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 AND COMMUNICATION ENGINEERING

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

January 2019

ANNA UNIVERSITY

CHENNAI - 600 025

BONA FIDE CERTIFICATE

Certified that this project report titled LYRIC AND MUSIC GENERATION USING DEEP LEARNING is the bona fide work of Harini Sri Reka J, Nivethika R, Aishwarya S 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
DATE:10-01-2019

Dr. SASWATI MUKHERJEE
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 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.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.

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, continuos 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**, Associate Professor, **DR. S. BAMA**, Assistant Professor, **DR. N. THANGARAJ** and **MR. B.R. YUVARAJ**, Teaching Fellow, Department of Information Science and Technology, Anna University, Chennai, for their valuable guidance and technical support.

Harini Sri Reka J
Nivethika R
Aishwarya S

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

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 a coherent, matching pair of music and lyrics. Can we make AI write music and lyrics for us? 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 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. Unlike previous work, our model defines its own rhyme scheme, line length, and verse length and to generate music from the generated lyrics aong with its musica accompaniments. This 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 continuation of that sequence for lyric generation and another LSTM module is used to generate music with generated lyric as input. The dataset for this model is a file containing lyrics and its corresponding musical accompaniment.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING SYSTEMS

Recent work (Sutskever et al., 2011; Graves, 2013) has shown the effectiveness of Recurrent Neural Networks (RNNs) for text generation. In their works, the authors use 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.

Text generation for artistic purposes, such as poetry and lyrics, has also been explored, often using templates and constraints (Oliveira et al., 2014; Barbieri et al., 2012).

Hirjee and Brown (2010b) have developed a rhyme detection tool based on a probabilistic model (Hirjee and Brown, 2010a) that analyzes phoneme patterns in words. The model is trained on a set of lyrics that were manually annotated for rhyming words.

Generation of music and poetry have been studied separately in the field of computational creativity and there have been a few attempts to study the interaction of textual and musical features (Mihalcea and Strap1 Sometimes songs with other instruments besides piano are referred to as vocal chamber music and songs for voice and orchestra are called orchestral songs. parava 2012). Some attempts have also been made to compose musical accompaniments for text (Monteith et al. 2011; Monteith, Martinez, and Ventura 2012).

Automatic Generation of Melodic Accompaniments for Lyrics. This work describes a system that can automatically compose melodic accompaniments for any given text. For each given lyric, it generates hundreds of different possibilities for rhythms and pitches and evaluates these possibilities with a number of different metrics in order to select a final output.

GRUV: Algorithmic Music Generation using Recurrent Neural Networks The paper compares the performance of two different types of recurrent neural networks (RNNs) for the task of algorithmic music generation, with audio waveforms as input. In particular, we focus on RNNs that have a sophisticated gating mechanism, namely, the Long Short-Term Memory (LSTM) network and the recently introduced Gated Recurrent Unit (GRU). Our results indicate that the generated outputs of the LSTM network were significantly more usically plausible than those of the GRU.

Automatical Composition of Lyrical Songs The paper address the challenging task of automatically composing lyrical songs with 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 recently applied to various related tasks. For instance, recurrent neural networks (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.

Our interest in this problem is motivated by two different different perspectives. First, we are interested in analyzing the formal structure of the lyrics and in developing a model that can lead to generating artistic work.

In the same way we thought about generating music in accordance with the lyric generated. Neural network came handy in this domain also. The form of RNN, LSTM models where also used to generate music.

CHAPTER 3

DESIGN

3.1 LYRIC GENERATION: CHARACTER LEVEL MODEL

A language model predicts the next word in the sequence based on the 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, 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. Well train RNN character-level language models. That is, well give the RNN a huge chunk of text and ask it to model the probability distribution of the next character in the sequence given a sequence of previous characters. This will then allow us to generate new text one character at a time. As a working example, suppose we only had a vocabulary of four possible letters helo, and wanted to train 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, we will encode each character into a vector using 1-of-k encoding (i.e. all zero except for a single one at the index of the character in the vocabulary), and feed them into the RNN one at a time. At test time, we feed a character into the RNN and get a distribution over what characters are likely to come next. We sample from this distribution, and feed it right back in

to get the next letter. Repeat this process and youre sampling text.

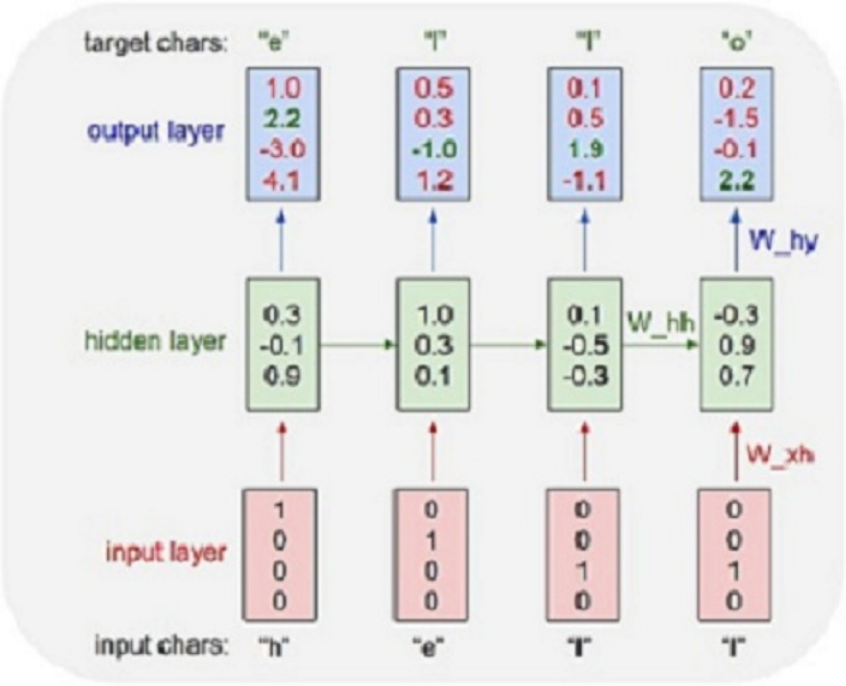

Figure 3.1: Character Level Model

The LSTM model uses a specific architecture for the 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 as input to generate music in the next phase.

3.2 MUSIC GENERATION

We will use keras to build our model architecture. We use a character level-based architecture to train the model. So 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 (at1) and the previous layers actual output (yt1) as input at the current time step tt. Our music model consists of a LSTM layer with1024 hidden layers. We use Sparse categorical cross entropy as the loss function and adam as the optimizer.The model is trained with the dataset(Lyrics along with its accompaniment) and predicts the musical notes for the given lyrics.

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

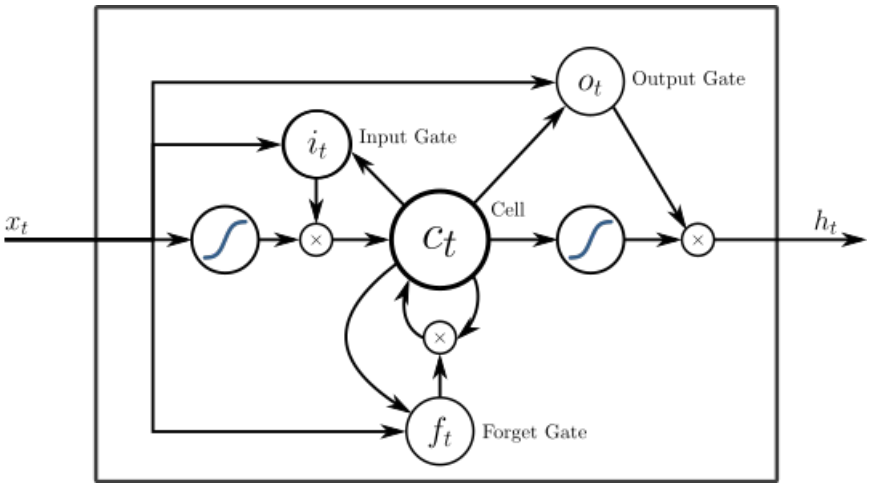

Figure 3.3: LSTM

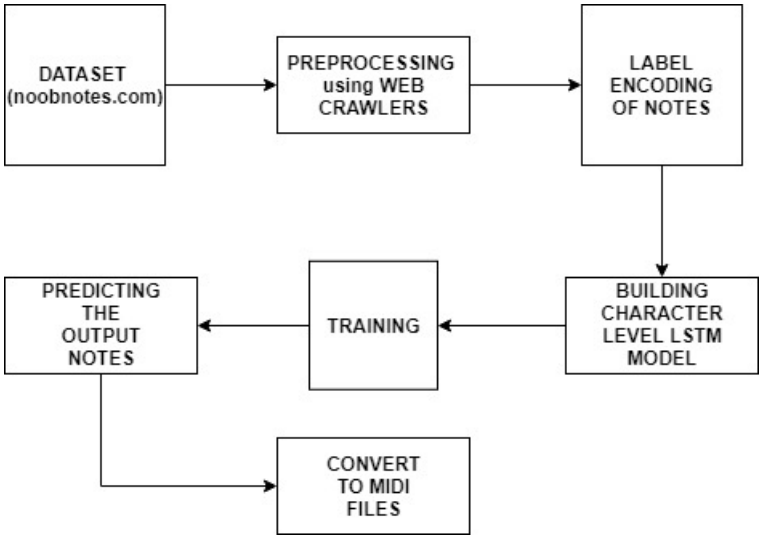

Figure 3.4: Music generation

CHAPTER 4

IMPLEMENTATION AND RESULTS

LYRIC GENERATION: DATA COLLECTION

First, we need to get lyrics for our computer to train! Before we do that, we need to select different genres to generate. We could pick whatever we want, but we decided to pick these genres: Childrens Music, Country, Jazz, Latin, Rap, Modern Religious (worship music),etc. we went on All Music (https://www.allmusic.com/genres) to get the top artists for each genre. For each artist listed above, we go to http://songmeanings.com/ to grab all songs listed for each artist.grabbing the lyrics was not very hard, since the lyrics is stored in plain HTML. We simply used the python html library to parse the html to get the text, and put them all in one file.We repeat this process until every song of each artist is saved in a text file.

PREPROCESSING

We removed the song directives, such as verse and chorus, as well as replacing the windows line endings (˚) with for consistency. We decided to leave the punctuation and s before doing any further preprocessing.We repeat this process for all the songs.This will be our training data.

MODEL

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

LSTM IMPLEMENTATION

For LSTM preprocessing, we began by creating a dictionary of letters from the training data, based on the conditions that letters are case sensitive, punctuation are treated as characters, and is treated as a character. Next, we converted the character sequences into a one-hot encoded array so that each character could be processed separately as a binary feature. Then, we split the input vectors into batches of size 2500 each, to feed into LSTM step by step for the optimizer to minimize MSE by tuning the weights of the LSTM nodes. We then continue to run the optimizer (train the model) until the end of the file is reached. When this happens, an epoch is complete.

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.

ALGORITHM

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

You can also skip training by loading the trained weights. Pick a random sequence and make the network continue. Insert your 'n'-chars long string as input seed text. Output lyrics is generated.

RESULT

We learned that 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.

INPUT:

MODEL PARAMETERS:

Optimizer: RMSprop

Learning rate: 0.01

Loss Function: categorical cross-entropy

Activation: softmax

SEQUENCE$_LENGTH = 40$

SEED TEXT : "you are my love and my life, girl, your "

OUTPUT:

—— 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

MUSIC GENERATION:DATA

Our data set includes numerous number of lyrics and its corresponding musical accompaniment.

DATA PREPROCESSING

We wiill 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.

MODEL TRAINING

We train the model using lstm model with 1024 hidden state.

MUSIC GENERATION

To generate new notes,we use the trained model to predict the next 'n' notes based on the input lyrics.

At each time step, the output of the previous layer (yt1) is provided as input (xt) to the LSTM layer at the current time step t.

ALGORITHM

Step 1: Import the classes and functions.

Step 2: Read the dataset

Step 3: The notes in the dataset are used as labels during the training.

Step 4: Label encode the notes .

Step 5: Define the structure of the model.

Step 6: Train the model.

INPUT: Dataset: Lyrics along with its musical accompaniments. Generated lyric will be given as input.

OUTPUT: Midi file

CHAPTER 5

CONCLUSION AND FUTURE WORKS

5.1 CONCLUSION

Thus we have proposed the task of generating lyrical songs as a research topic of computational creativity.As a first step towards a generative music-lyrics model we have implemented a system,generates simple lyrical art songs and music based on a lyric-music note dataset. 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, we have shown the effectiveness of an LSTM model for generating novel lyrics that are similar in style to a target artist. The music is generated based on the LSTM model were the dataset is lyric-music pair and the generated lyric is given as input.

5.2 FUTURE WORK

As of now we have got the musical notes as output. Finally the resulting song will be transformed to a music sheet and a midi file.The ultimate goal is to break the inherently sequential structure of the architecture, and to develop a song generation system with a much tighter integration or interaction between the lyrics writing and music composition processes.