Paper ID: 554

Automated Generation of Sinhala Lyrics using Recurrent Neural Networks

Sheng Kao# and Kalani Ilmini

Department of Computer Science, Faculty of Computing, General Sir John Kotelawala Defence University, Ratmalana, Sri Lanka

#fushenkao@gmail.com

Abstract: This paper discusses the approaches involved in implementing automated song lyrics system in the Sinhala language. Which includes an overview of the complexity of writing song lyrics and develop an automated application for Sinhala song lyrics generation. Before the implementation was carried out, a set of Sinhala song lyrics has been collected to create a corpus, and it has been used to develop an RNN model with LSTM layers using different temperatures and epochs. Then the created models were used to carry out a comparison process to evaluate the effect of the corpus size and the number of epochs per model training to get a better understanding of the RNN training behaviors. Finally, the system was served to a web host to give the user a friendly UI, where the user can enter desired keywords and generate new Sinhala song lyrics. The initial results were obtained through different models and we could see that with the increment of the number of epochs and the number of song lyrics that are trained in each model, the generated output had a clear growth in terms of accuracy and meaning of the song.

Keywords: Recurrent Neural Networks, Deep Learning, Lyrics Generation, LSTM

Introduction

From the very dawn of the human race, there has been one of the oldest arts that ever existed, which is, Music. It is a universal language that anyone can understand regardless of age, religion, gender, nationality, language, race, or skin color. Despite the booming entertainment industry today, for many people music production and songwriting have become professionalism. Currently, the singers in the music industry have faced an issue where they have to release songs at a quick rate to keep up in the industry since it has become more demanding than ever before. And together with music creation comes songwriting. Composing lyrics has consistently been a provoking assignment to new musicians or individuals who have the enthusiasm to get compose their self-written lyrics as a music lover, as it includes imagination as well as motivation. Numerous forthcoming lyricists and individuals who are energetic about songwriting additionally need a launch or a motivation, to begin with. The system that is supposed to be built will be a helping hand to this problem, which could become a very helpful resource for musicians and especially for the songwriters. Also, nowadays the lyrics of most of Sinhala songs have been very poor in value and in quality too.

One of the main issues of building a generating system in the Sinhala language is dealing with morphology, syntax, and semantics. Because it is much more complex compared to the English language. (Especially with **ස්වර පිල්ලම්**, **සන්ධි** in Sinhala) Also, make them correct and accurate to generate meaningful sentences or phrases. This automated lyrics generating system will be some assistance to this issue, which will be the possibility to turn into an exceptionally compelling system for lyricists and performers. Also finding a large corpus of Sinhala song lyrics is impossible while most English song corpora are readily available to download in an instant. This becomes a subproblem when digging into this project.

So, the system is supposed to be built using Recurrent

Neural Networks (RNN) with the aid of Long-Short-TermMemory (LSTM) layers. This creation of Sinhala lyrics corpus will be the first task to complete. The user will be enabled to enter a few keywords as his desire according to the topic the person wants to generate lyrics, then the system is supposed to generate lines of Sinhala lyrics according to the keywords given. It can be further developed to let the user choose the desired type of lyricist and generate lyrics according to the selected lyricist.

Many lyrics generating systems have been tried to create using LSTM and some of them are successful. But all of them are either in English, Chinese or some type of a widely used/popular language. But this research is going to be in Sinhala language and it's in a completely different grammar structure compared to the existing system languages' grammar structure. So, this will be a research conducted to test whether the use of LSTM networks can perform to generate lyrics in a complex language like Sinhala as well.

The rest of the paper is organized as follows, section II includes literature review, sections III includes technologies, section IV describes the implementation, section v includes results and evaluation and finally section VI discusses the conclusion of the project.

Literature Review

A. Lyrics generator with the use of rhythm

This system by Hugo R. and others (Oliveira, Cardoso and Pereira, 2007) was named as Tra-la-la-lyrics that is intended according to a specific pre-generated melody for the generation of lyrics. They researched the relationship between phrases, musical beats of melodies and rhymes. They introduced numerous algorithms by using Java that divided the syllables and it could extract the ' syllabic stress ' on a specific word. They also developed an SQL database that was intended to store words and grammar type. As they mentioned they could not achieve the results as they expected.

B. Generating rap lyrics

By Nguyen and Sa, a Rap Lyrics Generator was developed (Nguyen and Brian Sa, 2009). The software consisted of a collection of about forty thousand recorded rap lyrics. The words and verses were used in existing lyrics then generate a new lyric. The lyrics were made using a linear-interpolated tri-gram model technique. The result has, however, been graded as lacking flow. Hence, they have shifted the model into a quad-gram model from a tri-gram model. They also introduced a corpus of rhyming words in a MySQL database, containing two separate sentences. They have built sentences that rhyme with each other in this way. Ultimately, All the sentences were brought together according to the song's form and style.

C. Automated generation of poems

This system which includes natural language generation (NLG) techniques is the first poem generating system using artificial intelligence where it was developed by, Hisar Maruli, and others (Maruli Manurung, Graeme and Thompson, 2000). They named the implemented system as WASP (Wishful Automatic Spanish Poet). The program collects user inputs and consumed them as seeds. They used a Stochastic Hillclimbing Model because the creative element of the generation of poetry seems perfectly suited to a method with a certain element of randomness. The system is based on a forward reasoning ruled-based system. Even though the sample outputs showed that the stochastic hillclimbing search model is effective in producing text, results obtained have been evaluated as weak and not very productive.

D. Determining Semantic similarity in song lyrics To identify and classify music automatically and to determine artistic similarity, Logan, and others experimented with the use of lyrics around 400 artists (Logan, Kositsky and Moreno, 2004). On the internet, song lyrics have been collected from different sources. Different techniques such as Probabilistic Latent Semantic Analysis (PLSA) and methods of clustering k-means were used to identify and extract content and semantics. The evaluation was performed by comparing the system to another audio system to test the similarities. Both PLSA and K-nearest clustering methods had their advantages as well as disadvantages. So they have suggested a mix of the both methods which will provide much better results and it was mentioned in their future works.

E. LyriCloud and Titular

Burr Settles (Settles, 2010) created two software tools named Titular and LyriCloud which built for songwriters and musicians intending to create an artificial intelligent lyric generating system. A set of words related to the input seeds were suggested by LyriCloud, while the semiautomatic generation of song titles were provided by the Titular system. A technique based on templates was employed with the aid of a database which contains the existing song titles. Rude, slang and insulting phrases have been stripped out, and words that feature in the database more often would most definitely enter the output song title generated. They obtained good results, but there was still no semantic sense in the titles and the readers didn't had much of an understanding of the output.

F. Lyrics generation with NLP

By using basic natural language processing tools, Mahedero, and others (Mahedero et al., 2005) analyzed song lyrics. Lyrics have experimented so that the languages were known, categorized according to different topics, and the structure extracted, and similarities were sought among them. Selected from various websites was a list of 500 lyrics English, German, Spanish, French and Italian were the languages included. They succeeded to achieve a 92% accuracy from the results gained. A Naive Bayes classification has been deployed. The Inverse Document Frequency (IDF) was used to calculate similarity in long with cos distance (Cosine value of the distance measured). The similarity was assessed. Relative to the other researches, it was concluded that it is easier to identify the languages by these techniques.

G. Lyrics Classifier based on rhyme and style Rudolf and others (Mayer, Neumayer and Rauber, 2008) experimented with the characteristics of lyrics using the rhyme and style of a song. They managed to use statistical features to process lyrics with the aid of a bunch of words with POS (Parts-of-speech) tagging. Basically, two words which can sound similar when spelled are called as a rhyme. This feature is often used in the end of a verse. They performed experiments for a test collection of 397 song lyrics with Naive Bayes, k-Nearest Neighbor classifiers, and SVMs with linear and polynomial kernels. By all three classifiers, SVM classifier results given the best result of all experiments.

H. Generating Sega lyrics automatically

In the Mauritian criollo language, Didoral and Pudaruth (Pudaruth, Bhaukaurally and Didorally, 2012) worked to develop a lyrical tool for generating Sega lyrics. The results were given to 63 people to identify whether the given 10 songs are either written by a man or a whether it is generated by a computer. Five of them were computer-generated out of these 10 songs. However, the lyrics generated were in a very high standard because about almost half (50% of the participants) were uncertain about whether the lyrics were already existing or computer-generated. The main fault of this work is that there is no information on how the phrases are actually created or generated in the implementation section.

I. Essay Generation for a given topic

Xiaocheng and others (Feng et al., 2018) have developed an essay generating system in the Chinese Language. They thought to test their progress towards artificial intelligence would be a compelling way if they managed to create a computer program to generate essays automatically. To overcome common natural language generation challenges, they develop a multi-topic-aware long short-term memory (MTA-LSTM) approach for essay generation. Since there was no huge dataset to be tested, they have constructed two datasets of essays written in Chinese. The length of the produced essay was found to not affect the quality of the essay, also the final outcome of the methods they used were not very strong. Concluded as duplication and selfcontradiction of phrases were the main issues in the system.

J. Generating lyrics with LSTM

Depending on a particular artist's style, a lyrics generating system was developed by Olga, and others (Vechtomova et al., no date). The system has used a variational autoencoder with artist embeddings. They created a corpus of lyrics by picking seven artists. And they are picked on the basis of selecting one from each genre type. They trained a Convolutional Neural Network (CNN) classifier to predict artists. It was based on MEL spectrograms of their song clips. The initial findings discovered by the spectrogram classifier indicated that there is a gain in implementing the embedding of artists with the depictions. The final results indicated that except for one case, the generated results were satisfying, and related to the artists' style most of the time.

K. Word-Embedded for Text Classifier

Using machine learning to automate text classification will make the whole process super-fast and efficient was the motivation for Ya-Chuan and Chia-Wei (Huang, 2016) to build a text classifier with word-embedded. They adopted the Convolution Neural Network (CNN) without pre-trained words embedded on IMDb movie review data and evaluate with the baseline Naive Bayes Classifier to prove that their CNN algorithm works. developed model focus on extracting the sentiment from the 20 Newsgroups dataset so they could find the accurate category of the input news. They concluded that Naive Bayes outperformed CNN in such a simple task, but when it moves to a more complex task, the generative model NB has decreased dramatically. They have mentioned using Recurrent Neural Network (RNN) would be the perfect way to use neural networks over such a sequence of data.

L. Text Generation using Recurrent Neural Networks

Since text generating is difficult to perform by using Hidden Markov Models and Markov Chains, Partiksha and Karun (Taneja and Verma, 2017) have decided to use Recurrent Neural Networks (RNN) with its variants LSTM and GRU to develop a language model so that can generate whole new text word by word automatically. They have trained different RNN in two stages. First is the training of simple RNN, LSTM, and GRU on different datasets and in the second stage sampling them is done to generate the output text. They have implemented 5 different datasets as input and trained all threw network according to them. Then 15 models gave been generated text as output. By the results, they concluded that the performance is based on the input text and GRU performed better as it generated more realistic text and training loss was smallest in GRU.

M. Automatic Generation of News comments Hai-Tao and others (Zheng et al., 2018) have developed a system to automatic generation of news comments since they found out that it was not studied well in Natural Language Generation field. This task included the contextual connection with real-world news. So, to accomplish this they have decided to use a Gated Attention Neural Network model (GANN) to generate news comments. Also, to produce feedback with various topics and related grades, they used some arbitrary samples of data. To different perplexity scores, they showed that GANN is outperforming current word generation methods. The evaluation was done by using several human resources and the results were hardly distinguishable with real-world news comments and the generated ones.

N. Generating Text with Multiplicative Recurrent Neural Networks

Hessian-free optimization have achieved to outperform the obstacles with RNN trainings regarding to the other optimization methods. So, Ilya and others (Sutskever, Martens, and Hinton, 2011) decided to use it solve challenging sequence problems. And they have decided to develop a system to generate text using the variant of RNN, which is Multiplicative Recurrent Neural Networks (MRNN). The RNNs were trained using the Hessian-Free optimizer (HF) by applying them to character-level language modeling tasks. And they have trained them for five straight days on eight high-end GPU's. From the results they have concluded in contrast, the MRNN's nonlinear dynamics enabled to extract higher-level "knowledge" from the text and have produced better results compared to previous systems.

Summarized comparison of the most relatable works to this research has been discussed in the following Table 1.

By comparing and contrasting the technologies and the results of the above mentioned related works, it can be decided that the most suited approach to this project would be to use Recurrent Neural Networks (RNN) over the Conventional Neural Networks (CNN) and also other methods. CNN is best fitted to classify text data and has given better results but lacked in quality of text generation. While RNN has performed far better than CNN and other methods to generate text even though the training time is higher and complex in implementing.

Table 1: Comparison of the Literature Review

Work	Aim	Technology	Result
Tra-la-lalyrics	Generate lyrics for given melodies	Used three strategy Algorithms in Java, MySQL as a database engine	Generated lyrics with some meaning from three strategies but, there were no explicit semantics in the output
Rap Lyrics Generator	Outputs a rap song based on the words from the corpus	Linearinterpolated Tri-gram Model & Quad-gram model with MySQL database	Produces sentences that are usually grammatically correct, but not meaningful enough
Computatio nal Creativity Tools for Songwriters	Create intelligent and interactive lyrics writing tools	Semantic networks and templatebased approach to title synthesis	Better results than previous generating tools but titles didn't have semantic meaning
Natural Language Processing of Lyrics	Experiment on the use of standard natural language processing (NLP) tools for the analysis of music lyrics	Ted Dunning' statistical identification algorithm to identify language and NLP techniques	The results made believe that NLP techniques can be successfully used for the creation of extensive ground truth metadata of lyrics
A SemiAutomated Lyrics Generation Tool for Mauritian Sega	Develop a lyrical tool for generating Sega lyrics (a major form of art of the Mascarene islands)	Implemented in PHP: and MySQL as a database engine	48.15 % of people thought that semigenerated lyrics were written by human beings
Generating lyrics with LSTM	Generating song lyrics lines depending on a particular artist's style	CNN classifier with LSTM and MEL spectrograms	Generated lyrics related to the artists' style most of the time.
Generating Text with Recurrent Neural Networks	Demonstrate the power of large RNNs trained to computergenerate text	Recurrent Neural Networks (RNNs) with the HessianFree optimizer (HF)	Represented the largest recurrent neural network application to the published date and produced surprisingly good language models

Technologies

We will explore some context information in this segment of the main technologies used to build the lyrics generating system.

A. Recurrent Neural Network (RNN)

Recurrent Neural Networks have shown promising results in the field of AI especially in machine translation tasks (Liu et al., no date). In comparison to feedforward neural networks, RNNs are capable of controlling the feedback of a Sequence of variable length by having a recurrent hidden state in which each activation depends on the previous period. The below diagram describes an RNN, which is being unfolded to a full network. If we take an example of a sentence of four terms, the network will be unrolled into a neural network of four layers.

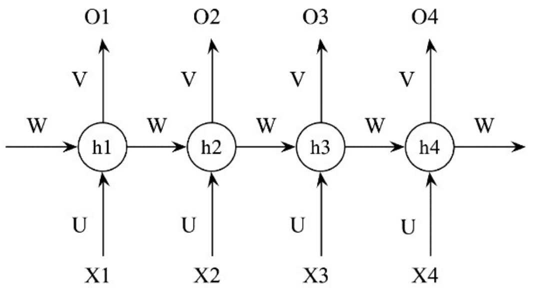

Figure 1: An unrolled RNN

The formulations in an RNN which govern the calculations are as follows.

Let x_t be the input at time step t. For example, x_i can be a one-hot vector or word embeddings corresponding to the ith word of a sentence.

h_t be the hidden state and the "memory" of the network at time step t. h_t is calculated according to the previous hidden state and the input at the current step:

$$h_t = (Ux_t + Wh_{t-1})$$

Here, to calculate the first hidden state, h_0 is usually initialized to a vector zero.

• o_t will be the output at each time step t. For sentiment classification of short texts, it would be a vector of probabilities across all the sentiment categories. o_t will be calculated as follows:

$$o_t = softmax(Vh_t)$$

B. Long Short Term Memory (LSTM) LSTM is a special kind of RNN. They are capable of learning long-term dependencies (Understanding LSTM Networks -- colah's blog, no date). LSTM is initially introduced by Hochreiter & Schmidhuber in 1997, and several men were mastered and popularized in subsequent research. They work immensely well on a wide range of problems and are now broadly used.

All recurrent neural networks take the shape of a series of repeating neural network modules. In both standards, RNN's and LSTM's they have a chain-like structure, but the repeating module in LSTM has a different structure compared to the RNN's simple structure. That is, it has 4 layers that interact in a typical way instead of having a single neural network layer.

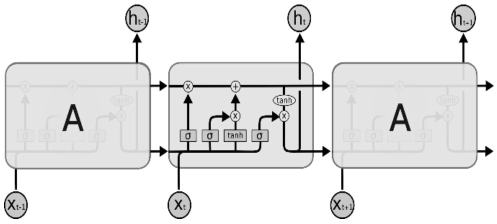

Figure 2: The repeating module of an LSTM

The learning phase depends on the magnitude of weights in the transition matrix. If the weights of the matrix are small this means that the gradients are called vanishing. But, if the transition matrix weights are large, it makes learning to get diverged. So, a typical LSTM model has a memory cell where it solves the issues of vanishing and diverging gradients.

C. Flask web framework

Flask is a Python web framework designed with a small core and in an easy-to-extend philosophy (Flask, no date). As the equivalent Flask web program becomes more transparent in common cases, Flask becomes considered more of a Pythonic software platform than other approaches. By default, Flask doesn't have a database abstraction layer, type validation, or anything else because there are already specific libraries that would handle them. Conversely, Flask allows plugins to apply to the program certain features as if it were implemented in Flask itself. This web framework work is used to connect the web application with the python coded program in this project as it will be the linker of the web UI.

D. Nvidia CUDA toolkit

CUDA (Compute Unified Device Architecture) is developed by Nvidia for a parallel computing platform and it is an application programming interface (API) model where it allows software developers to use CUDA-enabled graphics processing unit (GPU) for the general-purpose data processing ('CUDA', 2020). The CUDA interface is a software framework that provides direct access to the virtual instruction set of the GPU and parallel computing elements for computer kernel execution. Some of the advantages of using a CUDA enabled GPU for the training process are, it can scatter read the code from arbitrary addresses in memory, enables faster downloads and readbacks to and from the GPU and exposes a fast shared memory region that can be shared among threads.

Implementation

This system has been implemented in three main stages; named the creation of Sinhala lyrics Corpora and the Extraction & scraping stage, the Model training stage, and the Web serving stage. In this section, this paper will be discussing the implementation of the system based on these three stages.

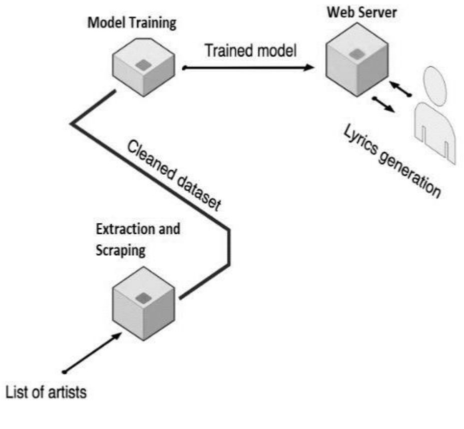

Figure 3: The big picture

A. Extraction & scraping stage

First, the creation of Sinhala lyrics corpora is done by collecting various song lyrics from different sources and categorizing them according to the lyricist and also another full dataset with all the lyrics combining without any specification of the lyricist type for the sake of different testing and evaluations. Since there is no such Sinhala song lyrics dataset/corpus available to be directly downloaded on the internet, creating a large set of such datasets is also a challenging task. Then the collected lyrics are scraped manually to eliminate the use of repetitions, signs, ellipses, and explanation abbreviations such as chorus, verse, etc.

B. Model training stage

Then the scraped cleaned dataset of lyrics are fed into the system for the creation of the training

RNN model which will be the core behind generating new lyrics. The models are trained with different parameters and temperatures to obtain a various set of outputs to compare and get a detailed idea about the accuracy and success of the generating output. Many neural network implementations are available online currently. But it is intended to choose one and fine-tuned implementation to fit the need: textgenrnn - a python project for text-generation using neural network. For this model training process, the CUDA enabled GPU acceleration has been used to obtain a faster training time compared to ordinary CPU training.

C. Web serving stage

After training the model the testing is done and the generated model is served in the localhost. A separate web UI front-end has been developed to interact with the user for the ease of using the system, where it will allow a user to enter just a few keywords according to his/her preference then hit a button to generate lyrics and display the generated lyrics in the same webpage. To serve the application, the flask python framework is used, so it will be easy to fit the system into a web server to provide a user-friendly UI.

Results and Evaluation

Three basic models were created initially with different parameters and epochs and the same input of two keywords was given to each model to test the results of the generated lyrics output for the comparison. Two outputs from the same keyword were generated for this task.

Input keywords: "ඔබට මම" A. Model-1 results

Corpus size: 30 songs

No.of epochs: 10

Table 2: Model-1 results

Output 1	Output 2
ඔබට මම ආදරය මට නිමල දැවෙන යටින් මා මා මන් බැඳී රංජනා	ඔබට මම ආදර හද කී ෙැලි තෙ තෙ තෙ මා තනිෙම අහම්වබන් දෙව ේ

B. Model-2 results

Corpus size: 78 songs

No.of epochs: 25

Table 3: Model-2 results

Output 1	Output 2
ඔබට මම ආදරය පෑ දෙව ේ ඔබ උන්ු තැන හවේ තාමත් හරිම උණුසුමයි	ඔබට මම ආදරය දෑ ම ඇසිපිය වනාවහළනො දුර ඇත සුලවේද පාවී හමන්වන්

C. Model-3 results

Corpus size: 78 songs

No.of epochs: 50

Table 4: Model-3 results

Output 1	Output 2
ඔබට මම එහා දුක් ජීවිවත් ේවන්හවේ ගැුවන් හිනැවහනො ඔබ මා හඬනා දෙව ේ ඔබ මා යළි හමුෙන්වන් ඒ ෙවග ල ේ න තෙ නෑ	ඔබට මම එහා මද අඳුවේ ෙරළ පුරා මල් පැළඳි ලිවේ වහාඳ හිත ුවෙ තරම් තෙ වලාෙකින් ලබන්නද

These are some outputs generated from the system and we can evidently see that with the increment of the number of epochs and the number of song lyrics that are trained in every three models, the generated output has a clear growth in terms of accuracy and meaning. Model 3 which is built with 78 songs and trained with 50 epochs has produced far better results compared to model 1 where it has only 30 songs and trained with 10 epochs. Furthermore, we can see that with the increase in the number of lyrics trained, and the training number of epochs, the generating results will be more accurate and semantically meaningful.

The final system will be having a model with ten times the number of song lyrics currently it has with it and they will be trained with more epochs into different temperatures to generate better Sinhala lyrics. And a human evaluation will be conducted in the future to assess how well the generated lines are in terms of the lyrics is either an existing one, written by a lyricist or a generated one from the system.

Conclusion

Today, nearly every industry in the world uses somewhat of software to perform a specific task. The purpose of this research work was therefore to incorporate an innovative method that could aid the songwriters in their task. This system uses lyrics corpus with a deep RNN model to train them and, the user can enter his desire keywords to the system to generate new lines of Sinhala lyrics. The generated initial results were promising than expected and they were grammatically meaningful most of the time. By increasing the number of lyrics count in the corpus and performing many epochs in different temperatures while training will be resulting to improve the quality of the generated lyrics. In future work, the system will be developed by using different models categorized by each lyricist, so the user will be able to select his/her desired lyricist and generate new lines of lyrics according to the selected lyricist style.

And also, to explore other approaches to determine whether there are any better solutions to generate lyrics except RNN models.

Future Works

As for the future works, it is supposed to develop the models with more song lyrics to increase the size of data in the corpora to develop even more rich models. Also, we can perform a data augmentation process to increase the size of the corpora so it will be more of an efficient and time-saving way rather than increasing the entire corpora size manually. Then it is supposed to fine-tune the models by changing the temperatures in the core system. Also, we can try out the results by changing the architecture we currently initialized and then check the results to find whether we can implement a better way compared to the current architecture. Another improvement of the system will be to categorize the generating lyrics according to the type of the lyricist. This can be done by creating different models for each lyricist and train them to generate according to that lyricist type. Finally, it is supposed to create a complete web interface for the user to interact with the system in a pleasant and userfriendly way.

References

'CUDA' (2020) Wikipedia. Available at: https://en.wikipedia.org/w/index.php?title=CUDA&oldid=96178 3159 (Accessed: 9 July 2020).

Feng, X. et al. (2018) 'Topic-to-Essay Generation with Neural

Networks', in Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence. Twenty-Seventh

International Joint Conference on Artificial Intelligence {IJCAI18}, Stockholm, Sweden: International Joint Conferences on Artificial Intelligence Organization, pp. 4078–4084. doi: 10.24963/ijcai.2018/567.

Flask (2017). Available at: https://www.fullstackpython.com/flask.html (Accessed: 9 July 2020).

Huang, R. (2016) 'Convolutional Neural Networks and WordEmbedded for Text Classifier', p. 90.

Liu, S. et al. (2014) 'A Recursive Recurrent Neural Network for Statistical Machine Translation', p. 10.

Logan, B., Kositsky, A. and Moreno, P. (2004) 'Semantic analysis of song lyrics', in 2004 IEEE International Conference on Multimedia and Expo (ICME) (IEEE Cat. No.04TH8763). 2004 IEEE International Conference on Multimedia and Expo (ICME), Taipei, Taiwan: IEEE, pp. 827–830. doi: 10.1109/ICME.2004.1394328.

Mahedero, J. P. G. et al. (2005) 'Natural language processing of lyrics', in Proceedings of the 13th annual ACM international conference on Multimedia - MULTIMEDIA '05. the 13th annual ACM international conference, Hilton, Singapore: ACM Press, p. 475. doi: 10.1145/1101149.1101255.

Maruli Manurung, H., Graeme, R. and Thompson, H. (2000) 'Towards A Computational Model of Poetry Generation', p. 11.

Mayer, R., Neumayer, R. and Rauber, A. (2008) 'RHYME AND STYLE FEATURES FOR MUSICAL GENRE CLASSIFICATION BY SONG LYRICS', p. 6.

Nguyen, H. and Brian Sa (2009) 'Rap Lyric Generator', p. 9.

Oliveira, H. R. G., Cardoso, F. A. and Pereira, F. C. (2007) 'Tra-la-Lyrics: An approach to generate text based on rhythm', p. 9.

Pudaruth, S., Bhaukaurally, B. F. and Didorally, M. H. A. (2012) 'A Semi-Automated Lyrics Generation Tool for Mauritian Sega', IAES International Journal of Artificial Intelligence (IJ-AI), 1(4), pp. 201–213. doi: 10.11591/ijai.v1i4.801.

Settles, B. (2010) 'Computational creativity tools for songwriters', p. 9.

Sutskever, I., Martens, J. and Hinton, G. (2011) 'Generating Text with Recurrent Neural Networks', p. 8.

Taneja, P. and Verma, K. (2017) 'Text Generation using different Recurrent Neural Networks'.

Understanding LSTM Networks -- colah's blog (2015). Available at: https://colah.github.io/posts/2015-08Understanding-LSTMs/ (Accessed: 3 July 2020).

Vechtomova, O. et al. (2018) 'Generating lyrics with variational autoencoder and multi-modal artist embeddings', p. 5.

Zheng, H.-T. et al. (2018) 'Automatic Generation of News Comments Based on Gated Attention Neural Networks', IEEE Access, 6, pp. 702–710. doi: 10.1109/ACCESS.2017.2774839.

Acknowledgement

I would like to express my highest graduate towards my supervisor Ms. Kalani Sriya Ilmini for the help in writing this paper. Thank you for your guidance and the support given throughout this project and I am eternally grateful once again.

Author Biographies

Sheng Kao is an undergraduate student in the Computer Science department, faculty of computing at General Sir John Kotelawala Defence University, Ratmalana, Sri Lanka. And he will be graduating in 2021 with a BSc in Computer Science. Sheng has a strong interest in the field of Machine learning, Artificial Neural networks, and Computer Music.

Kalani Ilmini is a lecturer at General Sir John Kotelawala Defence University, Ratmalana, Sri Lanka. Her research interests include Deep Leaning, Artificial Neural Networks, Data mining and Affective Computing.