Can a machine win a Grammy? An evaluation of AI-generated song lyrics

Junlan Lu, Magdalini Eirinaki
San Jose State University
San Jose, CA, USA
Email: {junlan.lu, magdalini.eirinaki}@sjsu.edu

2021 IEEE International Conference on Big Data (Big Data) | 978-1-6654-3902-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/BigData52589.2021.9671431

***Abstract*—Through lyrics, pitch, and rhythm, music is a natural way of expressing one's thoughts. As one of the essential music composition elements, lyrics' composition is complicated as it requires creativity and follows a particular rhythm pattern. In this work we design and train two neural network models for composing lyrics in three genres and propose scoring functions to select and evaluate the generated songs. We treat this problem as a text generation task, and optimize for features particular to song lyrics, including lyrics' quality, rhyme density, and sentiment ratio, for lyrics in different genres. The neural networks we have experimented with are generative adversarial network (GAN)-based transfer learning for a deep learning model and long short-term memory (LSTM)-based deep learning model. In addition to quantitative evaluation, we also conducted user studies, inviting 25 people to rate the generative songs selected by the scoring functions. Our findings show that the GAN-based models perform better than LSTM-based models and the scoring functions are useful in selecting good songs.**

***Index Terms*—Lyrics generation, Unsupervised learning, Evaluation metrics, Long short-term memory, Recurrent neural networks, Generative pretrained transformer**

I. Introduction

As part of music production, music lyrics composition requires both creativity to keep up with new music trends and professional lyrics' writing skills to produce sophisticated rhyme patterns. Due to the complicated requirements, music production remains a task reserved for only a group of professional artists. In this work we explore whether neural network language models can generate *meaningful*, and *genre-appropriate* song lyrics from only a few words or sentences that the user provides, and address the challenge of evaluating generative art models (in this case, lyrics) in the absence of ground truth by proposing and evaluating several scoring functions.

The first language model was trained based on the generative pretrained transformer (GPT-2)[1] deep learning model. The second language model was trained on a traditional long short-term memory (LSTM)-based deep learning model. The two types of language models were trained with three lyrics' genres, resulting in six models.

Sequence data processing has been used for text processing and text prediction. An early-stage recurrent neural networks (RNN) model is the bidirectional recurrent neural network (BiRNN) [1]. BiRNN splits the RNN's nodes into forwarding computation direction and backward computation direction. Recently, most RNN models use the long short-term memory (LSTM) network [2] [3] in which they can learn long-term dependencies and predict the sequence data much better. Our model uses character-level models, in which we tokenized each character into vectors from the corpus of letters and train the model with the LSTM networks.

Traditional machine learning (ML) learns from a given dataset, is isolated from other datasets, and it cannot pass knowledge from one model to another. In transfer learning, the knowledge obtained from a model, such as weights and feature inputs, can be shared or passed to generating a second, newer model [4]. A straightforward example of showing how humans share knowledge across real-life tasks is how children learn to swim. If knowing how to swim is essential knowledge, then learning to swim the sidestroke style is advanced knowledge derivable from the inherent wisdom.

A successful machine learning model relies mostly on data being used. However, it is challenging for unpopular tasks to find a large amounts of data used to train the model. Transfer learning shows advantages where little data is available for generating the model. While the base model of the transfer learning requires using large, easy to obtain, and more generally to a problem data, the datasets used to train the advanced model can be smaller but more relevant to the task [4]. In this work, one type of model has used transfer learning techniques. We obtained a pre-trained text generator model, GPT-2 Text Generation Model trained on an enormous dataset, as the base model. The model is an open-source, deep-learning model provided by OpenAI for predicting the next word with some context. Then, we trained and produced several newer models that can generate song lyrics by retraining the original model with our datasets, which include lyrics (text). In this way, we kept the middle layer parameters of the GPT-2 model and changed the parameters in the last layer of the neural network.

Our focus is to not only generate text using some input, but also to ensure that the result is meaningful and appropriate for the particular genre. Unlike traditional machine learning, which has a training and a testing dataset and can calculate the accuracy score of trained models, there is no established way of evaluating the performance of generative objects, since there is no ground truth. In the context of art, and in particular lyrics'

[1]OpenAI GPT-2 model https://openai.com/blog/better-language-models/

generation, one of the critical challenges is to automatically evaluate and score machine-generated lyrics without human interaction. In this work, we achieve this by first studying what features human written songs have, and analyzing three unique features of 39,721 human-written songs. Based on the results, we developed and propose a set of scoring metrics that can automatically evaluate and score generative songs quickly.

The scoring metrics can validate and score a large number of generated songs in a short time. Inspired by how generative text has been evaluated in previous work and articles about song composition [4] [5], our proposed approach to evaluate generated songs takes into consideration the song's structure, rhyme density, and sentiment ratio. In addition to some quantitative evaluation, we also evaluate the validity of the scoring metrics through a user study. In particular, we selected highly scored generated songs and presented them alongside real songs to end users as part of our quantitative evaluation.

We discuss our process and proposed methodologies as follows: Literature review is described in Section II. The dataset preprocessing, lyrics' analysis, and scoring metrics are presented in Section III. The lyrics' generator models are described in Section IV. The evaluation metrics and methodology are described in Sections V and VI respectively. The results of the qualitative and quantitative evaluation are detailed in VII. Finally, discussion on future work and our conclusions are presented in Sections VIII and IX respectively.

II. Literature Review

A. Text Generation

Generative adversarial networks (GANs) [6] are highly successful in image generation. Although GAN is an unsupervised machine learning algorithm, it frames the problem as a supervised learning task with two sub-models. It has a generator model that is trained to generate new objects and a discriminator model to distinguish real or generative objects. The feedback that the discriminator model provides can improve the generator model step by step. GANs have evolved from the original model through time, and improvements have been made and discussed in research such as Wasserstein GANs [7], and Loss-sensitive GANs [8]. Some research has applied GANs to text generation, such as sequence generative GANs [9], and neural dialogue GANs [10]. Other successful research that uses GANs for text generation [11] [12] also has achieved satisfactory results.

Recurrent Neural Networks (RNNs) have been shown to be very advantageous in sequence data processing due to their recurrent architecture. Many works have used Long-Short Term Memory (LSTM) and RNN algorithms for text generation [13] [14]. LSTM architecture is based on RNN and is famous for having a "memory" of the results of the previous steps. Other fields that apply LSTM based on text generation include machine translation [15] and speech recognition [16].

Another type of text generator uses the methodology of randomly choosing the next words from a dictionary to fill gaps in some incomplete sentences, based on the rules of phrase structure or a predefined template. Research using this method includes the haiku generator [17].

B. Transfer Learning

According to the survey paper of Weiss et al. [18], the challenges of transfer learning are concluded in the categories of what to transfer, when to transfer, and how to transfer. Since the problem addressed in our work is an unsupervised text generation task, the transfer learning method we could use is unsupervised transfer learning with feature-based-representation transfer. The work done by another team used transfer learning with deep learning for a classification problem with limited labeled data [19], and the authors in [20] used transfer learning for image classification and recognition tasks. Another survey paper has summarized transfer learning in the fields of reinforcement learning [21].

C. Lyrics Generation

Automatic lyrics generation is a type of text generation. Besides natural language processing, lyrics generation can involve the additional step of analyzing song structures, such as numbers of words and sentences, and the frequency of each type of term used in each song [5]. In [4], the authors focus on rap lyrics generation. They started by examining different rhyme types employed in rap lyrics followed by analyzing typical song structures used often. Using this knowledge, they created a neural language model that can predict a rap song's next line. The automatic lyrics generation model also varies with the language of the songs. In [22] the authors have used deep learning neural networks to create a Korean song model. They discovered that reversing Korean lyrics' sentences before feeding them into the machine learning algorithm performs better than using the original lyrics. The Korean language structure has a predicate in the latter part of the sentence. Other text generation techniques include using a deep learning algorithm [23] and studying the essential grammar for analyzing sentences [24].

In this work, our objective is to generate (and evaluate) original content for different music genres, focusing on lyrics written in English. We analyzed human written songs' features and based on the analytic results, and we created scoring functions to rate generative lyrics based on genres. We also contributed and evaluated lyrics generators' performance from traditional LSTM-based models with transfer learning-based models. We also have conducted a user study to validate our proposed scoring functions.

III. Lyrics Analysis

Automatically generating lyrics that resemble human-composed ones is a very challenging task and requires understanding what components human-composed lyrics have. This section introduces our lyrics' analysis based on different genres to understand the distinctive features present in each genre. In our analysis we used the 380000-lyrics-from-metrolyrics dataset downloaded from Kaggle[2].

[2]380000-lyrics-from-metrolyrics https://www.kaggle.com/gyani95/380000-lyrics-from-metrolyrics

A. Data Preprocessing

We imported a CSV file containing 362,237 lyrics to the Google Colab notebooks to execute code on Google's cloud servers. The pandas-DataFrame library is used for importing and cleaning irrelevant data. Figure 1 shows a sample of the original dataset before preprocessing. It contains lyrics from 1968 to 2016 from eleven genres, including rock, pop, hip-hop, country, metal, etc. The dataset also includes lyrics in different languages, such as English, Spanish, Korean, etc. The raw dataset statistics are shown in Table I, where we list the number of songs for each genre. In this work we focused on English songs composed in recent years in the pop, hip-hop, and rock genres. Therefore, we first collected all the pop, hip-hop, and rock genre songs between 2010 and 2016. Then we used Nakatani Shuyo's language-detection library[3] to select songs that are English. Given an input of a string, the library can return the language detected. Therefore, we looped all the pieces to the language detection algorithm and collected the lyrics detected as English. Then we cleaned the lyrics column by removing bad characters, numbers, and punctuations, and we split the cleaned dataset into three CSV files; each file contains lyrics from one genre. A sample of the pop genre's CVS file after preprocessing is shown in Figure 2. The CSV file contains only a column of cleaned songs. The resulting dataset, after preprocessing, contains 39,721 English song lyrics from 2010 and 2016, and it includes 11,718 pop songs, 9,097 hip-hop songs, and 18,906 rock songs. The statistics of the dataset after preprocessing are shown in Table II.

index	song	year	artist	genre	lyrics
55	back-to-black	2013	beyonce-knowles	Pop	(Andre 3000)\nI, I, I left no time to regret\n...
56	mine	2013	beyonce-knowles	Pop	[Verse 1: Beyonce]\nI've been watching for the...
57	superpower	2013	beyonce-knowles	Pop	[Verse 1]\nWhen the palm of my two hands hold ...
58	haunted	2013	beyonce-knowles	Pop	[Intro: Presenter]\nThe winner is\nBeyonce Kno...
59	flawless	2013	beyonce-knowles	Pop	[Intro]\nYour challengers are a young group fr...

Fig. 1: Snapshot of the dataset before preprocessing.

TABLE I: Lyrics Statistics of the Raw Dataset

Genres	Number of songs
Rock	131377
Pop	49444
Hip-hop	33965
Metal	28408
Country	17286
Jazz	17147
Electronic	16205
R and B	5935
Indie	5732
Folk	3241
Other	53497
Total	362237

B. Lyrics' Analysis and Metrics

To evaluate machine-generated songs automatically, we need to produce ways to define good songs. Pudaruth et al. [5] considered the song structure, such as the frequency of words used, numbers of words, and sentences. In another work focusing on rap songs, Hirjee and Brown [25] analyzed the different rhyme types used in the lyrics. Inspired by their research, we looked through the lyrics in the dataset and identified some unique structures that only songs have: good songs should rhyme, and are emotional. Therefore, we identified three main categories to evaluate generative songs. The first category is the study of lyrics' *quality*, which considers the number of lines, words, and syllables in a song. The second category analyzes *rhyme density*. It finds whether the last word of a sentence and its adjacent sentences in a song form a rhyme. The third category is the *sentiment analysis* of the lyrics. It determines how many sentences in a song are emotional. Our assumption is that different genres have different values for each category, which makes them unique compared to other genres. We thus calculate the proposed metrics separately for each genre.

[3]Nakatani Shuyo's language-detection library https://pypi.org/project/langdetect/

	Pop
0	Andre . I, I, I left no time to regret. Kept m...
1	Verse Beyonce. I've been watching for the sig...
2	Verse . When the palm of my two hands hold eac...
3	Intro Presenter. The winner is. Beyonce Knowle...
4	Intro. Your challengers are a young group from...
...	...
11713	When the photographs you're taking now. Are ta...
11714	I met Moko jumbie,. He walks on stilts through...
11715	Chill on the hollow ponds. Set sail by a kid. ...
11716	Celebrate the passing drugs. Put them on the b...
11717	When the serve is done. And the parish shuffle...

11718 rows × 1 columns

Fig. 2: Dataset for the pop genre after preprocessing.

TABLE II: Lyrics Statistics after Preprocessing

Genres	Number of songs
Pop	11718
Hip-hop	9097
Rock	18906
Total	39721

1) Lyrics Quality Analysis: The quality of a song is defined as the distance of the number of lines, words, and syllables of each piece from the median of the number of lines, words, and syllables in most songs. To find the median values, we analyzed the lyrics of 40k songs. These 40k songs consist of pop, hip-hop, and rock genres, as shown in Table II. We calculated the average and median of the number of lines, words, and syllables of each category's lyrics. The results can be seen in Table III, and verify our initial hypothesis that different genres present different characteristics. This is more evident between genres that have litlle overlaps (e.g. rock and hip-hop), whereas for genres that are more similar and often overlapping (e.g. pop and rock) the differences are less striking. Using the median values in the Table III as the

standard, if the number of words is n, the number of sentences is m, and the number of syllables is k, we define the lyrics quality score as shown:

$$QualityScore = (n + m + k) - (\tilde{n} + \tilde{m} + \tilde{k}) \quad (1)$$

TABLE III: Lyrics' Analysis

Median	Pop	Hip-hop	Rock
Number of lines	42	60	33
Number of words	259	438	201
Number of syllables	349	582	272

Using Equation 1, we calculated the quality scores of all lyrics. The distribution of the the quality scores for the pop, hip-hop, and rock genres is shown in Figures 3, 4, and 5 respectively. From the three distribution diagrams, we can see they are symmetric distributions, with most scores falling between -125 and 125 in the pop songs distribution, the most frequent score is 0, and there are about 600 songs at scores 0. From the distribution graph of hip-hop songs, we can see that the scores are concentrated between -65 and 500. The most frequent score is around 0, and there are about 550 songs around 0. In the distribution graph of rock songs, the distribution is more even. The scores appearing are concentrated between -125 and 125, the highest frequency score is also 0, and there are about 800 songs at 0. In comparing the three figures together, we found that the highest frequency of their scores all appeared around 0, i.e. the median.

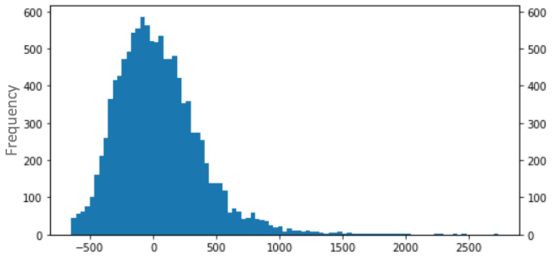


Fig. 3: Quality score distribution for pop lyrics.

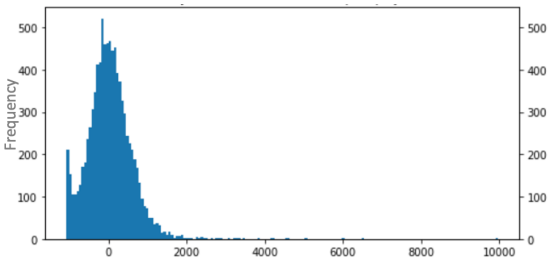


Fig. 4: Quality score distribution for hip-hop lyrics.

2) Rhyme Density Analysis: Every good song has rhymes, but not all songs have a verse. To understand the rhyme proportion in each genre, we analyzed the rhyme sentences' ratios in each genre. We say that two sentences rhyme if the last syllable's pronunciation of the two sentences' ending

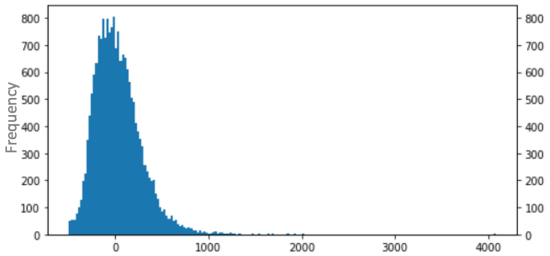


Fig. 5: Quality score distribution for rock lyrics.

words is the same. In order to compare the syllables of words, we used a syllable library of CMU[4]. This library takes an English word as input, and it outputs its syllables. For example, the syllables of the word "lucky" are ['L', 'AH1', 'K', 'IY0'], and the syllables of the word "happy" are ['HH', 'AE1', 'P', 'IY0']. The last syllable of the two terms are both 'IY0', so these two words rhyme according to the definition. Similarly, in the lyrics of a song by Taylor Swift, "Burning brighter than the sun. And when you're close, I feel like coming undone", the last syllable of the ending words of the two sentences "sun" and "undone" are both 'N', so these two sentences rhyme.

We went through every sentence in every song, and compared the last word of each sentence with the last term of adjacent sentences to discover how many rhyme pairs are in a song. We define the rhyme density as the ratio of the number of rhyme pairs to the total number of sentences in a song:

$$RhymeDensity = \frac{|RhymePairs|}{|Sentences|} \quad (2)$$

Using Equation 2, we calculated the rhyme density of all songs. The distribution diagrams are shown in Figures 6, 7, and 8. From the diagrams, we can see that the rhyme density distribution is not uniform across all songs; some songs have a rhyme density of 0. When we investigated this finding by checking our dataset, we realized that the skewed distribution is because not all words can be found in the CMU syllable dictionary, so even if there are rhyme pairs, they are not counted. The second reason of this is that, while our definition of rhyme pairs expects the last syllable of the two words to be the same to be counted as rhyming, often the rhyme can also be found in the second-to-last syllable of two words. Improving the search for rhyming verses is part of our plans for extending and further refining this work. Taking into consideration the songs with non-negative rhyme density, we observe that the median pop song rhyme density is around 16%, the median rhyme density for hip-hop song is approximately 17%, and the median rock song rhyme density is about 16%.

3) Sentiment Analysis: A good song is touching, and we want to check how emotional each piece is. We used an NLTK sentiment analysis library called SentimentIntensityAnalyzer [5] to discover the sentiment intensity of each song. A sentence can emote with a positive attitude, negative attitude, or no

[4]http://www.speech.cs.cmu.edu/cgi-bin/cmudict
[5]https://www.nltk.org/api/nltk.sentiment.html

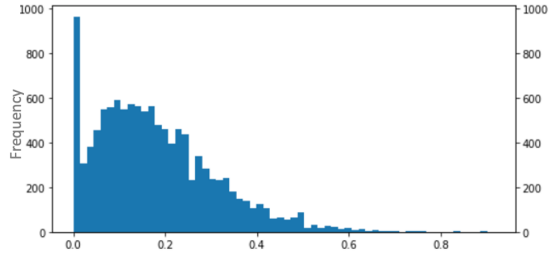


Fig. 6: Rhyme density distribution for pop lyrics.

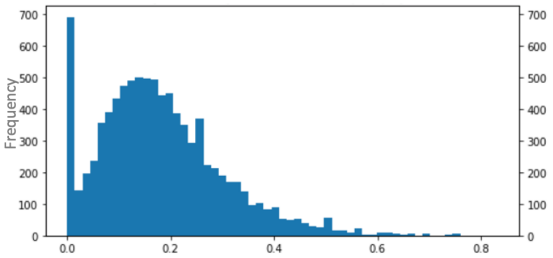


Fig. 7: Rhyme density distribution for hip-hop lyrics.

emotion. The library takes a paragraph as the input, checks each sentence in the section and outputs the percentages of the positive, negative, and neutral sentences in the paragraph. We define the song's sentiment score as the sum of the positive and negative percentage scores. A sentiment score of 0.20 means 20% of the sentences in a piece have emotions. Note that we treat both positive and negative emotions equally. The equation for the sentiment score is as below:

$$SentimentScore = negativeEmotion + positiveEmotion \quad (3)$$

Using the above Equation 3, we calculated the sentiment score for all songs. We can find the distribution of the pop genre's sentiment score, hip-hop genre, and rock genre in Figures 9, 10, and 11 respectively. The three distribution graphs show that they are all normal distributions. The distribution chart of the pop genre shows between the score 0.15 and 0.3 have the most data. The highest frequency is at score 0.2, and there are about 600 data points (songs) in this score. The distribution graph of the hip-hip genre shows that most scores are also between 0.15 and 0.3. The highest frequency score is 0.25, and there are about 570 data points in this score. Most data is between score 0.15 and 0.3 in the rock distribution

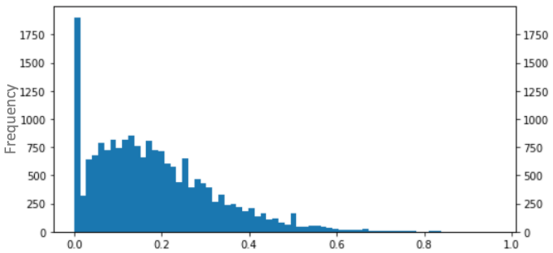


Fig. 8: Rhyme density distribution for rock lyrics.

graph, the highest frequency score is around 0.22, and there are 800 data points in the highest frequency score.

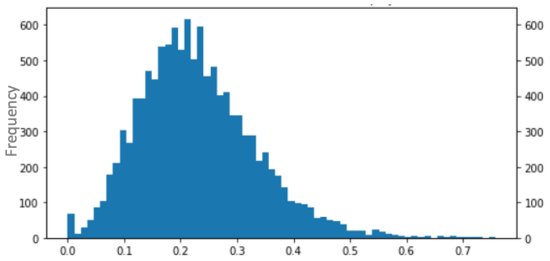


Fig. 9: Sentiment score distribution for pop lyrics.

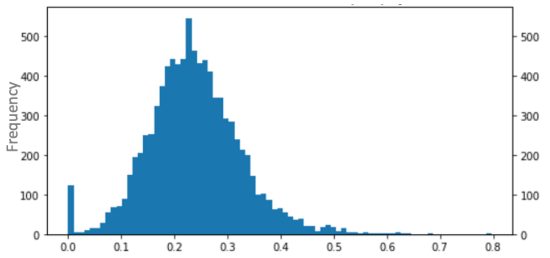


Fig. 10: Sentiment score distribution for hip-hop lyrics.

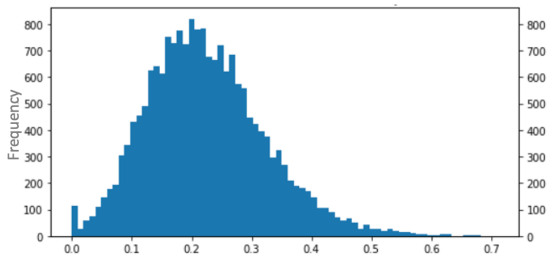


Fig. 11: Sentiment score distribution for rock lyrics.

IV. Lyrics Generator

In this section, we introduce the two models we trained for song lyrics generators, which used GPT-2 and transfer learning networks and LSTM based neural networks. Those models were trained on Google Colaboratory[6] with NVIDIA-SMI 455.23.05 GPU, Driver Version: 418.67, CUDA Version: 10.1. The Python programs were developed using Google Colaboratory's notebook.

A. GPT-2 and Transfer Learning Model

Our GPT-2 transfer learning models are trained based on a pre-trained GPT-2 Text Generation Model. For this work, we used the model that introduced 124 million parameters. The model with 124 million parameters is one of the two models released to the public and can be fine-tuned, and it is also the smallest model that will take a shorter time to fine-tune and generate text. With the GPT-2 model, we fine-tuned the last layer of the neural network by feeding our dataset into it.

[6]https://colab.research.google.com/

B. LSTM Deep Learning Model

Each characteris encoded into boolean vectors, and the vectors are fed into the model. The deep learning model has an LSTM layer with 128 output shape, a dense layer with 156 output shape, and an activation layer with 156 output shape. The models were trained with a batch size of 32, 50 epochs, and RMSprop optimizer. The learning rate starts at 0.001 and decreases by a factor of 0.2 each time the loss rate plateaus, and the minimum learning rate is set to 0.0001. Figures 12 a, b, and c represent the loss rate of pop, hip hop, and rock songs, respectively. The x-axis is the number of steps, and the y-axis is the loss rate. We observe that their loss rates have dropped from 2 to around 1. There is a steep drop in each figure when the loss rate is about to be flat because we defined that when the loss rate flats, we reduce the learning rate; therefore, the loss rate reduces.

C. Lyrics Generation

Section IV-A and IV-B described the setups of the two types of lyrics generation models. In this section, we describe the process of model training and text generation. We have used three datasets for training the models, and each dataset consists of songs from one genre. The dataset we have used is 1/10 of the datasets we used for lyrics analysis in Section III. Because the RNN-LSTM-based models took a long time to train and to generate text, we only used 1/10 of the dataset due to time constraints. The generated models are "rnn-pop," "rnn-hiphop," "rnn-rock," "gpt2-pop," "gpt2-hiphop," and "gpt2-rock." When generating lyrics in each model, we need to give a seed of inputs to start with, and we fed random words as inputs. Each model has generated 50 songs, so we generated 300 songs in total.

V. Lyrics Evaluation Metrics

The biggest challenge in creating a generative model is to evaluate the quality of the generated objects. Unlike supervised learning, where we have the training and test sets to evaluate predicted results, there is no ground truth in text generation since the algorithms generate new text. The new text can be diverse, so there is no established answer. Therefore, one challenge in this work was to discover ways to evaluate whether the lyrics generated from the model are what humans would write. In Section III, we defined and analyzed the lyrics' quality (*QualityScore*), rhyme density (*RhymeDensity*), and emotional density scores (*SentimentScore*) of pop, hip-hop, and rock songs and found the scores' distributions. In this section, we present a scoring function to judge the lyrics generated by our generators. We defined the perfect score for a song as 100 points, with a maximum of 20 points for lyrics quality, 40 points for rhyme density, and 40 points for sentiment ratio.

Our scoring functions depend on the distribution diagrams in Figures 3 - 11. We classified the lyrics' scores into three intervals. The first interval is for songs that perform well, the second interval is for songs that perform below average, and the third interval is for songs that perform poorly. We gave the same score to the songs within the same interval. The upper and lower bound of each interval is determined according to the respective genre's distribution diagram and standard deviation. We defined the first interval for songs that score around the mean obtained within one standard deviation. The range of the second score interval is between one and two standard deviations. The third interval is between two and three standard deviations. Each genre has independent scoring functions.

Tables IV, V, and VI, list the mappings of the respective range of scores to the final score for each genre. The tables are created by following the rules we discussed to find the upper and lower bound of each interval using standard deviation. Let the final score for lyrics quality be *Q*, rhyme density be *R*, and sentiment ratio be *S*. then the final score of the whole song is calculated as:

$$finalScore = Q + R + S \tag{4}$$

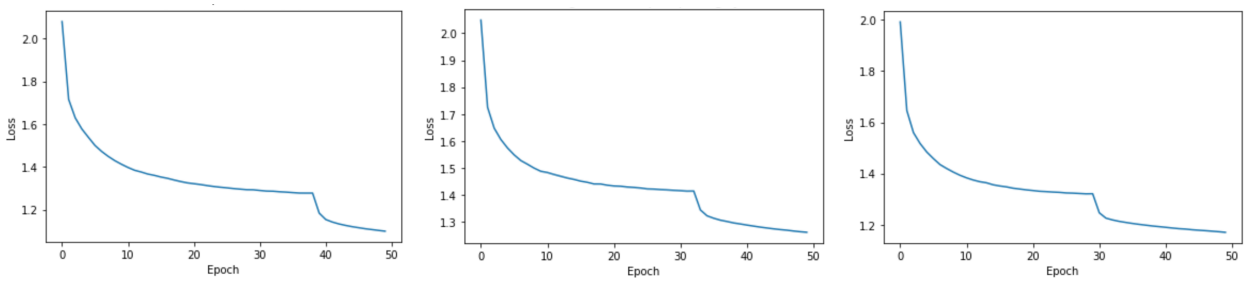
VI. Evaluation Methodology

In Section III, we analyzed three unique features that human written song lyrics have, that could be used to determine which machine-generated songs (lyrics) are good, and applied them to a dataset containing song lyrics from three genres. In Section V, we discussed producing the lyrics scoring metrics based on the results from Section III. Section IV discussed the setups of two lyrics generator models, which used two neural networks. We fed the three datasets to the two models, resulting in 6 trained lyrics generators.

To evaluate how well each generator performs, which is one of the primary objectives of this work, we generated 50 songs in each lyrics generator. The generators were fed into a seed of random words as inputs to start with and processed as described in Section IV-C. Then we scored each piece by using the lyrics evaluation metrics we designed, and we call this scoring process automatic evaluation. We will discuss automatic evaluation in Section VI-A below. Then, we picked top-scored songs for each genre and mixed them with real songs, and conducted a user study presenting them blindly to users to evaluate. The user study is discussed in Section VI-B. The results of both evaluations are discussed in Section VII.

A. Automatic Evaluation

The goal of the automatic evaluation is to select the good performing lyrics out of many generated lyrics. In Section V, we listed three evaluation standards for the three genres. We first preprocessed the 300 generated songs by calculating the sum of their quality score Q, rhyme score R, and sentiment ratio S of each song, calculated based on the range of the scores and each song's genre. The final scores' distributions of the 300 songs (50 songs $*$ 6 generators) are shown in Figure 13. The higher the score, the closer to the standard of human-composed songs. We discuss our findings in Section VII-A. Using the scores, we were able to select 11 good songs to use in our user study.



(a) pop songs (b) hiphop songs (c) rock songs

Fig. 12: Training loss rate of RNN models of pop, hip hop, and rock songs, respectively.

TABLE IV: Quality Score to Final Score

Score (Q)	Pop quality score	Hip-hop quality score	Rock quality score
20	[-125 - 125]	[-125 - 125]	[-125 - 125]
10	[-550, - 125),(125, - 550]	[-700, - 125),(125, - 700]	[-500, - 125),(125, - 500]
15	(-inf, -550), (550, inf)	(-inf, -700), (700, inf)	(-inf, -500), (500, inf)

TABLE V: Rhyme Score to Final Score

Score (R)	Pop rhyme score	Hip-hop rhyme score	Rock rhyme score
40	[0.15, 0.2]	[0.15, 0.2]	[0.15, 0.2]
20	[0, 0.15), (0.2, 0.4]	[0, 0.15), (0.2, 0.4]	[0, 0.15), (0.2, 0.4]
5	[0.4, inf)	[0.4, inf)	[0.4, inf)

TABLE VI: Sentiment Score to Final Score

Score (S)	Pop sentiment score	Hip-hop sentiment score	Rock sentiment score
40	[0.15, 0.35]	[0.15, 0.35]	[0.15, 0.3]
20	[0, 0.15), (0.35, 0.4]	[0, 0.15),(0.35, 0.5]	[0, 0.15),(0.3, 0.4]
5	(0.4, inf)	(0.5, inf)	(0.4, inf)

B. User Study

There are two goals in our user study. The first goal is to determine which models generate songs that are more appealing to people. The second goal is to validate the proposed scoring metrics by evaluating whether they can select songs close to human-composed songs. To achieve this, we designed the following survey. We first picked 11 high-scoring songs from a pool of 300 generated songs, by selecting one or two songs from each model. We also selected three human written songs, one from each genre, from the original dataset, and added them to the survey.

Table VII shows the survey statistics. We selected two generated songs from all models except for the rnn-hiphop, in which we only picked one song for the user study because all other songs in the model were scored too low. By comparing the rating of all generated songs, we will observe which models produce better pieces. By comparing the rating results between generative songs and real songs, we can know if the scoring metrics have done an excellent job selecting lyrics from a generated song pool.

We mixed those 14 songs and blindly (in terms of the "source" being human or machine) presented them to users to rate the lyrics as one of "very good," "good," "poor," "very poor." The demographics of the 25 people who participated in the user study are shown in Table VIII.

TABLE VII: Survey Statistics

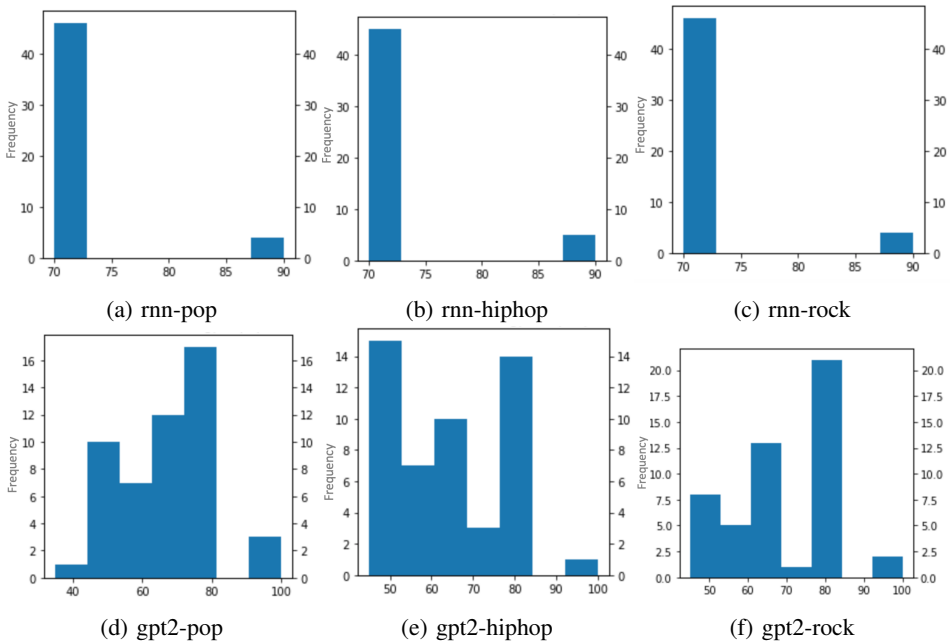
Model	Number of songs
rnn-pop	2
rnn-hiphop	1
rnn-rock	2
gpt2-pop	2
gpt2-hiphop	2
gpt2-rock	2
real-pop	1
real-hiphop	1
real-rock	1
total	14

TABLE VIII: Survey Demographics

Gender	Percentage (%)
Male	32
Female	68
Age	
20 - 30	60
30 - 40	16
40 +	24

We used Google Forms[7] to create our survey, which we shared via email. The full survey is included in [26]. A screenshot of the survey is displayed in Figure 14.

[7]Google Form https://www.google.com/forms/about/



(a) rnn-pop (b) rnn-hiphop (c) rnn-rock

(d) gpt2-pop (e) gpt2-hiphop (f) gpt2-rock

Fig. 13: Final score distribution of models.

VII. Results

In Section VI, we introduced evaluation methodologies for generative songs. We used an automatic evaluation methodology based on our proposed metrics to select top-performing songs from many machine-generated songs quickly. These songs are picked evenly from each generator. This process is done by giving each song scores, and the results of the scores are discussed in Section VII-A. From the results, we can analyze the trend of songs generated by each model. From our user study results in Section VII-B, we can also see which model generates better songs and if the evaluation process is effective.

A. Results of Automatic Evaluation

Figure 13 shows the score distributions of 300 songs generated from the six models. As a reminder, the higher the score, the closer the song is to a human-written one, with 100 being the top score. We notice that in the rnn-pop model, most songs are scored at 80 points, and a few are scored at 60 and 100 points. No song has a score between 65 to 80 and 80 to 95. This shows that most songs from this model perform above average, and a few songs are very good or very bad. In the rnn-hiphop model, most songs are scored at 70 points, and only two songs are scored at 90 points, and there are no other songs scored between 70 to 90 or above 90. This shows most songs generated by this model do not have good quality, and only a few songs are good. The score distribution graph of rnn-rock mode is very much like that of rnn-hiphop, with most songs having low scores and only a few pieces having high scores. From the graph of gpt2-pop, we can see that there are songs in every score interval from 40 to 100 points. Most songs are scored around 60 to 80, the second peak is at 50 points, and a few songs perform well at 100 points. This shows that the scores generated by this model are evenly distributed, and a few songs have high scores. In the gpt2-hiphop model diagram, the first peak is at 50 points, and the second peak is at 80 points, and a few songs have reached 100 points. In the model of gpt2-rock, we can observe that most pieces are scored from 80 to 70 points, and there are a few songs at 100 points.

Figures *a*, *b*, and *c* in Figure 13 represent rnn-based models. Most songs generated by those models are concentrated on one or two relatively low scores. Compared with the other two models, rnn-pop's model is better, with an average score of 10 points higher than the other two models. The scores of rnn-hiphop and rnn-rock model are notably biased towards low scores. Figures *d*, *e*, and *f* in Figure 13 show gtp2-based models. The scores of the three models are not as extreme as the ones of rnn-based models. They are distributed across almost every score interval. Comparing the three models together, the gpt2-pop model produces the highest scores.

With the information gathered from above, we can conclude that gpt-2-based models perform better than rnn-based models. We also have ranked each model in Table IX, based on our observation of the aggregate evaluation results, and we would expect the user study results to show a similar ranking, to validate our hypothesis that these are accurate scoring metrics for this type of analysis.

B. Results of User Studies

As previously discussed, our user study selected one or two songs from each generator and mixed them with real songs before presenting them to the users. The songs were presented in random order. The participants were told of the type of song

TABLE IX: Automatic Evaluation Results

Model	Evaluation results
rnn-pop	good
rnn-hiphop	poor
rnn-rock	poor
gpt2-pop	very good
gpt2-hiphop	good
gpt2-rock	very good

(pop, hip-hop, rock) but were not told whether it was a real or a machine-generated song. The full-length results of the user study survey included in [26]. Due to lack of space, here we summarize the major findings in Table X. In the table, we list the 14 songs that have been presented to the participants of the user study, along with the percentage of responses (user ratings) for each song. We have highlighted the most frequent and the second most frequent ratings.

The rnn-pop model songs appear to have a mixed rating of 'excellent' and 'poor', with most users agreeing that the first one is between 'very good' and 'good'. For the rnn-hiphop model, we only selected one song from all generated rock songs because other songs are scored too low to be included in the user study. The majority of the rating result is between 'good' and 'poor' with some rated as 'very good'. For the rnn-rock model, the result shows that both songs' ratings lean towards 'poor', although the second most frequent rating is 'good'.

The gpt2-pop model is rated as 'good'. The second most frequent rating for the first song is 'excellent', while it is 'poor' for the second song. The gpt2-hiphop model is a mix between 'good' and 'poor', and most people voted 'good' for the first song, whereas for the second song, most people voted

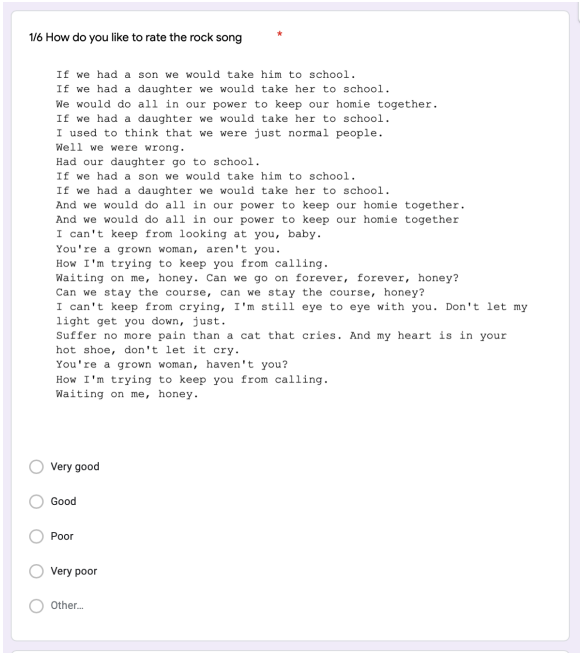


Fig. 14: A sample of user study survey.

it as 'poor'. However, the second song also got many 'very good' and 'good' reviews. For the gpt2-rock1 model, most people voted for 'poor' and then 'good', and for the second song, most people voted for 'good' followed by 'very good'. So, it's a mixed rating as well in favor of 'good'. The last three songs are real songs we selected from the dataset. Each piece belongs to one genre. Overall, the actual songs got high ratings, as expected. The most frequent rating for pop songs and rock songs is 'excellent', and for rock songs, the second most frequent rating is 'excellent'. It is interesting to see that even in the real songs' case, the majority of our participants voted especially the hiphop but also the rock songs as 'good' (instead of very good), showing some inherent selection bias in our participants pool that could also be one of the reasons why the hiphop and rock machine-generated songs were not favorably rated either.

Overall, rnn-based models received three low ratings and two good ratings and gpt2-based models got two poor ratings and four good ratings as the most frequent rating. Therefore, we can deduce that the gpt2 models are comparatively better than the rnn-based models, which aligns with our automatic evaluation findings summarized in Table IX. The best model is gpt2-pop, since it has two 'good' and one 'excellent' rating from the above ratings. Comparing the ratings of gpt2-pop model songs and the real songs, we can see that the best rated generative piece got scores as close as the real song, although not all the selected songs got good scores. Overall, by comparing the automatic evaluation results in Table IX with the user study results in Table X, we can find close correlations. They agree that gpt2-pop and gpt2-rock performances are the highest, gpt2-hiphop and rnn-pop perform good, while rnn-hiphop and rnn-rock perform the worst. Based on this, we can conclude that the scoring metrics represent human evaluation very well and can be used as an evaluation metric to automatically identify good songs without the need of human participation.

TABLE X: User Study Results

Songs	Very good(%)	Good(%)	Poor(%)	Very poor(%)
rnn-pop1	**56**	**20**	20	4
rnn-pop2	4.2	**33.3**	**50**	12.5
rnn-hiphop1	20	**24**	**48**	8
rnn-hiphop2	-	-	-	-
rnn-rock1	4	6.4	**24**	**8**
rnn-rock2	4.2	**37.5**	**45.8**	12.5
gpt2-pop1	**25**	**58.3**	8.3	8.3
gpt2-pop2	4	**48**	**9**	4
gpt2-hiphop1	16	**52**	**28**	4
gpt2-hiphop2	**26.8**	20.8	**50**	8.3
gpt2-rock1	**40**	24	**32**	4
gpt2-rock2	**37.5**	**41.7**	12.5	8.3
real-pop	**48**	**44**	8	0
real-hiphop	**28**	**64**	8	0
real-rock	**32**	**56**	8	4

VIII. Future Work

Additional work can be done to improve the lyrics' evaluation criteria. We considered the counts of the words, lines,

rhyme density, and sentiment ratio in the section. Although these are essential features, and we have achieved good results by incorporating them in our scoring metric, we can enhance the evaluation quality by adding additional standards. One is having a grammar evaluation step, checking if the generative sentences have proper grammar structure. The second one is checking if adjacent sentences make any sense putting them together and if the song's story is consistent. The last one is to check for plagiarism, i.e. whether the generative songs are fully or partially copied from the original dataset. We have already started working towards this direction.

Another area of further exploration would be to evaluate alternative models of lyrics generators, such as GANs. Regardless of the model, another thing we expect to have a positive impact in the quality of the generated songs is increasing the size of the dataset. We only used 1/10 of the preprocessed dataset due to time and computational constraints. If we could improve the algorithm's time complexity and have adequate time, we could improve the models.

Finally, one more area we can improve is in the automatic evaluation process. In our work, we have manually set up the scoring function for each genre. It would be interesting if we could consider the evaluation task as a classification problem. We could train several classifiers, one for each genre, and feed into each generated song to the classifiers and determine if the generated lyrics belong to where it is supposed to be. This method will make the process much more efficient and reduce the time used to manually set up the scoring functions.

IX. Conclusions

Lyrics composition is intricate as it requires artists to keep up with music trends and deliver creative work. This work presents lyrics composition models that can generate song lyrics in pop, hip-hop, and rock genres from a few words or sentences that the user provides. Since this is an unsupervised process, we proposed three lyrics evaluation metrics and an automatic evaluation methodology that can be used to evaluate the models' performance, and it is also used to filter out songs that perform poorly. We validated our models through a user study, where the participants were presented with a mix of human and machine-generated songs to rate. The results of the user study aligned very well with our automatic evaluation, further validating our hypothesis that the proposed scoring methodology can be adopted to evaluate machine-generated songs and select songs close to human standards.

References

[1] M. Schuster and K. Paliwal, "Bidirectional recurrent neural networks," *Signal Processing, IEEE Transactions on*, vol. 45, pp. 2673 – 2681, 12 1997.

[2] D. Pawade, A. Sakhapara, M. Jain, N. Jain, and K. Gada, "Story scrambler - automatic text generation using word level RNN-LSTM," *International Journal of Information Technology and Computer Science*, vol. 10, pp. 44–53, 06 2018.

[3] M. Islam, S. Sultana Sharmin, S. Abujar, and S. Hossain, "Sequence-to-sequence bangla sentence generation with LSTM recurrent neural networks," *Procedia Computer Science*, vol. 152, 01 2019.

[4] E. Malmi, P. Takala, H. Toivonen, T. Raiko, and A. Gionis, "Dopelearning: A computational approach to rap lyrics generation," *ACM SIGKDD International Conference*, pp. 195–204, 08 2016.

[5] S. Pudaruth, S. Amourdon, and J. Anseline, "Automated generation of song lyrics using cfgs," *2014 Seventh International Conference on Contemporary Computing (IC3)*, pp. 613–616, 08 2014.

[6] J. Islam and Y. Zhang, "GAN-based synthetic brain PET image generation," *Brain Informatics*, vol. 7, 12 2020.

[7] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," *34th International Conference on Machine Learning*, 01 2017.

[8] G.-J. Qi, "Loss-sensitive generative adversarial networks on lipschitz densities," *International Journal of Computer Vision*, vol. 128, 01 2017.

[9] L. Yu, W. Zhang, J. Wang, and Y. Yu, "SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient, journal = Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)," 09 2016.

[10] J. Li, W. Monroe, T. Shi, A. Ritter, and D. Jurafsky, "Adversarial learning for neural dialogue generation," *2017 Conference on Empirical Methods in Natural Language*, 01 2017.

[11] Y. Deng, C. Fu, and Y. Li, "CodeeGAN: Code generation via adversarial training," *5th International Conference, DependSys 2019*, pp. 18–30, 11 2019.

[12] S. Semeniuta, A. Severyn, and E. Barth, "A hybrid convolutional variational autoencoder for text generation," *2017 Conference on Empirical Methods in Natural Language*, 02 2017.

[13] A. Graves, S. Fernández, and J. Schmidhuber, "Bidirectional LSTM networks for improved phoneme classification and recognition," *15th International Conference*, pp. 799–804, 01 2005.

[14] A. Tjandra, S. Sakti, R. Manurung, M. Adriani, and S. Nakamura, "Gated recurrent neural tensor network," *2016 International Joint Conference on Neural Networks*, pp. 448–455, 07 2016.

[15] K. Cho, B. van Merriënboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, 06 2014.

[16] I. Serban, A. Sordoni, R. Lowe, L. Charlin, J. Pineau, A. Courville, and Y. Bengio, "A hierarchical latent variable encoder-decoder model for generating dialogues," *Thirty-First AAAI Conference on Artificial Intelligence*, 05 2016.

[17] M. Boden, *The Creative Mind: Myths and Mechanisms*, 11 2003.

[18] K. Weiss, T. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big Data*, vol. 3, 12 2016.

[19] Z. Huang, Z. Pan, and B. Lei, "Transfer learning with deep convolutional neural network for SAR target classification with limited labeled data," *Remote Sensing*, vol. 9, p. 907, 08 2017.

[20] Y. Gao and K. Mosalam, "Deep transfer learning for image-based structural damage recognition," *Computer-Aided Civil and Infrastructure Engineering*, 04 2018.

[21] M. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey," *Journal of Machine Learning Research*, vol. 10, pp. 1633–1685, 07 2009.

[22] S.-H. Son, H.-Y. Lee, G.-H. Nam, and S.-S. Kang, "Korean song-lyrics generation by deep learning," *ICIIT '19: Proceedings of the 2019 4th International Conference on Intelligent Information Technology*, pp. 96–100, 02 2019.

[23] R. Lebret, D. Grangier, and M. Auli, "Neural text generation from structured data with application to the biography domain," *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1203–1213, 01 2016.

[24] L. Danlos, "The linguistic basis of text generation," *EACL '87: Proceedings of the third conference on European chapter of the Association for Computational Linguistics*, pp. 1–1, 01 1987.

[25] H. Hirjee and D. Brown, "Automatic detection of internal and imperfect rhymes in rap lyrics," *Proceedings of the 10th International Society for Music Information Retrieval Conference, ISMIR 2009*, pp. 711–716, 01 2009.

[26] J. Lu, "An evaluation of generated lyrics," Master's thesis, San Jose State University, 2020.