Department of Mathematics and Computer Science
Department of Education and Psychology
Freie University Berlin

NATURAL LANGUAGE PROCESSING
PROF. DR. ARTHUR M. JACOBS

Summer Semester 2021

GENERATING TEXT/LYRICS BASED ON CHARACTER-LEVEL LANGUAGE MODEL BY MULTI-LAYER RECURRENT NEURAL NETWORK

Md Golam Kibria Asif
5506220
asif.kibria@fu-berlin.de

Table of Contents

1 Abstract

Lyrics are the one of the most important creative creation of the human culture. It reflects the philosophy and time. We will try to generate lyrics after analysing available lyrics. This is a well known research area in the Natural Language Processing. There are several methods available to generate the lyrics. We will use character-level multi-layer RNN and LSTM.

2 Introduction

Lyrics are the most powerful factor that determines the quality of a song. Generally lyrics is a set of words consisting versus and choruses. The meaning of those words might be abstract or direct. Lyrics writer generally writes lyrics and composer put the rhythm and life to those words. Other than being creative one can write lyrics using different machine learning techniques. For example, Dopeleraning [1] is a approach to predict the next line of existing lyrics based on RankSVM algorithm [2] and deep neural network model with novel structures. In our project we will train character level language modes based on Recurrent Neural Network (RNN) [3] and Long Short-Term Memory (LSTM) [4]. In other words, we will train a large set of texts to generate similar type of text one character at a time. Our goals of this project are:

- Select and prepare a usable dataset of existing song lyrics
- Select an existing module that implements multi-layer Recurrent Neural Network (RNN, LSTM, and GRU) for training from character-level language models on our selected dataset
- Finally, generate some lyrics from our trained model.

3 How to Generate Lyrics

Lyrics writing is a creative work. Generally lyrics construct a meaningful and rhythmic pattern which might be an abstract or direct. We used multi-layer RNN and LSTM to produce this. The reasons of choosing those are explained below:

3.1 What is RNN

Neural Network is extremely powerful in the field of machine learning. The Convolutional Neural Network (CNN) [5] used a fixed size input and output. These are game changer in terms of non-sequential data. But, when Sequential or temporal data needed to be trained then it becomes less useful. For example, If we train a CNN on a novel starting from first page, the CNN Will analyze the name of the characters stated in the first page and then in the second page while training a event in that story it will forget the characters name from the first page. Because CNN can't handle sequential or temporal data. For this type of problem (Sequential and temporal) Recurrent Neural Network [6] would be extremely good. Like our project goals, we need to generate complex sentence, rhythm and sequence to used as song lyrics. RNN generally takes the input from the output of the previous. It basically looping into one another neuron. After completing a forward pass on the second forward pass it takes some information from the first forwards pass to predict the second. Therefore the information from the first pass can be transferred throughout the network. Thus, it can remember the "past". Following is a simple picture of how RNN Works.

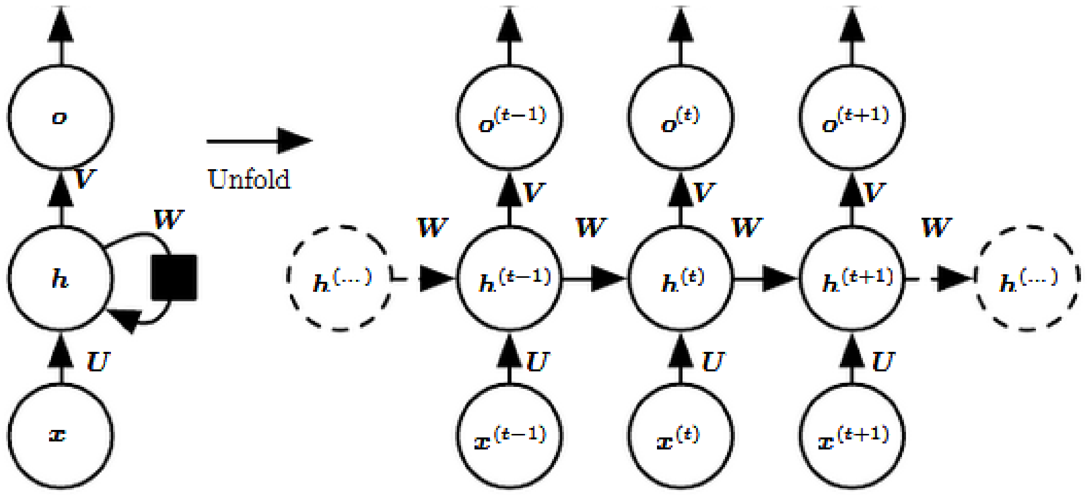


Figure 1: How RNN Works. Source [7]

3.2 Long Short Term Memory

If we backpropagate the model generated by RNN and calculate the gradients of the loss, we will find that the gradients are gradually becomes smaller into the network. This means that the more layers we add in the network, the less efficient the training becomes. This is oppositely true for a very large large gradients. This is called the vanishing gradient problems [8]. To solve this problem Long Short Term Memory [9] introduced for the prediction in RNN. It consider both the long term memory and the short term memory while making a prediction.

4 The Data

For this project I used a kaggle dataset [10] named "Song lyrics from 6 musical genres". There are two datasets "artists-data.csv" and "lyrics-data.csv", both have data on six musical genres:

- Rock
- Hip Hop
- Pop music
- Sertanejo (Basically the Brazilian version of Country Music)
- Carioca (Originated 60s US Funk)
- Samba (Typical Brazilian music)

I used the "lyrics-data.csv" file to train the model in this project after a proper cleaning and preprocessing the raw csv. There were several foreign languages in the list of lyrics. I took only "English" language and rest I drooped. The number of English language song lyrics was 114723. I only took 5000 song lyrics in a single dataframe and do the preprocessing and cleaning. Later I exported the full dataframe into a single .txt file for training. The full process including the notebook [11] is in the github repository. Following is a graphical picture of the most used words in the lyrics data out of final dataset.

5 Project Implementation

I implemented the project from a docker container. The docker container has all the required dependency and run on ubuntu 14.04. The torch-rnn is a alternate implementation with torch7 [12]. The reason I choose docker over installing in my computer is, My current computer is not compatible with the torch7 version. The main implementation was done over torch, lua and nural net [13]. I am describing the process here, you can also find the process in the readme file of the github repository too.

- Docker Composer

  Connect to the docker bash in the container

  ```
  docker run --rm -ti crisbal/torch-rnn:base bash
  ```

- Processed dataset download

  Download the processed dataset into the docker container

  ```
  curl https://raw.githubusercontent.com/AsifKibria/NLP_Text_Generation/main
  /Data/input.txt -o data/lyrics.txt
  ```

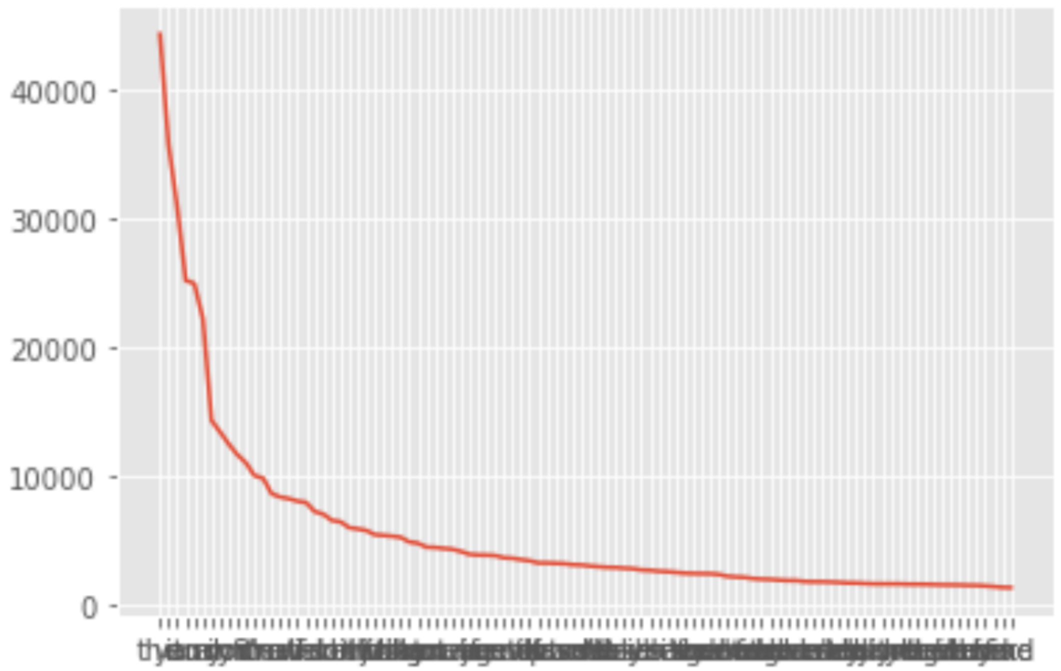
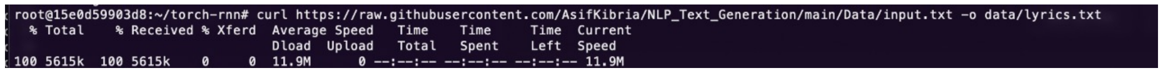


Figure 2: Most used words line plot, First 30 words used most frequently including ”the”, ”i”, ”in”, ”to” etc.



```
root@15e0d59903d8:~/torch-rnn# curl https://raw.githubusercontent.com/AsifKibria/NLP_Text_Generation/main/Data/input.txt -o data/lyrics.txt
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 5615k  100 5615k    0     0  11.9M      0 --:--:-- --:--:-- --:--:-- 11.9M
```

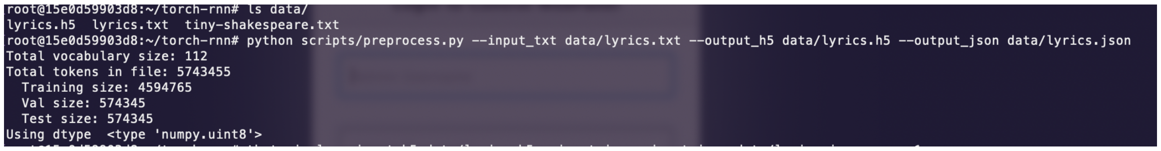
Figure 3: Curl download the processed dataset

- Preprocess the lyrics

  We will preprocess the lyrics to create the h5 files and json files of the lyrics in the ’data’ directory.

  ```
  python scripts/preprocess.py --input_txt data/lyrics.txt --output_h5
  data/lyrics.h5 --output_json data/lyrics.json
  ```

- Run the training on the dataset
  Now we will train the .h5 and .json files in the docker container. The training time will be very long depending on the performance of the system. the ’-gpu -1’ flag is for not having any gpu, if you have gpu accelerated with CUDA you need to download

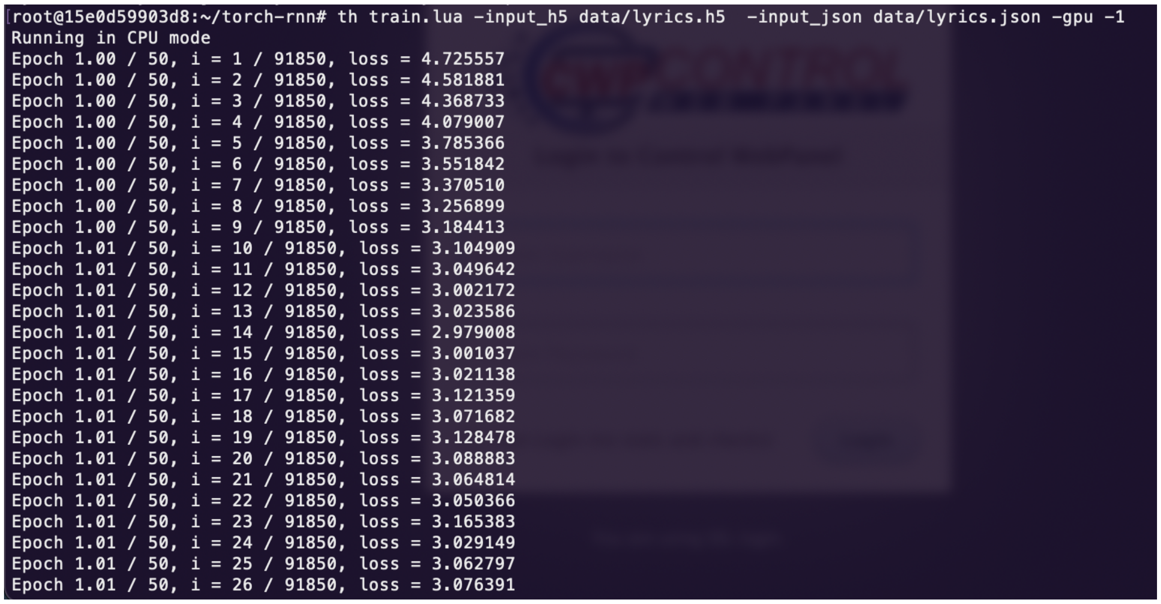


```
root@15e0d59903d8:~/torch-rnn# ls data/
lyrics.h5  lyrics.txt  tiny-shakespeare.txt
root@15e0d59903d8:~/torch-rnn# python scripts/preprocess.py --input_txt data/lyrics.txt --output_h5 data/lyrics.h5 --output_json data/lyrics.json
Total vocabulary size: 112
Total tokens in file: 5743455
  Training size: 4594765
  Val size: 574345
  Test size: 574345
Using dtype  <type 'numpy.uint8'>
```

Figure 4: Preprocessing the dataset to get the training size.

the nvidia-docker from https://github.com/NVIDIA/nvidia-docker . and run all the command without '-gpu -1' flag.

```
th train.lua -input_h5 data/lyrics.h5  -input_json data/lyrics.json -gpu -1
```



```
root@15e0d59903d8:~/torch-rnn# th train.lua -input_h5 data/lyrics.h5  -input_json data/lyrics.json -gpu -1
Running in CPU mode
Epoch 1.00 / 50, i = 1 / 91850, loss = 4.725557
Epoch 1.00 / 50, i = 2 / 91850, loss = 4.581881
Epoch 1.00 / 50, i = 3 / 91850, loss = 4.368733
Epoch 1.00 / 50, i = 4 / 91850, loss = 4.079007
Epoch 1.00 / 50, i = 5 / 91850, loss = 3.785366
Epoch 1.00 / 50, i = 6 / 91850, loss = 3.551842
Epoch 1.00 / 50, i = 7 / 91850, loss = 3.370510
Epoch 1.00 / 50, i = 8 / 91850, loss = 3.256899
Epoch 1.00 / 50, i = 9 / 91850, loss = 3.184413
Epoch 1.01 / 50, i = 10 / 91850, loss = 3.104909
Epoch 1.01 / 50, i = 11 / 91850, loss = 3.049642
Epoch 1.01 / 50, i = 12 / 91850, loss = 3.002172
Epoch 1.01 / 50, i = 13 / 91850, loss = 3.023586
Epoch 1.01 / 50, i = 14 / 91850, loss = 2.979008
Epoch 1.01 / 50, i = 15 / 91850, loss = 3.001037
Epoch 1.01 / 50, i = 16 / 91850, loss = 3.021138
Epoch 1.01 / 50, i = 17 / 91850, loss = 3.121359
Epoch 1.01 / 50, i = 18 / 91850, loss = 3.071682
Epoch 1.01 / 50, i = 19 / 91850, loss = 3.128478
Epoch 1.01 / 50, i = 20 / 91850, loss = 3.088883
Epoch 1.01 / 50, i = 21 / 91850, loss = 3.064814
Epoch 1.01 / 50, i = 22 / 91850, loss = 3.050366
Epoch 1.01 / 50, i = 23 / 91850, loss = 3.165383
Epoch 1.01 / 50, i = 24 / 91850, loss = 3.029149
Epoch 1.01 / 50, i = 25 / 91850, loss = 3.062797
Epoch 1.01 / 50, i = 26 / 91850, loss = 3.076391
```

Figure 5: Running the Training process, The number 91850 indicated the number of iteration will be done

- Sampling/Generating Lyrics
  After the training has been done, you'll see checkpoints are created after every 1000 iterations. The .t7 model files will be used to generate the lyrics. To generate a lyrics of 600 characters use the following.

```
th sample.lua -checkpoint cv/checkpoint_83000.t7 -length 600 -gpu -1
```

You can choose any checkpoint from the generated checkpoints to sampling.

6 Samples of Generated Lyrics

Following is the four output generated in this project.

- Sample 1: (Seed: Do I dare disturb the universe)

```
Do i dare disturd the universe wide eyes.
Break at you when everything.
Heart this day, yeah yeah, new mountate that wings.
And I ain't so replaced of lime.
Sinces was a gross "Repeat is Jenge??.
Caubles it's the clear, show felt, tied and wake  crumper.
Elevil ale so cold home. Where did me since ever and you all find.
Maiden arms or running. Their eyes Â¬Âsorta may crazy dies.
These cold mirror shit looks ya.
No red and you'll.
From real. on the closed with your dead.
I got song.
This is off a perfect.
Oh sky alone to spoke, But now it's only.
Behat I'm just one.
When hook out of more half day.
You're stare to prensemmer.
Forgotten is precity.
I'm pleasure land woman like a dop. Baby thatâÃÃ´s not resperits
```

- Sample 2: (Seed: Do I dare disturb the universe)

```
Do i dare disturd the universe, not think of the pain again.
To nothing to find out! hen - barely. What she made me days gonne On.
You're not over and, your side.
We'll know what your heart.
But I want to throw our bed.
Just a small. Avoids is down it.
Do me a punk havin' me watch.
Slildromite gays. There's a party man chack you tonight.
Buying race of La, The Wackest,.
And I useigin I'm tired. to say the best alone. And hard to take your faster, he fl
```

```
I find you go up town. Too much longer girl. Let go down the world.  Don't listen.
```

- Sample 3: (Seed: Love)

```
Love to be alone, I'm head you so feeling cared attage.
Over me of all your tomorrow on. You've been again acroll last side.
Lost elections come true goes. To be broken nightmare.
Oh my hard. Promise you needle emang dream. we ain't getting learning at the heart.
I'll find yourself until that you waste.
Get out there's a mikuntary. Before it mercy.
I'm not the jack. My letter, beauty was between knew.
Running you roll and got no sickly.
I'm not control. Ooh own pessivis start.
Sew sailtured holy fate. Take it to me to the truth that I just pull the stars.
```

- Sample 4: (Seed: Love)

```
Love Spy glocks of a front of air wheel. Caught up watch for getting.
It's coming. I'd let me follow me home. How do I carry around.
and your crying sense of here, baby. And I won't be all this day.
What takes you running. Goes a done darlin'.
we're the path Children balling.
Surroigined, I'm like one outside.
And you're not another biet. Don't need a mether greez.
We, we'll hold off my reason and stance.
Yes down I'll tried for sorry. Sinch of a hopical lives.
You and those are ronnisite step.
You all shot like deach you understand. How more never a little.
```

7 Conclusion and Future Prospect

I would say the quality of lyrics is not great, we need to improve that in a further project. In future we can also try to generate text and evaluate the text against a grammar and context/sentiment classification for making the text more sense and more humanly. Modern computing changing very fast, our life is being changing with the development of technology over time. I am certain there will be a day when we will enjoy computer generated lyrics with music and vocal just like we are enjoying. That day is marching fast.

References

[1] H. T. T. R. A. G. Eric Malmi, Pyry Takala, “Dopelearning: A computational approach to rap lyrics generation,” 2016.

[2] T. Joachims, “Optimizing search engines using clickthrough data.” *In Proceedings of the 8th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 133–142, 2002.

[3] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” *arXiv preprint arXiv:1406.1078*, 2014.

[4] F. A. Gers, J. Schmidhuber, and F. Cummins, “Learning to forget: Continual prediction with lstm,” *Neural computation*, vol. 12, no. 10, pp. 2451–2471, 2000.

[5] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, “Face recognition: A convolutional neural-network approach,” *IEEE transactions on neural networks*, vol. 8, no. 1, pp. 98–113, 1997.

[6] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[7] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning.* MIT Press, 2016, http://www.deeplearningbook.org.

[8] S. Hochreiter, “The vanishing gradient problem during learning recurrent neural nets and problem solutions,” *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 6, no. 02, pp. 107–116, 1998.

[9] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, “Lstm: A search space odyssey,” *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2222–2232, 2016.

[10] A. Neisse, “Song lyrics from 6 musical genres.” [Online]. Available: hhttps://www.kaggle.com/neisse/scrapped-lyrics-from-6-genres

[11] M. G. K. Asif, “preprocessing of the lyrics data set.” [Online]. Available: https://github.com/AsifKibria/NLP$_{T}ext_{G}eneration/blob/main/Data_{P}reprocessing.pdf$

[12] J. Johnson, “Torch-rnn implementation.” [Online]. Available: https://github.com/jcjohnson/torch-rnn

[13] A. Karpathy, “Char-rnn implementation.” [Online]. Available: https://github.com/karpathy/char-rnn