**LYRIC AND MUSIC GENERATION USING LSTM NETWORK**

**Abstract**

In this paper, we propose a model to generate English lyrics and corresponding music based on the lyrics. The objective of the model is to generate lyrics based on a given mood that is new and not alike to pre-existing lyrics. The generated lyric is used to generate music using musical accompaniments for lyrics. To achieve this objective, this project depends on various Long Short Term Memory architectures. Using the model, unconstrained lyric is generated in the process of lyric generation and the musical structure (notes) are used to assist in the learning process of music generation.

**Introduction**

Musical compositiondefines a novel work of music, either vocal or instrumental. People who create new compositions are called composers. Composers of song lyrics are called songwriters or lyricists. In many cultures, the act of composing typically includes the creation of music notation, such as a sheet music score, which is then performed by the composer or by other instrumental musicians or singers. Composing songs and writing creative lyrics for the songs are challenging tasks. Even the most practiced and skilled songwriters and composers go through writer's block at some point in their career.

The goal of this project is to solve this problem by producing coherent pair of matching pair of lyrics and music based on a given mood.

The contribution of the project is in two-fold: (1) Generating lyrics that are not identical to existing lyrics (2) Generating music based on the generated lyric.

**Related Work**

The 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 character level RNN is used to create a language model. 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 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.

In regards to music generation, 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. 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 musically 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.

**LSTM**

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).

The foundation of an LSTM is a word embedding E that provides a vector representation for each of the words in our corpus. Given a history of words ** if the value to be determined is  where,  is a set of parameters used by the LSTM model. In the context of an RNN we define this probability by:



At each time-step the RNN computes f given an observation x and a previous state s. The input goes through a transformation where it passes through one or several hidden layers.

**Experimental Design**

**Dataset**

The dataset for lyric generation consists of song lyrics that are categorized by mood. The moods that were taken into account were happy, sad and romantic. The datasets were scraped from <http://songmeanings.com/>. After scraping the lyrics of songs from various genres into corresponding UTF-8 text files, lyrics were cleaned. The song directives, such as “verse” and “chorus”, as well as replacing the windows line endings (\r\n) with \n for consistency were removed. The punctuation and upper cased letters were excluded in the pre-processing.

The dataset for music generation phase consists of words and corresponding piano notes from various songs. The songs and notes were scraped from <https://noobnotes.net/>.

**Lyric Generation - Implementation**

We used a Python implementation of an LSTM that is built on top of Keras to build a character level language 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, 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 training data consists of mood categorized datasets of three moods. The datasets are trained separately. For training, 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 were 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.

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. The output lyric is generated by picking a random sequence and making the network continue. An ’n’-chars long string is inserted as input seed text.

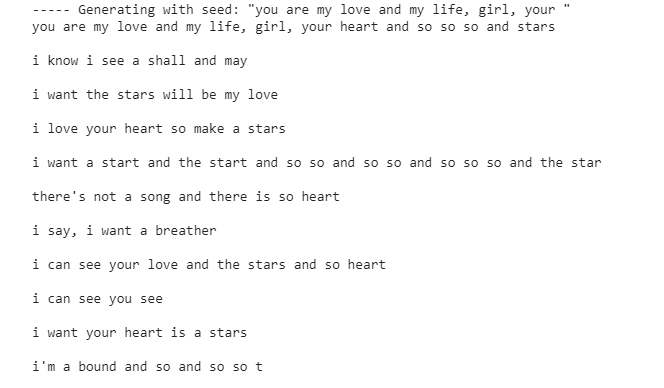
**Music Generation – Implementation**

A similar character level language model is used for the music generation process. The LSTM architecture is modified to process the input of word-note pair dataset. For a given input lyrics, the output generated will be a musical note for each word of the lyric. The notes are then converted into sheet music score and MIDI file using a converter called ABC Player and Editor (<http://www.clivew.com/abc.php>).

**Results**

**Example of Generated Lyrics**

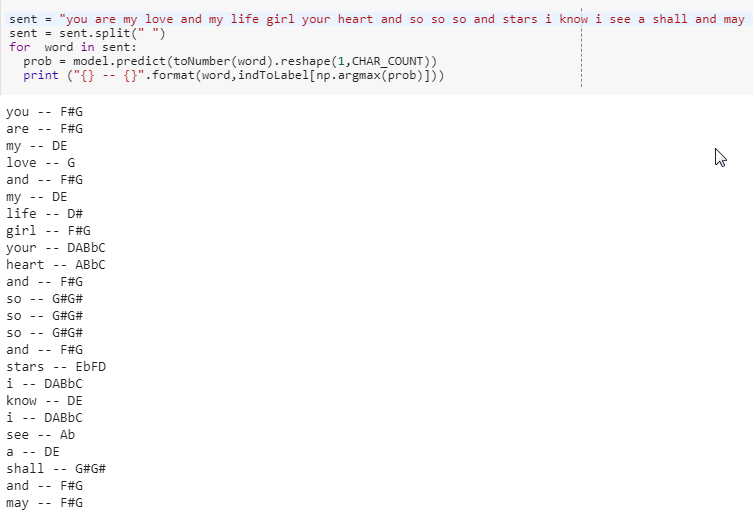
Given below is the sample of lyrics in the romantic mood generated by our model

****

The obtained output more or less obeys the grammar rules and contains words that rhyme.

**Sample of Generated Music notes**

Given below is a sample of generated music notes for the obtained output lyrics:



The sheet music score for the generated notes obtained is:



**Conclusion**

In this work, the model of generating lyrics and a corresponding music 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 paper 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 is not identical to existing lyrics is shown. The music is generated based on the LSTM model where 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.

**References**

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." *Advances in neural information processing systems*. 2014.

Daza, Angel, Hiram Calvo, and Jesús Figueroa-Nazuno. "Automatic text generation by learning from literary structures." *Proceedings of the Fifth Workshop on Computational Linguistics for Literature*. 2016.

Hirjee, Hussein, and Daniel Brown. "Using automated rhyme detection to characterize rhyming style in rap music." (2010).

Malmi, Eric, et al. "Dopelearning: A computational approach to rap lyrics generation." *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016.

Monteith, Kristine, Tony R. Martinez, and Dan Ventura. "Automatic Generation of Melodic Accompaniments for Lyrics." *ICCC*. 2012.

Nayebi, Aran, and Matt Vitelli. "GRUV: Algorithmic music generation using recurrent neural networks." *Course CS224D: Deep Learning for Natural Language Processing (Stanford)*(2015).

Toivanen, Jukka, Hannu Toivonen, and Alessandro Valitutti. "Automatical composition of lyrical songs." *The Fourth International Conference on Computational Creativity*. 2013.