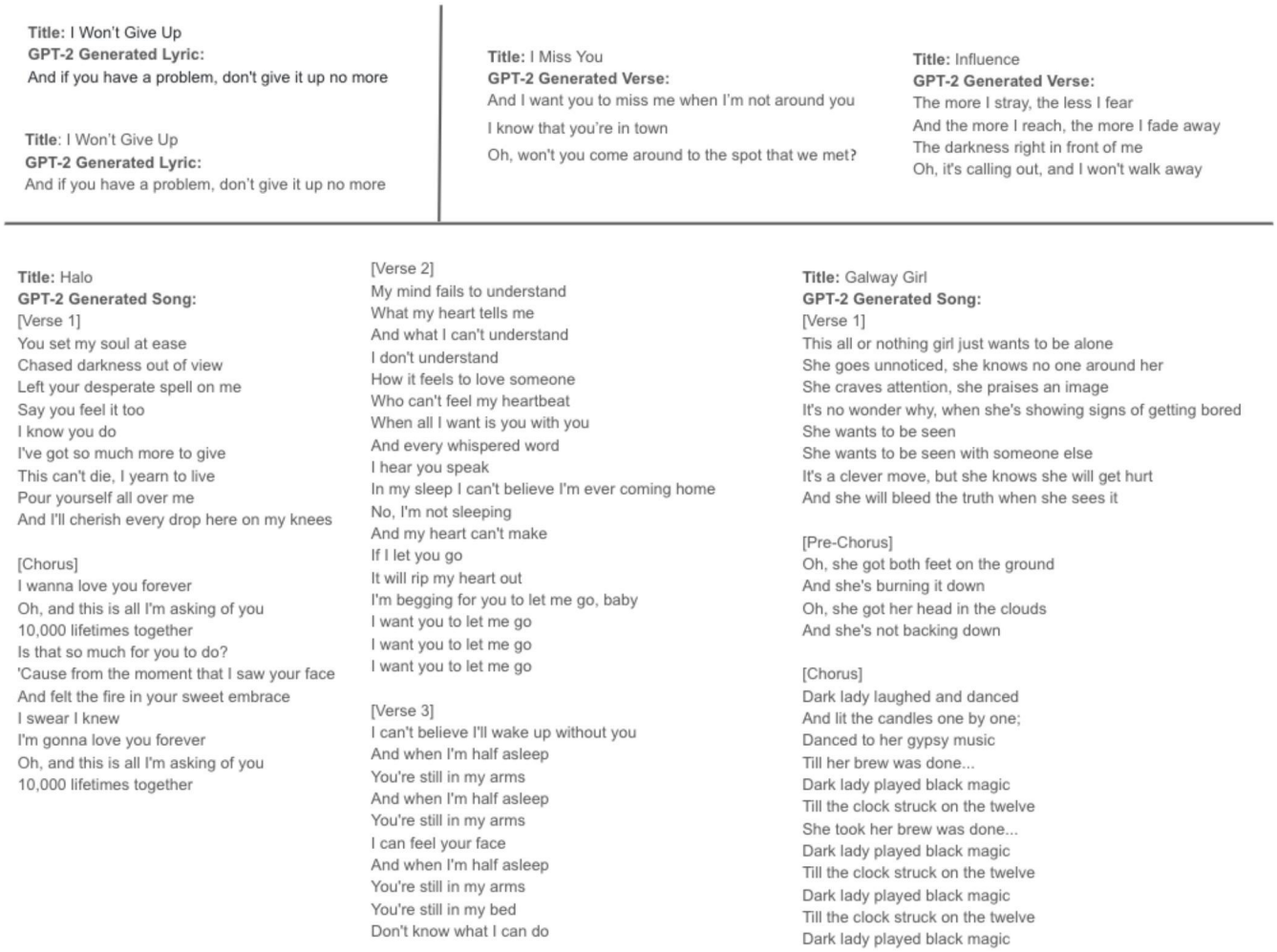

Figure 4: Lyrics, verses, and songs that were generated by fine-tuned GPT-2 models.

to consistently outperform the out-of-the-box GPT-2 model on all three generation tasks (lyric, verse, and song), and the GPT-2 model outperformed the LSTM model across all three generation tasks.

We were primarily limited in terms of computing power, computer memory, and time. With more resources, we may explore further hyper-parameter tuning for parameters such as temperature. By increasing the temperature, that is scaling up the logits before applying softmax in the decoding step of GPT-2, we can make the probability distribution of words softer, hence we are more likely to pick low probability words and this could be a source of "creativity" in lyric/verse/generation. We would also consider modifying the attention mechanism for LSTM so that it would be better at generating songs which do not have as much repetition. One possible way to do this could be by altering the hidden state of a word by appending the attention for the previous word, allowing information about the model to propagate information about attention for previous words. In order to improve our GPT-2 model, we could incorporate regularization and dropout in order to prevent the GPT-2 model from over-fitting.

References

[1] Gabriele Barbieri, Francois Pachet, Pierre Roy, and Mirko Degli Esposti. Markov constraints for generating lyrics with style. volume 242, 08 2012.

[2] Kento Watanabe, Yuichiroh Matsubayashi, Kentaro Inui, and Masataka Goto. Modeling structural topic transitions for automatic lyrics generation. In *Proceedings of the 28th Pacific Asia Conference on Language, Information and Computing*, pages 422–431, Phuket,Thailand, December 2014. Department of Linguistics, Chulalongkorn University.

[3] Enrique Manjavacas, Mike Kestemont, and Folgert Karsdorp. Generation of hip-hop lyrics with hierarchical modeling and conditional templates. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 301–310, Tokyo, Japan, October–November 2019. Association for Computational Linguistics.

[4] Peter Potash, Alexey Romanov, and Anna Rumshisky. GhostWriter: Using an LSTM for automatic rap lyric generation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1919–1924, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[5] Xing Wu, Zhikang Du, Y. Guo, and H. Fujita. Hierarchical attention based long short-term memory for chinese lyric generation. *Applied Intelligence*, 49:44–52, 2018.

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

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[8] Yi Liao, Yasheng Wang, Qun Liu, and Xin Jiang. Gpt-based generation for classical chinese poetry. *CoRR*, abs/1907.00151, 2019.

[9] Brendan Bena and Jugal Kalita. Introducing aspects of creativity in automatic poetry generation, 2020.

[10] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[11] How can i calculate perplexity using nltk, Jan 2019.

[12] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with BERT. *CoRR*, abs/1904.09675, 2019.

[13] Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. CTRL: A conditional transformer language model for controllable generation. *CoRR*, abs/1909.05858, 2019.

[14] Originally published by Manik Soni on. Understanding architecture of lstm cell from scratch with code., Jul 2020.

A Dataset

The artists that we retrieved songs from where Tove Lo, Ariana Grande, Beyoncé, Rihanna, Taylor Swift, Billie Eilish, Post Malone, Dua Lipa, Harry Styles, Justin Bieber, Drake, Ed Sheeran, Lewis Capaldi, Khalid, Bruno Mars, Maroon 5, H.E.R, Lady Gaga, Imagine Dragons, P!nk, Halsey, Katy Perry, Madonna, Britney Spears, Selena Gomez, Mariah Carey, Justin Timberlake, Adele, The Weekend, Miley Cyrus, Kelly Clarkson, Shawn Mendes, Demi Lovato, Christina Aguilera, Bazzi, Alicia Keys, Sam Smith, Celine Dion, Usher, Backstreet Boys, Jennifer Lopez, Janet Jackson, Cher, Whitney Houston, Tina Turner, Spice Girls, Paula Abdul, Gwen Stefani, Jessica Simpson, Nick Jonas, One Direction, Joe Jonas, Adam Lambert, Kesha, Josh Groban, Lizzo, Lana Del Rey, Carly Rae Jepson, Twenty One Pilots, Fifth Harmony, Jason Mraz, Dido, Ellie Goulding, Camila Cabello, Bebe Rexha, Charlie Puth, Zayn Malik, Tori Kelly, 5 Secconds of Summer, Andy Grammer, Julia Michaels, Alessia Cara, Anne-Marie, Rachel Platten, and Melanie Martinez.

B Diagrams

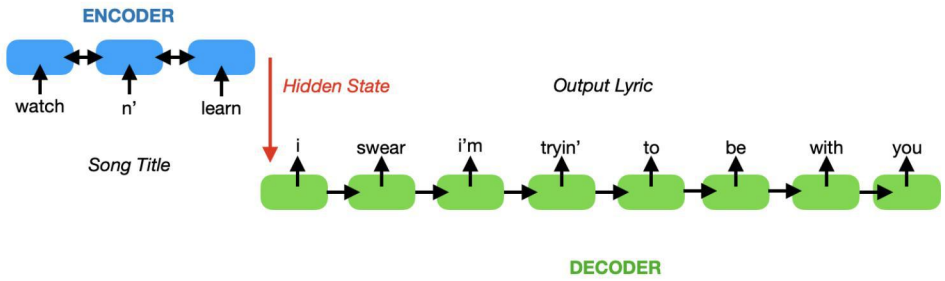

Figure 5: Overview of Seq2Seq LSTM

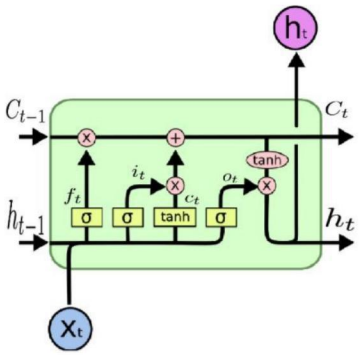

Figure 6: Detailed View of LSTM Cell [14]