LYRIC AND MUSIC GENERATION USING DEEP LEARNING

A PROJECT REPORT

Submitted by

HARINI SRI REKA J 2015115015

NIVETHIKA R 2015115034

AISHWARYA S 2015115122

submitted to the faculty of

INFORMATION SCIENCE AND TECHNOLOGY

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY

COLLEGE OF ENGINEERING, GUINDY,ANNA UNIVERSITY

CHENNAI 600 025

APRIL 2019

ANNA UNIVERSITY
CHENNAI - 600 025
BONAFIDE CERTIFICATE

Certified that this project report titled LYRIC AND MUSIC GENERATION USING DEEP LEARNING is the bonafide work of Harini Sri Reka J (2015115015), Nivethika R (2015115034), Aishwarya S (2015115122) who carried out project work under my supervision. Certified further that to the best of my knowledge and belief, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or an award was conferred on an earlier occasion on this or any other candidate.

PLACE: CHENNAI	**Dr. SASWATI MUKHERJEE**
DATE:	**PROFESSOR**
	PROJECT GUIDE
	DEPARTMENT OF IST, CEG
	ANNA UNIVERSITY
	CHENNAI 600025

COUNTERSIGNED

Dr. SASWATI MUKHERJEE
HEAD OF THE DEPARTMENT
DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY
COLLEGE OF ENGINEERING, GUINDY
ANNA UNIVERSITY
CHENNAI 600025

ABSTRACT

In this project, the lyrics and music are generated based on the given inputs (Datasets for training the model). Neural network models are a preferred method for developing statistical language models because they can use a distributed representation where different words with similar meanings have similar representation and also they can use a large context of recently observed words when making predictions. Building networks that are capable of learning from a given dataset can enable generating original content. A language model can predict the probability of the next word in the sequence, based on the words already observed in the sequence. The goal of this model is to generate lyrics of the chosen mood say (happy, sad, etc), by training itself based on the given datasets of lyrics of various artists, but not identical to existing lyrics. This is implemented using Long Short-Term Memory language model. The idea is to build a machine learning model that can take a sequence of words as an input and output a continuation of that sequence. For the second phase music is generated by using a deep learning model that would take data set as the generated lyrics and its melodic accompaniments as input and train itself. The output of this module would be midi file. The same Long Short Term Memory is used for music generation also. The music is also generated based on the emotion of the lyrics by training based on the dataset given to it.

திட்டப்பணிச் சுருக்கம்

இந்த திட்டத்தில், பாடல் மற்றும் இசை உள்ளீடுகளின் (மாதிரியை பயிற்சி செய்வதற்கானக தரவுத்தளங்கள் உபயோகப்படுகின்றன) அடிப்படையில் உருவாக்கப்படுகின்றன. நரம்பியல் வலைப்பின்னல் மாதிரிகள் புள்ளியியல் மொழி மாதிரிகளை உருவொக்குவதற்கொன ஒரு விருப்பமொன முளறயொகும், ஏனெனில் அவை பல்வேறு சொற்கள் எங்கே விநியோகிக்கப்பட்ட பிரதிநிதித்துவத்தைப் பயன்படுத்தலாம். இதே போன்ற அர்த்தங்களைக் கொண்ட பல்வேறு சொற்கள் எங்கே பிரதிநிதித்துவம் வழங்கப்படுகின்றன. கொடுக்கப்பட்ட தரவுத்தளத்திலிருந்து அசல் உள்ளடக்கத்தை உருவாக்கும். ஒரு மொழி மாதிரி வார்த்தைகளின் அடிப்படையிலான அடுத்த சொற்களின் நிகழ்தகவை கணிக்க முடியும் ஏற்கனவே காட்சியில் காணலாம். இந்த மாதிரியின் குறிக்கோள் பாடல் உருவாக்குதல் ஆகும். தேர்ந்தெடுக்கப்பட்ட மனநிலையில் (மகிழ்ச்சி, சோகம், போன்றவை), கொடுக்கப்பட்ட அடிப்படையில் அடிப்படையாகக் கொண்டு பயிற்சி அளிக்கப்படுகிறது. பல்வேறு கலைஞர்களின் பாடல்களின் தரவுகள், ஆனால் தற்போதுள்ள பாடல் வரிகள் ஒத்ததாக இல்லை. இது நீண்ட குறுகிய கால நினைவக மொழி மாதிரியைப் பயன்படுத்தி செயல்படுத்தப்பட்டது. யோசனை தான் ஒரு இயந்திரம் கற்றல் மாதிரியை உருவாக்கவும், அது ஒரு உள்ளீடாக வார்த்தைகளின் வரிசையை எடுக்கவும் முடியும் வெளியீடு அந்த வரிசையின் தொடர்ச்சி. இரண்டாம் கட்ட இசை உருவாக்கப்படுகிறது. ஒரு ஆழமான கற்றல் மாதிரியைப் பயன்படுத்துவதன் மூலம் தரவு உருவாக்கப்பட்ட தொகுப்புகளை உருவாக்கும் மற்றும் இசை குறிப்பை உள்ளீடுகளாக கொண்டு தனக்கு பயிற்சி அளித்து கொள்ளும். மிடி கோப்பு வெளியீடாக இருக்கும். அதே நீண்ட குறுகிய கால நினைவகம் பயன்படுத்தப்படுகிறது. இசைத் தொகுப்பு உணர்வின் அடிப்படையிலும், கொடுக்கப்பட்ட தரவுத்தளத்தின் அடிப்படையிலும், பயிற்சி மூலம் உருவாக்கப்பட்டிருக்கிறது.

ACKNOWLEDGEMENT

First and foremost, we would like to express our deep sense of gratitude to our guide **Dr. SASWATI MUKHERJEE**, Head of the department, Department of Information Science and Technology, counsel, continuous support and patience. She has helped us to come up with this topic and guided us in the development of this project. She gave us moral support and freedom to finish our project.

We are thankful to the project committee members **Dr. K. VANI** , Professor **Dr. P. YOGESH**, Professor, **Dr. S. SENDHIL KUMAR**, Associate Professor, **Dr. N. THANGARAJ** , Assistant Professor, **Mr. B. R. YUVARAJ**, Teaching Fellow, Department of Information Science and Technology, Anna University, Chennai for their valuable guidance and technical support.

We express our heartiest thanks to all other teaching and non teaching staff those who have helped us in one way or other for the successful completion of the project. Last but not the least we would like to thank our parents, friends for their indirect contribution to the successful completion of my project.

HARINI SRI REKA J
NIVETHIKA R
AISHWARYA S

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
GRU	Gated Recurrent Unit
HTML	HyperText Markup Language
LSTM	Long Short Term Memory
MIDI	Musical Instrument Digital Interface
MSE	Mean Square Error
RNN	Recurrent Neural Network

CHAPTER 1

INTRODUCTION

One major area of human creativity involves the production of lyrics and music. Creation of songs, combinations of music and lyrics, is a challenging task for computational creativity. Obviously, song writing requires creative skills in two different areas: composition of music and writing of lyrics. A crucial challenge in computational song writing is to produce coherent, matching pair of music and lyrics. Can AI be used to write music and lyrics? The idea is to build a machine learning model that can take a sequence of words as an input and output a continuation of that sequence. To achieve this goal we need to build a suitable model and find some training data. Neural network models are a preferred method for developing statistical language models because they can use a distributed representation where different words with similar meanings have similar representation and because they can use a large context of recently observed words when making predictions. Building networks that are capable of learning from a given dataset can enable generating original content. The goal of this model is to generate lyrics of a chosen mood (happy, sad, etc.,) by training itself based on the given datasets of lyrics of various artists. These new lyrics must not be identical to existing lyrics. This project is implemented using Long Short-Term Memory model. The idea is to build a machine learning model that can take a sequence of words as an input and output a meaningful lyrics and further another LSTM module is used to generate music with generated the corresponding music with the generated lyric as input. The dataset for this model is a file containing lyrics and its corresponding musical accompaniment. The music generated is also based on emotion of the given lyrics and the corresponding dataset is used.

1.1 LONG SHORT TERM MEMORY

For almost all of the sequence prediction problems, Long short Term Memory networks have been observed as most effective solution. LSTMs have an edge over conventional feed-forward neural networks and RNN in many ways. This is because of their property of selectively remembering patterns for long durations of time. With LSTMs, the information flows through a mechanism known as cell states. This way, LSTMs can selectively remember or forget things. The information at a particular cell state has three different dependencies. The dependencies are,

1. The previous cell state (i.e., the information that was present in the memory after the previous time step).
2. The previous hidden state (i.e., this is the same as the output of the previous cell).
3. The input at the current time step (i.e., the new information that is being fed in at that moment).

1.2 PROBLEM STATEMENT

The main focus of the project is to use deep learning to generate lyrics and the corresponding music to the generated lyrics automatically. This is done with the help of Long Short Term Memory. The LSTMs can be trained to generate lyric in a specific genre and a corresponding music to that generated lyrics.

1.3 ORGANIZATION OF THE REPORT

The project report is organized as follows:- Chapter 2 discusses the literature survey of existing systems and various methods used by various researchers. Chapter 3 discusses the working of various modules of the proposed system along with the overall system architecture. Chapter 4 discusses the implementation details of the proposed system along with the necessary algorithms. Chapter 5 concludes the report by summarizing the results and proposes possible enhancements that can be done in future.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING SYSTEMS

Recent work of Sutskever et al. [1] has shown the effectiveness of Recurrent Neural Networks (RNNs) for text generation. In their works, the authors used an RNN to create a language model at the character level. The models learn various grammatical and punctuation rules, such as opening and closing parentheses, plus learning a large vocabulary of English words at the character level.

Daza et al. [2] have developed a text generation for artistic purposes, such as poetry and lyrics, often using templates and constraints.

Hirjee and Brown [3] have developed a rhyme detection tool based on a probabilistic model that analyzes phoneme patterns in words. The model is trained on a set of lyrics that were manually annotated for rhyming words.

Malmi et al. [4] studied the generation of music and poetry. This has been separated in the field of computational creativity and there have been a few attempts to study the interaction of textual and musical features. Sometimes songs with other instruments besides piano are referred to as vocal chamber music and songs for voice and orchestra are called orchestral songs. Some attempts have also been made to compose musical accompaniments for text.

Monteith et al. [5] proposed the Automatic Generation of Melodic Accompaniments for Lyrics. This work creates a system which can automatically compose melodic accompaniments for any given text. For each given lyrics, it generates hundreds of possibilities for rhythms and pitches and evaluates these possibilities with a number of different metrics in order to select a final output.

Nayebi and Vitelli [6] proposed the Algorithmic Music Generation using RNN's. The paper compares the performance of two different types of recurrent neural networks for the task of algorithmic music generation, with audio waveforms as input. In particular, the focus is on RNNs that have a sophisticated gating mechanism, namely, the Long Short-Term Memory (LSTM) network and the Gated Recurrent Unit (GRU). The results indicate that the generated outputs of the LSTM network were significantly more usically plausible than those of the GRU.

Toivanen et al. [7] stated the Automatical Composition of Lyrical Songs. The paper addresses the task of automatically composing lyrical songs with the matching musical and lyrical features, and we present the first prototype. The proposed approach writes lyrics first and then composes music to match the lyrics. The crux is that the music composition subprocess has access to the internals of the lyrics writing subprocess, so the music can be composed to match the intentions and choices of lyrics writing, rather than just the surface of the lyrics.

2.2 OBSERVATION AND MOTIVATION

A major part of all future AI applications is building networks that are capable of learning from some dataset and then generating original content.

The premise of a Language Model is to learn how sentences are built in some body of text and use that knowledge to generate new content.

Neural networks have been applied extensively to various related tasks. For instance, RNNs have shown promise in predicting text sequences. Other applications include tasks such as information extraction, information retrieval and indexing.

A deep neural network model can be developed for capturing the semantic similarity of lines. This feature carries the most predictive power of all the features we have studied.

The major focus of this problem is motivated by two perspectives. First, the interest in analyzing the formal structure of the lyrics and in developing a model that can lead to generating artistic work.

In the same way the thought about generating music in accordance with the lyric generated. Neural network came handy in this domain also. The LSTM model is used to generate music based on the given lyrics.

CHAPTER 3

DESIGN

This chapter gives the detailed design and understanding about the project.

3.1 LONG SHORT TERM MEMORY'S GATES

The key to LSTMs is the cell state, the horizontal line running through the top of the Figure 3.3. The cell state is kind of like a conveyor belt. It runs straight down the entire chain, with only some minor linear interactions. Its very easy for information to just flow along it unchanged. The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates. Gates are a way to optionally let information through. They are made up of a sigmoid neural net layer and a pointwise multiplication operation. The sigmoid layer outputs numbers between zero and one, describing how much of each component should be let through. A value of zero means let nothing through, while a value of one means let everything through! An LSTM has three of these gates, to protect and control the cell state. The first step in LSTM is to decide what information is to be thrown away from the cell state. This decision is made by a sigmoid layer called the forget gate layer. It looks at ht1 and xt, and outputs a number between 0 and 1 for each number in the cell state Ct1. A 1 represents completely keep this while a 0 represents completely get rid of this. The next step is to decide what new information were going to store in the cell state. This has two parts. First, a sigmoid layer called the input gate layer decides which values well update. Next, a tanh layer creates a vector of new candidate values, C t, that could be added to the state. In the next step, well combine these two to create an update to the state. Finally, what is to be outputed is decided. This output will be based on the cell state, but will be a filtered version. First, a sigmoid layer is ran which decides what parts of the cell state is to be outputed. Then, we put the cell state through tanh (to push the values to be between 1 and 1) and multiply it by the output of the sigmoid gate, so that the parts that is decided is outputed.

3.2 LYRIC GENERATION: CHARACTER LEVEL MODEL

A language model predicts the next word in the sequence based on specific words that have come before it in the sequence.The benefit of character based language models is their small vocabulary and flexibility in handling any words, punctuation, and other document structure. In the case of a character based language model as mentioned in Figure 3.1, the input and output sequences must be characters. The number of characters used as input will also define the number of characters that will need to be provided to the model in order to elicit the first predicted character. After the first character has been generated, it can be appended to the input sequence and used as input for the model to generate the next character. The RNN character-level language models were trained. That is, the RNN was given a huge chunk of text and asked it to model the probability distribution of the next character in the sequence given a sequence of previous characters. This will then allow the generation of new text one character at a time as shown in Figure 3.2. As a working example, suppose there is only a vocabulary of four possible letters ”helo”, and is to be trained on an RNN on the training sequence ”hello”. This training sequence is in fact a source of 4 separate training examples: 1. The probability of ”e” should be likely given the context of ”h”, 2. ”l” should be likely in the context of ”he”, 3.”l” should also be likely given the context of ”hel”, and finally 4. ”o” should be likely given the context of ”hell”. Concretely, each character is encoded into a vector using 1 hot encoding (i.e. all zero except for a single one at the index of the character in the vocabulary), and fed into the RNN one at a time. At test time, a character is fed into the LSTM and get a distribution over what characters are likely to come next. This distribution is sampled, and fed right back in to get the next letter. This process is repeated.

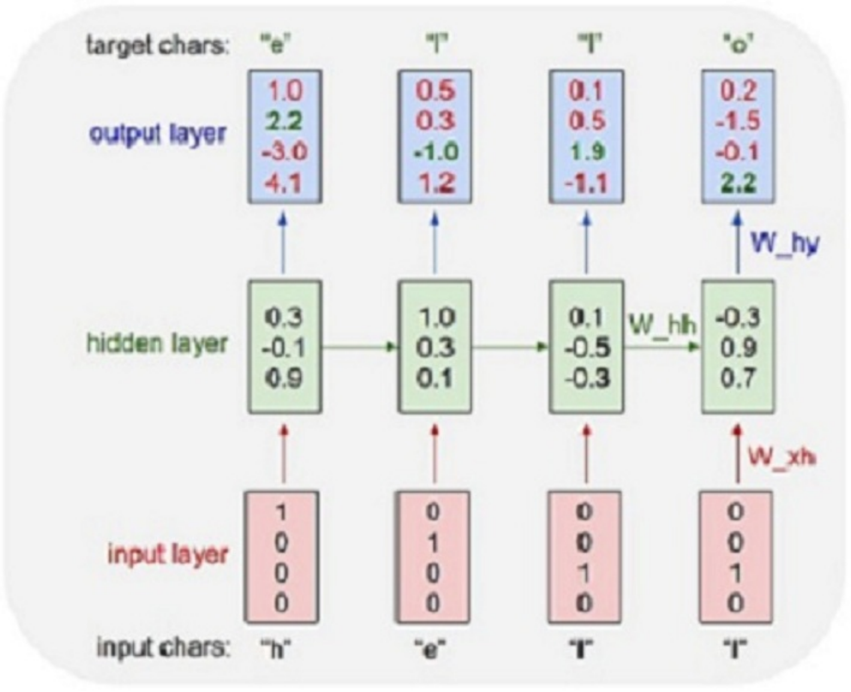
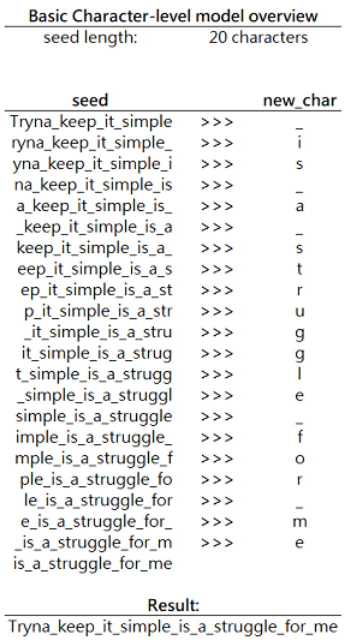


Figure 3.1: Character Level Model

LSTM model uses a specific architecture for hidden transformation defined by the LSTM memory cell. The key feature to the LSTM memory cell is the presence of an input gate, output gate, forget gate, and cell/cell memory, which manifest themselves in the model activation vectors. Each of these gates/cells has its own bias vector, and the hidden layer at each time-step is now a complex nonlinear combination of gate, cell, and hidden vectors. LSTMs are explicitly designed to avoid the long-term dependency problem. All recurrent neural networks have the form of a chain of repeating modules of neural network. The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates. Gates are a way to optionally let information through. They are composed out of a sigmoid neural net layer and a point wise multiplication operation.This lyrical output is given



Basic Character-level model overview

seed length: 20 characters

seed		new_char
Tryna_keep_it_simple	>>>	_
ryna_keep_it_simple_	>>>	i
yna_keep_it_simple_i	>>>	s
na_keep_it_simple_is	>>>	_
a_keep_it_simple_is_	>>>	a
_keep_it_simple_is_a	>>>	_
keep_it_simple_is_a_	>>>	s
eep_it_simple_is_a_s	>>>	t
ep_it_simple_is_a_st	>>>	r
p_it_simple_is_a_str	>>>	u
_it_simple_is_a_stru	>>>	g
it_simple_is_a_strug	>>>	g
t_simple_is_a_strugg	>>>	l
_simple_is_a_struggl	>>>	e
simple_is_a_struggle	>>>	_
imple_is_a_struggle_	>>>	f
mple_is_a_struggle_f	>>>	o
ple_is_a_struggle_fo	>>>	r
le_is_a_struggle_for	>>>	_
e_is_a_struggle_for_	>>>	m
_is_a_struggle_for_m	>>>	e
is_a_struggle_for_me		

Result:

Tryna_keep_it_simple_is_a_struggle_for_me

Figure 3.2: Character Level Model Overview

as input to generate music in the next phase.

3.3 MUSIC GENERATION

The flow of music generation phase using Character level model is shown in Figure 3.4. The Keras framework is used to build the model architecture. Each input note in the music file is used to predict the next note in the file, i.e., each LSTM cell takes the previous layer activation and the previous layers actual output as input at the current time step tt. The music model consists of a LSTM layer with 1024 hidden layers. ”Sparse Categorical Cross Entropy” is used as the loss function and ”adam” is used as the optimizer. When the number of classes is larger than 2 cross-entropy is used. Sparse Categorical Cross-entropy and multi-hot categorical cross-entropy use the same equation and should have the same output. The difference is both variants covers subset of usecases and the implementation can be different to speed up the calculation. When there is a single-label, multi-class classification problem, the labels are

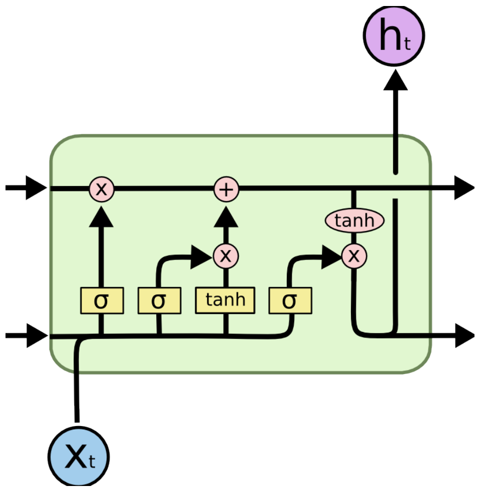


Figure 3.3: Long Short Term Memory

mutually exclusive for each data, meaning each data entry can only belong to one class. This saves memory when the label is sparse. Adam is an optimization algorithm that can be used instead of the classical stochastic gradient descent procedure to update network weights iteratively based on training data. The dataset consists of the lyric along with its musical notes (It will be in the form of words in the lyrics along with its musical notes eg. ”the” ”F”). The datasets are collected based on the emotions (happy, sad, etc.,). Finally the generated music is based on the given lyrics(Lyric generated from the previous phase) and its corresponding emotion. Henver found that tempo plays the largest and most consistent role of any musical parameter in carrying the expressiveness in music. More recent studies, some testing emotional effects of multiple musical parameters, have found that tempo affects the arousal dimension of emotion. Fast tempi are associated with high-arousal emotions, such as excitement, joy, and fear and slow tempi associates with low-arousal emotions such as sadness, peacefulness and boredom. Modulating the tempo enabled us to generate music relevant to the required emotion. The overall flow diagram is as shown in the Figure 3.5.

The generated lyrics from the first phase (i.e., Lyric generation phase) is

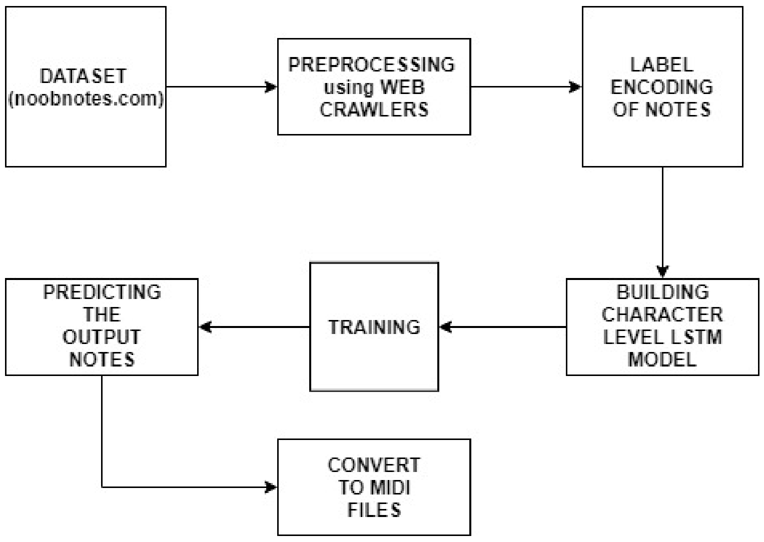

DATASET (noobnotes.com)
PREPROCESSING using WEB CRAWLERS
LABEL ENCODING OF NOTES
BUILDING CHARACTER LEVEL LSTM MODEL
TRAINING
PREDICTING THE OUTPUT NOTES
CONVERT TO MIDI FILES

Figure 3.4: Music Generation

used to generate music in the second phase. This implies that the musical accompaniments corresponding to the lyrics is used for generating music. The accompaniments used in this case are piano notes. The datasets of various songs and their corresponding piano notes are used to find the notes of the given lyrics.

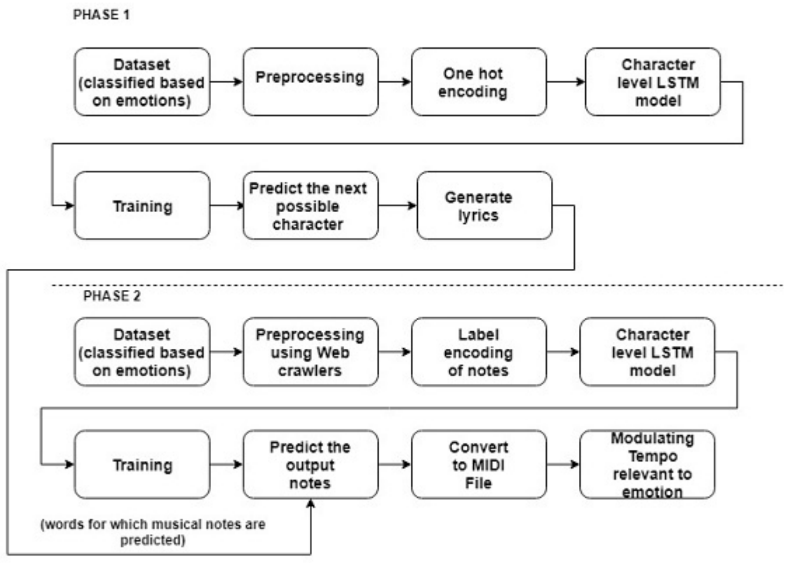

PHASE 1
Dataset (classified based on emotions)
Preprocessing
One hot encoding
Character level LSTM model
Training
Predict the next possible character
Generate lyrics
PHASE 2
Dataset (classified based on emotions)
Preprocessing using Web crawlers
Label encoding of notes
Character level LSTM model
Training
Predict the output notes
Convert to MIDI File
Modulating Tempo relevant to emotion
(words for which musical notes are predicted)

Figure 3.5: Flow Diagram for Lyric and Music Generation

CHAPTER 4

IMPLEMENTATION

This chapter explains the implementation details of the project which includes the softwares, packages and libraries used and the other implementation details.

4.1 LYRIC GENERATION

DATA COLLECTION

First, dataset of lyrics is needed for the computer to train! Before that, different genres should be selected .The selected genres were Childrens Music, Country, Jazz, Latin, Rap, Modern Religious (worship music),etc. Dataset was collected from All Music (https://www.allmusic.com/genres) to get the top artists for each genre. For each artist listed above, the link http://songmeanings.com/ was used to grab all songs listed for each artist. Grabbing the lyrics was not very hard, since the lyrics is stored in plain HTML. Python html library was used to parse the html to get the text, and put them all in one file.This process as repeated until every song of each artist was saved in a text file.

4.2 PREPROCESSING

The song directives like verse and chorus was removed for consistency. The punctuation and line breaks were left as it is before doing any further preprocessing. This process is repeated for all the songs.This will be the training data.

4.3 MODEL

Long short-term memory (LSTM) Character Level Word Modeling was used to train the lyric generation.

4.4 LSTM IMPLEMENTATION

For LSTM preprocessing, a dictionary of letters from the training data was created based on the conditions that letters are case sensitive, punctuation are treated as characters, and is treated as a character. Next, the character sequences was converted into a one-hot encoded array so that each character could be processed separately as a binary feature. Then, the input vectors was split into batches, to feed into LSTM step by step for the optimizer to minimize MSE by tuning the weights of the LSTM nodes. The optimizer was allowed to run (train the model) until the end of the file is reached. When this happens, an epoch is complete.

4.5 TRAINING THE MODEL

As the number of epochs increases, the variety in repeated words becomes greater. The training time greatly depends on your training data size.

The training can also be skipped by loading the trained weights. Pick a random sequence and make the network continue. Insert ’n’-chars long string as input seed text. Output lyrics is generated.

Algorithm 4.1 Algorithm for Generating Lyrics

```
    Input : Dataset with around 500 song lyrics.
    Output : Generated lyrics.
1:  Import the classes and functions.
2:  Read the corpus and get unique characters from the corpus.
3:  Create sequences that will be used as the input to the network.
4:  Create next chars array that will serve as the labels during the training.
5:  Vectorize the character array.
6:  Define the structure of the model.
7:  Train the model.
```

4.6 RESULT

Tuning models is very important for getting good results. There was a huge difference between the quality of the lyrics generated by our first LSTM model before tuning and the quality of the generated lyrics after tuning. Also, we need different preprocessing methods for each dataset.From our results, it looks like the modern worship lyrics are the easiest to generate, while jazz and rap lyrics are comparably more difficult generate using LSTM.
The output is as follows:

—– diversity: 0.2 —– Generating with seed: ”you are my love and my life, girl, your ” you are my love and my life, girl, your start the way the streets in the back i was in the man the back to the time to make a plans i'm a beat the streets to the truth the part in the struggle in the track in the stars stay was the car the streets of the track the back and so i got the sound i got the stronger the streets of the soul i got the mic still and the stay in the tark the streets to the taste the beat the back to the train the

—– diversity: 0.5 —– Generating with seed: ”you are my love and my life, girl, your ” you are my love and my life, girl, your change you can't want to be outta the way and feeling all the way the house in the track love you got the bigger that be have to make a sold in the crack she can't know why i be the time that afrain on the pain to care what you can are a grands in the man i'm a might was been on the back to be the way i said you got the some the way they got the brain i gotta be a man girl the back the tropped my

4.7 MUSIC GENERATION: DATA

Our dataset includes numerous number of lyrics and its corresponding musical accompaniment. The dataset will be of the form ”word musicalnotes”. eg., ”the” ”F”. The datasets are collected based on the emotion.

4.8 DATA PREPROCESSING

We will first have to gather datasets. The website https://noobnotes.net/ has the lyrics and their corresponding musical notes in letter notations. Web Crawlers were used to scrape the contents of the website. It is then processed into a csv file in a format suitable for processing namely word - note pair.

4.9 MODEL TRAINING

The model was trained using LSTM with 1024 hidden state. Character level modelling was used as language model. The main advantage of working with character-level generative models is that the discrete space you're working with is much smaller – there are about 97 English-language characters in common usage if we include all punctuation marks. By contrast, a vocabulary is many thousands of words. This implies that just storing the word embeddings will require a lot of memory, and including word embeddings in a model adds many,

many parameters to the model so the computational cost is much higher on this account than the character-level model. Then we use ”softmax” activation function. ”Sparse categorical cross entropy” is used as loss function. Since musical notes overlap each other categorical cross entropy loss function is used.

4.10 MUSIC GENERATION

To generate new notes the trained model was used to predict the next ’n’ notes based on the input lyrics. At each time step, the output of the previous layer is provided as input to the LSTM layer at the current time step t.

Algorithm 4.2 Algorithm for Music Generation

```
    Input : Dataset of 200 songs as word - note pair.
    Output : Midi file containing music based on the emotion.
1:  Import the classes and functions.
2:  Read the dataset.
3:  The notes in the dataset are used as labels during the training.
4:  Label encode the notes .
5:  Define the structure of the model.
6:  Train the model.
7:  Modify the tempo to get the emotion based on the given lyric's emotion.
```

4.11 RESULT

The generated MIDI file’s sheet music is as given in the Figure 4.1. A tool called ABC Player and Editor was used to convert the result of the music generation phase which is musical notes to sheet music and corresponding MIDI file. Various factors like the tempo, pitch, etc., can be modified accordingly.

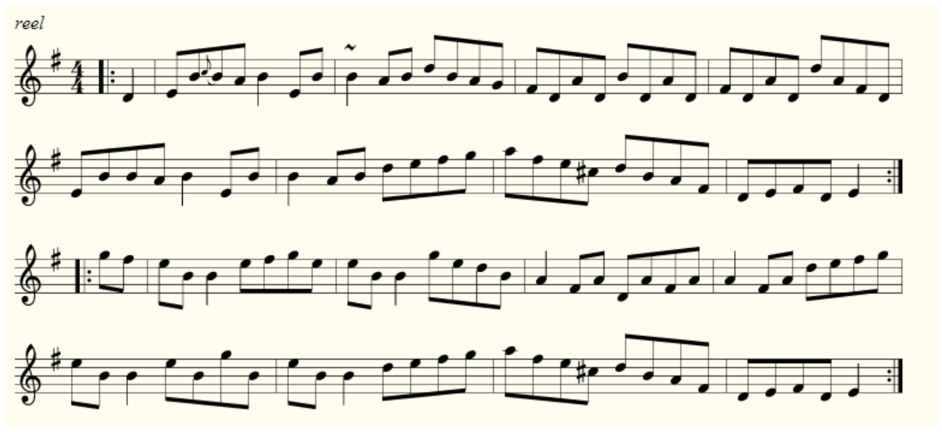


Figure 4.1: Sheet Music

4.12 PERFORMANCE ANALYSIS

The performance analyis for our model (i.e.,) the accuracy and loss is depicted in graphs. The accuracy of the model for 20 epochs is 54 percentage. This is represented in the Figure 4.2.

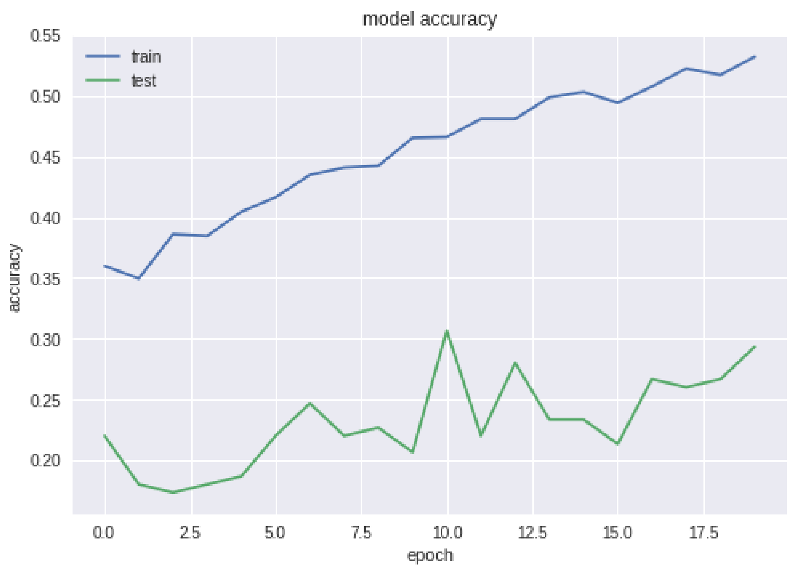


Figure 4.2: Accuracy For 20 Epochs

The loss for 20 epochs for both train and test data is depicted in the Figure 4.3.

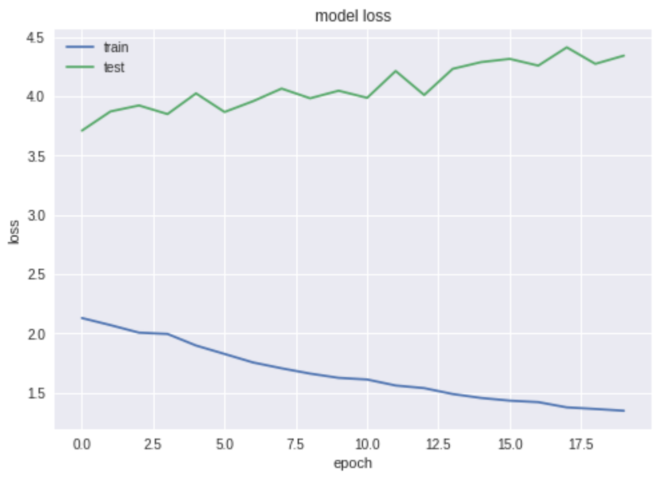


Figure 4.3: Loss For 20 Epochs

The model is now trained and tested for more number of epochs for better results. The accuracy of the model for 75 epochs is 60 percentage. This is represented in the Figure 4.4.

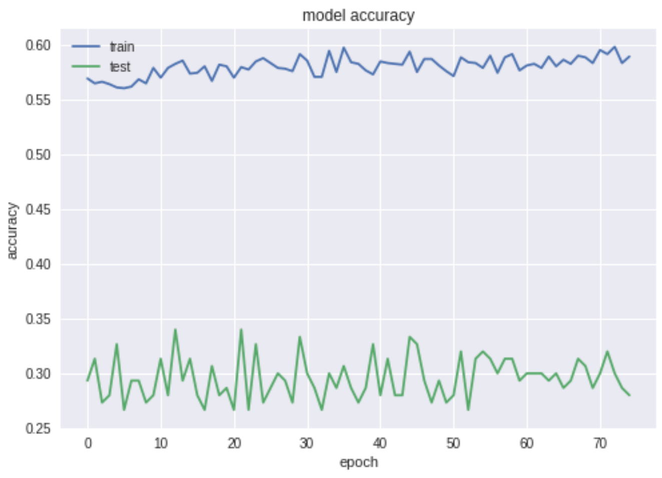


Figure 4.4: Accuracy For 75 Epochs

The loss for 75 epochs for both train and test data is depicted in the Figure 4.5.

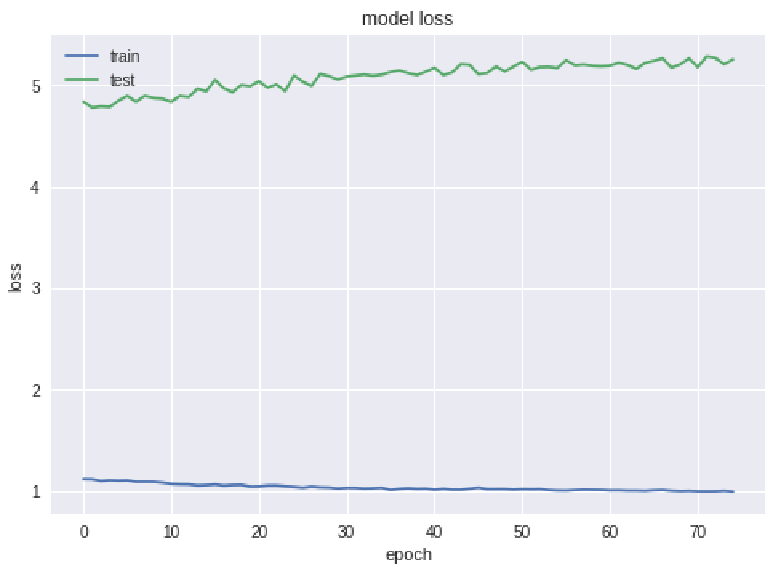


Figure 4.5: Loss For 75 Epochs

The results show that the proposed LSTM architecture is able to predict the notes for about 65 percentage accuracy for more number of epochs.

CHAPTER 5

CONCLUSION AND FUTURE WORKS

5.1 CONCLUSION

The task of generating lyrics and a corresponding music as a research topic of computational creativity has been proposed.As a first step towards a generative music-lyrics model a system that generates simple lyrical art songs and music based on a lyric-music note dataset was implemented. Thereby this documentation explains how to create a generative model for text, character-by-character using LSTM recurrent neural networks in Python with Keras. The lyrics are generated based on the given inputs. A set of new lyrics based on the given emotion will be output of this project.In this work, the effectiveness of an LSTM model for generating novel lyrics that are similar in style to a target artist was shown. The music is generated based on the LSTM model were the dataset is lyric-music pair and the generated lyric is given as input. The music will also be based on the emotion of the given lyric and the model's corresponding dataset.

5.2 FUTURE WORK

As of now music is generated for certain emotions. In future the work may be extended for music generation for jazz and rap lyrics since these two categories need to be worked on different musical instruments like piano, drums, flute, guitar, etc.

REFERENCES

[1] Ilya Sutskever, Oriol Vinyals and V.Le Quoc. "Sequence to Sequence Learning with Neural Networks". *In the proceedings of Computer-Supported Collaborative Learning*, pp. 31-50, 2014.

[2] Angel Daza, Hiram Calvo and Jesus Figueroa-Nazuno. "Automatic Text Generation by Learning from Literary Structures". *In the proceedings of the Workshop on Computational Linguistics for Literature*, pp. 9-19, 2016.

[3] Hussein Hirjee and G. Daniel Brown. "Using Automated Rhyme Detection to Characterize Rhyming Style in Rap Music". *In the proceedings of Music Information Retrieval Conference*, pp. 101-139, 2010.

[4] Eric Malmi, Pyry Takala, Hannu Toivonen, Tapani Raiko and Aristides Gionis. "DopeLearning: A Computational Approach to Rap Lyrics Generation". *In the Journal of ACM*, Vol. 5, No. 2, pp. 99-140, 2010.

[5] Kristine Monteith, Tony Martinez and Dan Ventura. "Automatic Generation of Melodic Accompaniments for Lyrics". *In the proceedings of Computational Creativity*, pp. 44-92, 2012.

[6] Aran Nayebi and Matt Vitelli. "GRUV: Algorithmic Music Generation using Recurrent Neural Networks". *In the Course Project for CS224D: Deep Learning for Natural Language Processing,Stanford University*, pp. 11-32, 2015.

[7] M. Jukka Toivanen, Hannu Toivonen and Alessandro Valitutti. "Automatical Composition of Lyrical Songs". *In the proceedings of International and Corporate Change*, pp. 401-557, 2013.