PROJECT REVIEW III

TITLE : **Music Generation using Deep Learning**

AUTHORS : **1. Divyansh Singh Raghav 19BEE0067**

2. Naman Kapoor 19BEE0012

PROJECT GUIDE : **Dr. Monica Subashini M**

1. Abstract

Our aim is to solve this problem of generating new music of a particular genre by using the knowledge of Deep Learning. Our model simply should not return the output, very close to the actual data, but also it should not provide the output far off from the data provided. We cannot expect the model to give us a professional quality music in one-go but we can expect results that are decent quality tunes and worth hearing to.

Since the model we are going to be dealing with will be a sequence of musical notes, so the true option left for us is RNNs, or maybe LSTMS. Since, the number of inputs will be equal to outputs, it makes a total sense to use Many-to-Many RNN.

The simpler way to define the on-going flow is considering we have a music that is represented as [A,B,C,D,....,W,X,Y,Z]. Let's say we give "A" an input to the model, and expect it to give "B" as an output. Similarly, in next time-step we will input "B" as an output and expect "C" as an output. This is how we train the whole sequential music to the model.

Our model inputs a sequence of nodes, it is a multi-class classification, and hence we will use "Categorial Cross-Entropy" as a loss function. The final layer in the model will be a "SOFTMAX" activation layer. The number of "Softmax" activation units in final layer will be equal to the number of all unique characters in all of the music in train data.

Selection of Data :-

There are several sources available for providing a dataset, but the ones i have used is as follows :

The ABC Music project - The Nottingham Music Database
Maintainer: James Allwright (home page) Download the collection The Nottingham Music Database

abc.sourceforge.net

At the end of this project, we expect to build a low-end pre trained model which generates new samples for a particular genre.

2. Introduction

-> Literature Review

Music Generation with AI is a relatively new topic as compared to the fields like computer vision, sentiment analysis, stock prediction etc. The earliest work in field of music generation by Neural network was by Chi J Chen and Rislo M in 2001. Where the created the simple melody. One of the main hinderance cited was the 'lack of a global structure in music'.

We can apply different deep learning algorithms for our purpose such as:
1- 'MIDI NET 2017' by L C Yang and S Y Chow who used GAN(Generative Adversarial Network) approach with use of CNNs.

2- H W Dong have introduced a new framework 'Musegan' which consists of a GAN based models for multi track music generation.

3- Choi Et al, in 2016 have explored applications of RNNs with LSTM units for automatic generation of Jazz cord music progressions and rock music drum tracks.

Other notable papers on this topic are :

- **DEEPBACH: A STEERABLE MODEL FOR BACH CHORALES GENERATION** Aug 2017 by Gaëtan Hadjeres, Francois Pachet
- **Music Generation by Deep Learning - Challenges and Directions** Feb 2020 by Jean-Pierre Briot, Francois Pachet
- **ChordRipple: Recommending Chords to Help Novice Composers Go Beyond the Ordinary** Mar 2016 by Cheng-Zhi, Anna Huang
- **A First Look at Music Composition using LSTM Recurrent Neural Networks** Aug 2002 by D. Eck, J. Schmidhuber
- **Computational Creativity and Music Generation Systems: An Introduction to the State of the Art** Apr 2020 by Filippo Carnovalini
- **Dual-track Music Generation using Deep Learning** May 2020 by Sudi Lyu, Anxiang Zhyang
- **LSTM based approach for Generating Music from MIDI notes** Jan 2020 by Beschi Raja, Sandhya Gangadharan, S Sampeter

-> Novelty of our idea

We have implemented an easier to understand model for music generation compared to the earlier works, the network mimics the given data to produce original tunes well enough.

The simplicity of our approach regarding this advanced field, makes it easier for the developers to further implement improvements on top of it.

3. Methodology

-> **Dataset Description**

- The “Nottingham Music Database” (NMD), assembled around 2001, has been appearing more and more in applied machine learning research and teaching over the past few years

- This database, presented by Eric Foxley contains about 1200 folk melodies, mostly British & American. They mostly come from the repertoire over the years of Fred Folks Ceilidh Band, and are intended as music for dancing.

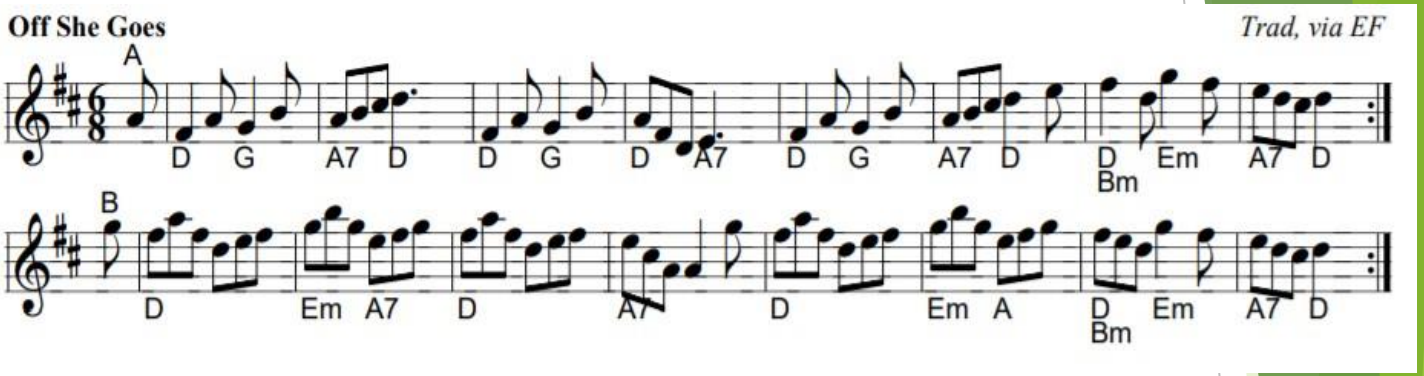
- The music consists of Jigs, reels etc. which do not comprise of complex musical structures, ideal for prediction via LSTM.

->**Sample Data**

- Dataset provides us with the sheet music of each melody, which we process further into MIDI format. Here’s an audio sample for one of the reels “Off She Goes”:

Digitally we can convert the sheet music, ABC format, that is, information of notes to be played. In our case-study we will focus much more on ABC-notation because it is easy to understand and easy to represent music using just sequence of characters .

Here's the ABC notation :



Here's the MIDI notation:

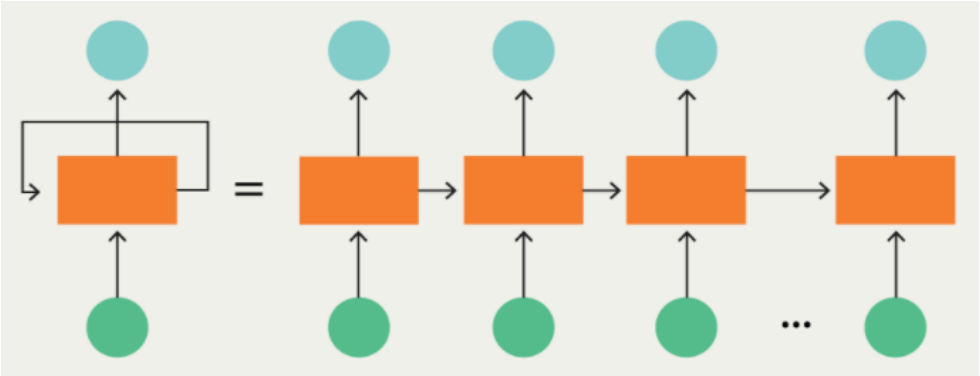
```
...
<music21.note.Note B> 72.0
<music21.chord.Chord E3 A3> 72.0
<music21.note.Note A> 72.5
<music21.chord.Chord E3 A3> 72.5
<music21.note.Note E> 73.0
<music21.chord.Chord E3 A3> 73.0
<music21.chord.Chord E3 A3> 73.5
<music21.note.Note E-> 74.0
<music21.chord.Chord F3 A3> 74.0
<music21.chord.Chord F3 A3> 74.5
<music21.chord.Chord F3 A3> 75.0
<music21.chord.Chord F3 A3> 75.5
<music21.chord.Chord E3 A3> 76.0
<music21.chord.Chord E3 A3> 76.5
<music21.chord.Chord E3 A3> 77.0
<music21.chord.Chord E3 A3> 77.5
<music21.chord.Chord F3 A3> 78.0
<music21.chord.Chord F3 A3> 78.5
<music21.chord.Chord F3 A3> 79.0
...
```

*Chords are essentially a container for a set of notes that are played at the same time.

->**Block Diagrams**

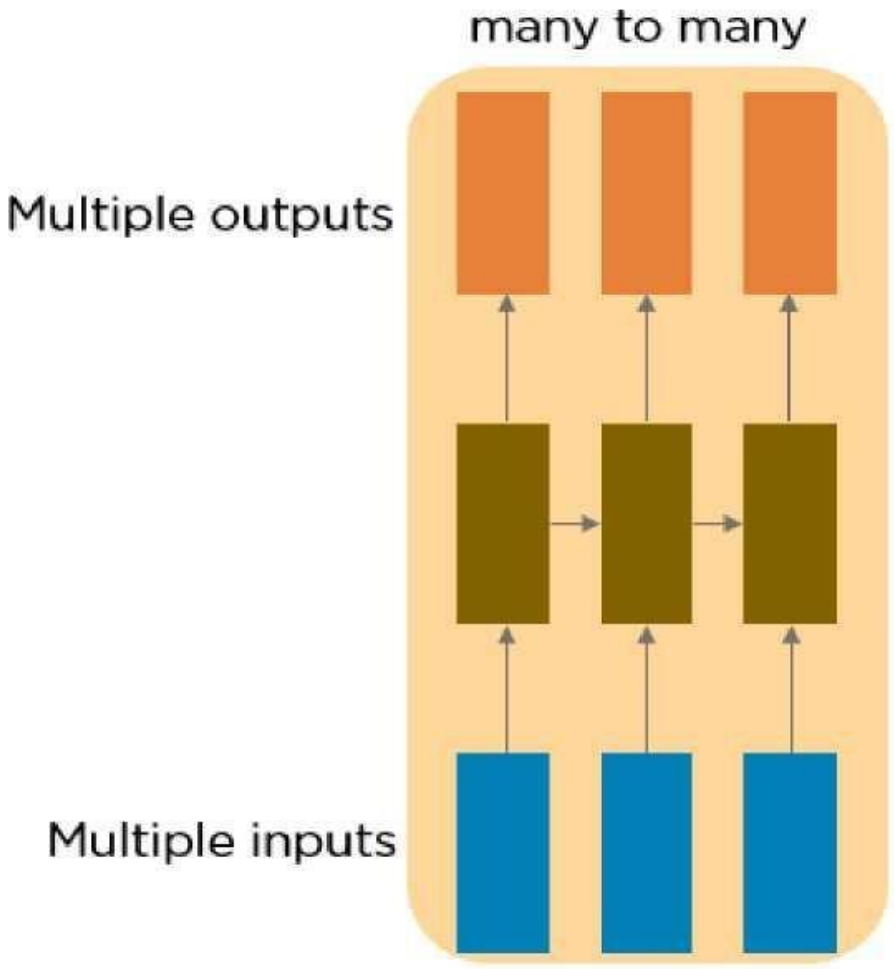
Henceforth, we have the obtained MIDI file, basically the sequence of notes and chords per each melody. We would like to feed these sequences as input to a RNN based model architecture, since Recurrent Neural Networks (RNNs) are observed to work best for these sequential type inputs.

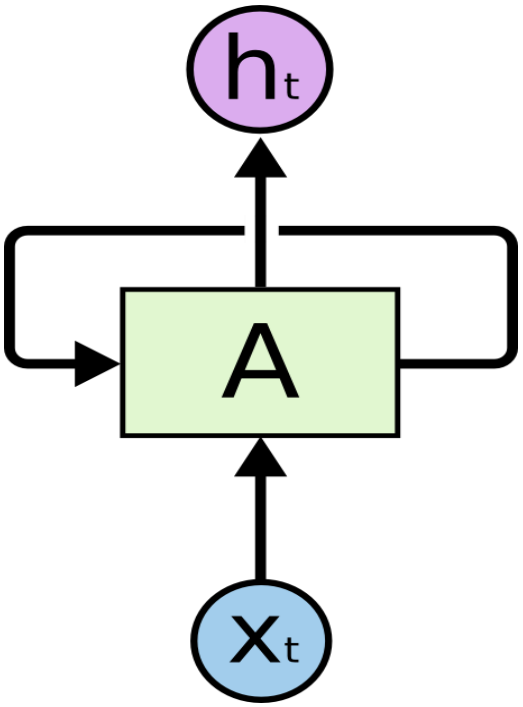
RNN are networks with loops in them, allowing information to persist.



In the above diagram, a chunk of neural network, A, looks at some input xt and outputs a value ht. A loop allows information to be passed from one step of the network to the next. A recurrent neural network can be thought of as multiple copies of the same network, each passing a message to a successor.

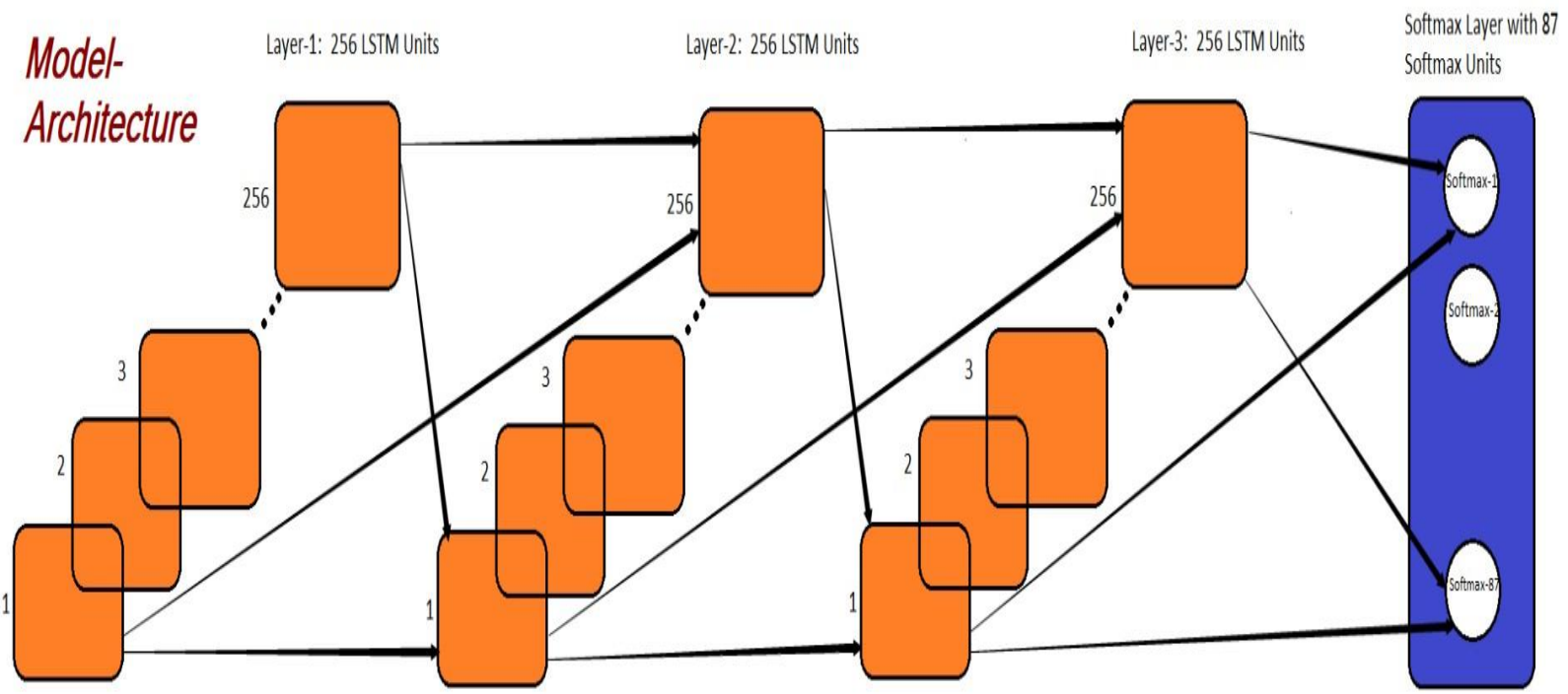
Our data is a sequence of characters. We will feed one after the other character of the sequence to RNN and the output will be the next character in the sequence. So, therefore, the number of output will be equal to the number of inputs. Hence, we will be using Many-to-Many RNN, where number of output is equal to the number of input





For the above image, Xt is a single character at time ‘t’ which is given as an input to RNN unit. Here, O_t-1 is an output of the previous time ‘t-1’ character which is also given as an input to RNN at time ‘t’. It then generates output ‘ht’. This output ‘ht’ will again be given as input to RNN as ‘O_t-1’ in next time step.
This output ‘ht’ will be a next character in sequence. Let say our music is represented as [a, b, c, a, d, f, e,…]. Now, we will give first character ‘a’ as an input and expects RNN to generate ‘b’ as an output. Then in next time-step we will give ‘b’ as an input and expects ‘c’ as an output. Then in next time-step we will give ‘c’ as an input and expects ‘a’ as an output and so on. We will train our RNN such that it should output next character in the sequence.

This is how it will learn whole sequence and generate new sequence on its own.



Putting together previous blocks, here's our final model :

For each LSTM layer we have 256 cells of input which it maps to 256 output units. LSTM is a special kind of RNN, for remembering long term dependencies, suited for our purpose.

At each time step all of the RNN units generate output which will be an input to the next layer and also the same output will again be an input to the same RNN unit.

We have 87 unique characters in our dataset and we want that the output at each time-stamp will be a next character in the sequence which is one of the 87 characters. So, time-distributed dense layer contains 87 "Softmax" activations and it creates a dense connection at each time-stamp. Finally, it will generate 87 dimensional output at each time-stamp which will be equivalent to 87 probability values as to which character is the next output.

4. Implementation

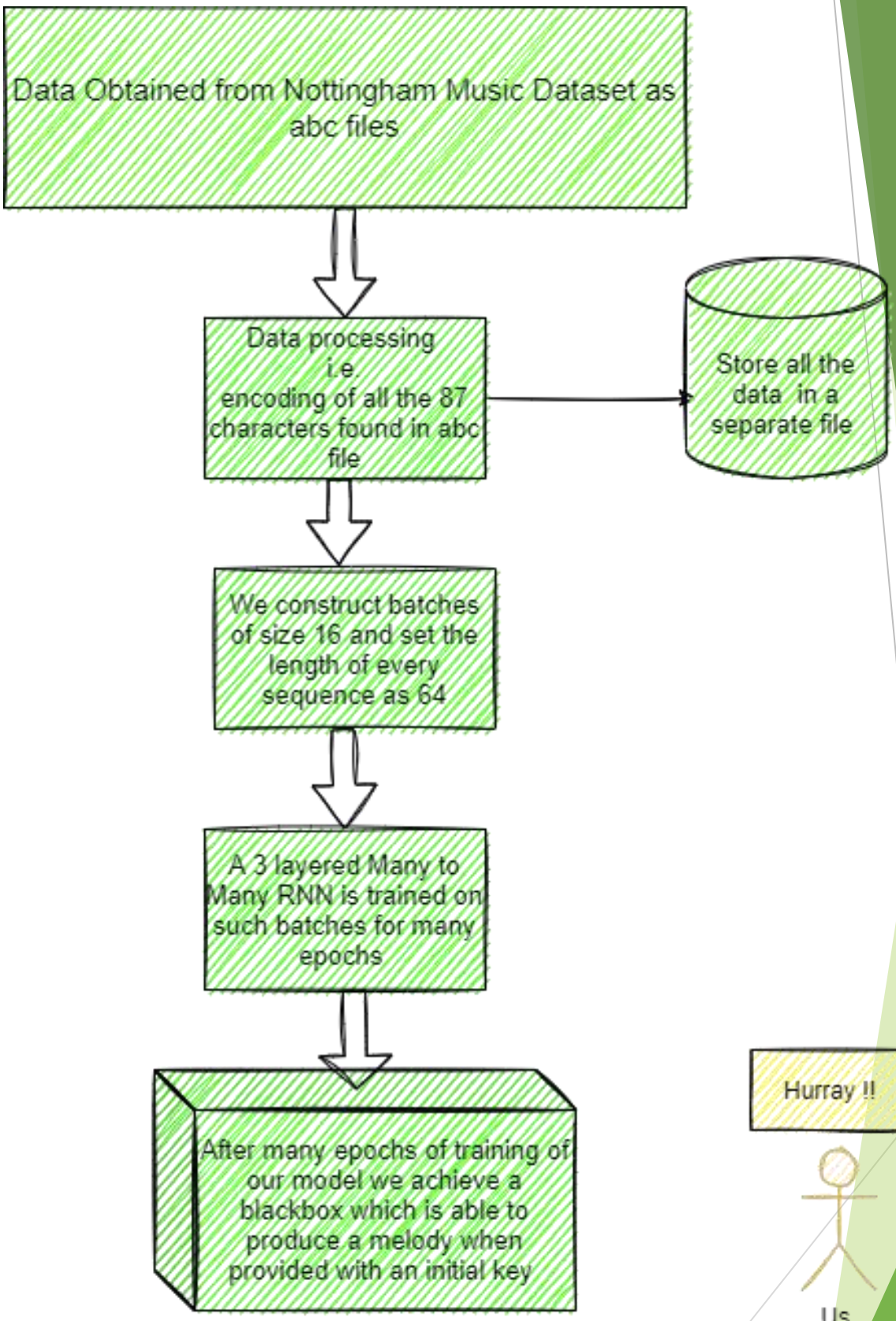
-> **Brief details on Software**

We used Google Colaboratory notebook for this project, as formatting, running different parts of code is easier in form of cells than in the local IDE. Also, the training each epoch might take up to few hours, hence it is better to use cloud based computation in place of cluttering our CPU during training period. Google Colab notebooks can provide runtimes as much as 12 hours.

Python Libraries used :
Numpy : We store the arrays of data into numpy arrays, as numpy arrays offer much faster and efficient operations for arrays as compared to traditional Python Data Structures.

Keras: Keras is a high level Deep Learning framework, it takes care of all the tasks pertaining to Deep Learning in this project, that is, building the many to many RNN network, training the network, optimizing using ‘ADAM’, implementing regularization by ‘dropout’. Without a framework, this project would have exceeded more than a thousand lines of code while taking double the time.

-> **Flowchart**


Data Obtained from Nottingham Music Dataset as abc files
Data processing i.e. encoding of all the 87 characters found in abc file
Store all the data in a separate file
We construct batches of size 16 and set the length of every sequence as 64
A 3 layered Many to Many RNN is trained on such batches for many epochs
After many epochs of training of our model we achieve a blackbox which is able to produce a melody when provided with an initial key
Hurray !!
Us

->Pseudocode

- Assign : indexes to 87 unique chars
- Function : Create Batch; Size =16; Length=64 chars
- Initiate : Keras RNN Model
- Add: 3 LSTM layers; units =256; dropout=0.2
- Final Activation : ‘Softmax’, size[0]=87
- Train : epochs = 10; optimizer=‘ADAM’
- Plot : Accuracy v/s epochs while testing

5. Results and Discussion

-> Brief Discussion

While training the model we can think of the task at the hand as a multiple classification problem, that's why we use "Softmax" in the network architecture, although, in contrast to a regular classification problem, here all the data is used as the training dataset. The accuracy and "cross entropy loss" is measured with respect to how well our model learns the musicality or the structure of the samples provided to it.

Please note that the, accuracy in this case reflects how well the model is understanding the British Folk tunes. However, higher accuracies are not very good as it would mean that the network is simply copying, rather than composing something original. We use accuracy v/s epoch plot to see if our model is adapting as we feed it more data.

Also, since we place human observers at the testing end of the model, we don't have a numeric accuracy measure of the output performance. We tune the network's hyperparameters according to the user's understanding of how well the model is performing

-> Results

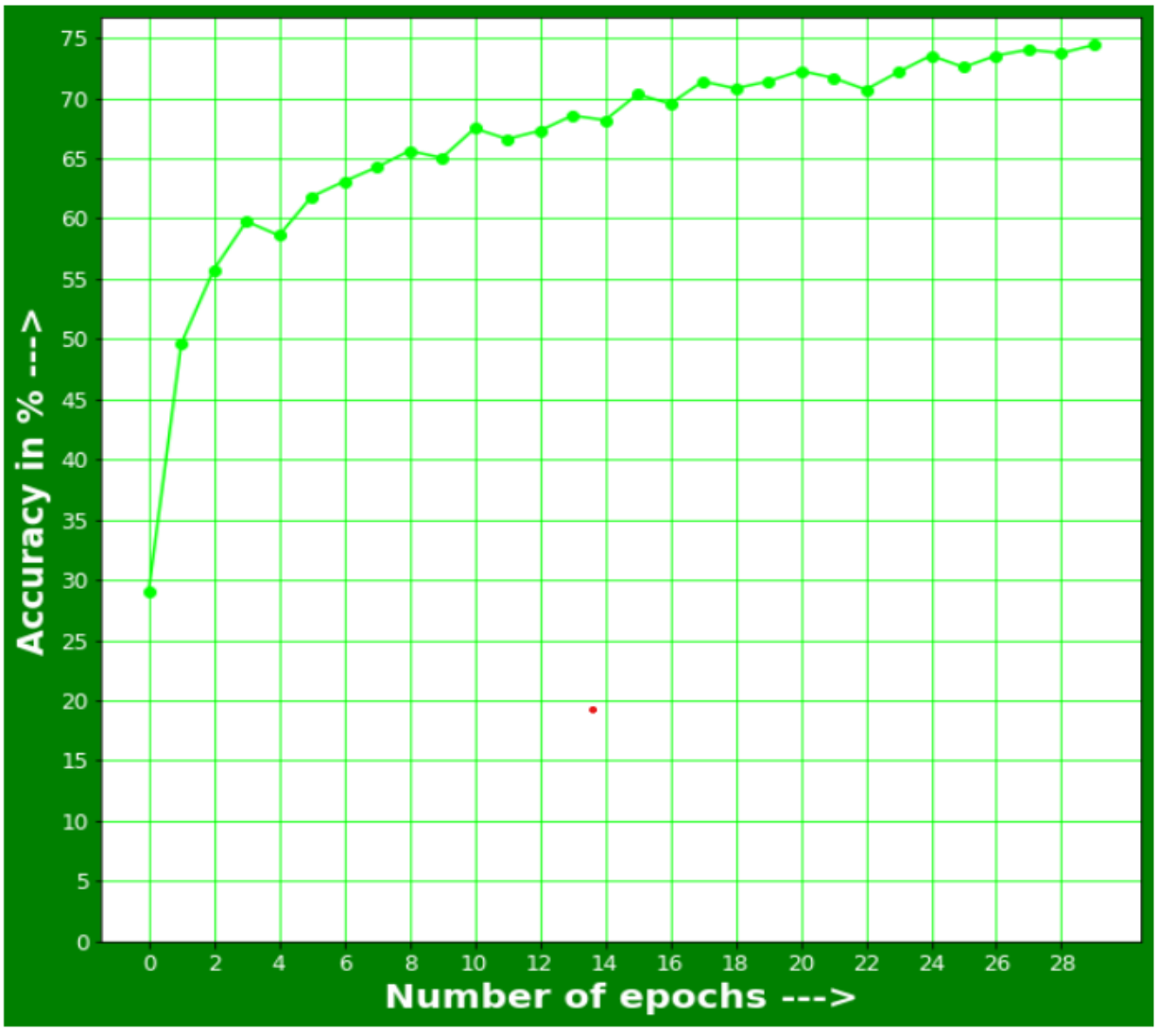
TESTING ACCURACY FOR 30 EPOCHS :

J Comp_EEE1007.ipynb

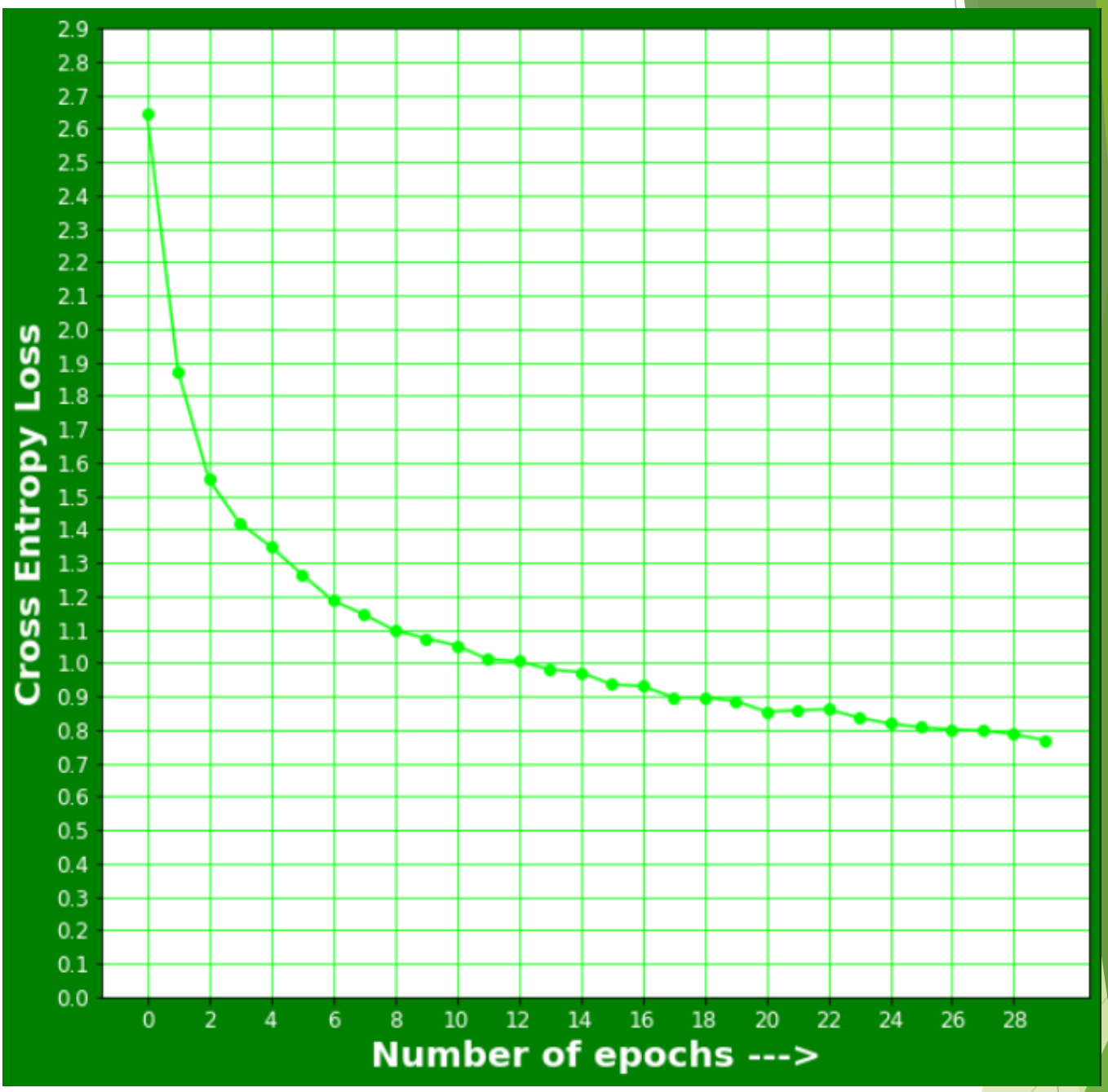
File Edit View Insert Runtime Tools Help Saving...

+ Code + Text

```
16 plt.yticks(np.arange(0,80,5),color='w',size='12')
17 plt.savefig('ACCvsEPO')
18 plt.show()
19
```



CROSS ENTROPY LOSS FOR 30 EPOCHS :



6. Conclusion

As we draw to a conclusion of this project, we would exclusively like to show our gratitude to Dr. Monica Subashini who patiently guided us and encouraged us, without which this project wouldn't have come along so efficiently

Our project does a good job at capturing the essence of the music sample provided to it. Output samples after 50 epochs are able to pass of as authentic English folk jigs. We want to extend our model to more exquisite genres like Indian Classical or contemporary Hollywood.

Also, the prospect of user tuning the network according to his liking of the tracks being produced, can be realized with a similar approach, but that would require much faster training of the LSTM units.

At the end, we would just like to say that AI has got a huge prospect in music industry, be it generation of new tracks or re-mixing the old ones. Hence, this a emerging new area of Deep Learning and should receive adequate attention of the researchers to develop into something commercially viable.

7. References

1 https://www.appliedaicourse.com/
2 https://folkrnn.org/
3 https://en.wikipedia.org/wiki/ABC_notation
4 https://abcjs.net/abcjs-editor.html
5 https://towardsdatascience.com/how-to-generate-music-using-a-lstm-neural-network-in-keras-68786834d4c5
6 http://karpathy.github.io/2015/05/21/rnn-effectiveness/
7 https://medium.com/artists-and-machine-intelligence/neural-nets-for-generating-music-f46dffac21c0

Research Papers

- **A research about automatical music generation by computer**
  Authors: Liao Enyang; Tao Jun
  Published in: 2012 IEEE Symposium on Robotics and Applications (ISRA)

- Automatic melody generation using Neural Networks and Cellular Automata
  Authors: **Ivana D. Matić; António Pedro Oliveira; Amílcar Cardoso**
  Published in: **11th Symposium on Neural Network Applications in Electrical Engineering**

- Komposer – Automated Musical Note Generation based on Lyrics with Recurrent Neural Networks
  Authors: **Dulan. S. Dias; T. G. I. Fernando**
  Published in: **2019 1st International Conference on Artificial Intelligence and Data Sciences (AiDAS)**

- Automatic Ontology Generation for Musical Instruments Based on Audio Analysis
  Authors: **Şefki Kolozali; Mathieu Barthet; György Fazekas; Mark Sandler**
  Published in: **IEEE Transactions on Audio, Speech, and Language Processing (Volume: 21, Issue: 10, Oct. 2013)**

- Several models and applications for deep learning
  Authors: **Hengchang Hu; Bo Liu; Pan Zhang**
  Published in: **2017 3rd IEEE International Conference on Computer and Communications (ICCC)**

- Person Re-Identification by Deep Learning Multi-Scale Representations
  Authors: **Yanbei Chen, Xiatian Zhu, Shaogang Gong**
  Published in: ICCV_2017_workshops

- Recurrent neural network training with dark knowledge transfer
  Authors: **Zhiyuan Tang; Dong Wang; Zhiyong Zhang**
  Published in: **2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**

- Recurrent Neural Networks: An Embedded Computing Perspective
  Authors: Nesma M. Rezk; Madhura Purnaprajna; Tomas Nordström; Zain Ul-Abdin
  Published in: IEEE Access (Volume: 8)

- Restricted Recurrent Neural Networks
  Authors: Enmao Diao; Jie Ding; Vahid Tarokh
  Published in: 2019 IEEE International Conference on Big Data (Big Data)

- Recurrent Neural Network Architectures for Analysing Biomedical Data Sets
  Authors: Mohammed Khalaf; Abir Jaafar Hussain; Robert Keight; Dhiya Al-Jumeily; Russell Keenan; Carl Chalmers; Paul Fergus
  Published in: 2017 10th International Conference on Developments in eSystems Engineering (DeSE)