MeMozart: Generating Music with Recurrent Networks

Chand Anand
Department of Computer Science
University of California, San Diego
San Diego, CA 92093
canand@ucsd.edu

Gurkanwal Singh Batra
Department of Computer Science
University of California, San Diego
San Diego, CA 92093
gbatra@ucsd.edu

Divya Shree Pitta
Department of Computer Science
University of California, San Diego
San Diego, CA 92093
dpitta@ucsd.edu

Kaustubh Tanmane
Department of Computer Science
University of California, San Diego
San Diego, CA 92093
ktanmane@ucsd.edu

Jigya Yadav
Department of Computer Science
University of California, San Diego
San Diego, CA 92093
jyadav@ucsd.edu

Abstract

Recurrent Neural Networks can be trained to produce sequences given some input, as exemplified by recent results in machine learning research, particularly in machine translation and sequence modeling. While vanilla RNNs work in these settings, they have certain problems (such as vanishing gradient) that make learning of deep models difficult. LSTM (Long Short Term Memory Networks) are a variant of RNNs that, due to their architecture, enable easy passage of gradient while retaining memory sequences for predictions. In this paper, we explore LSTMs and how they can be used for music generation. We conduct a variety of experiments with different architectures, optimizers, and sampling strategies to better understand the behavior of LSTMs in these settings. Finally, we provide music samples generated by our LSTM models.

1 Introduction

This project report discusses the use of Long Short Term Memory networks for the composition of music. The LSTM network is trained using characters from a music database, where the music notes are given in ABC format.

Recurrent neural networks make use of sequential information. They have a ”memory” which is used to capture information about the previous states. Given an input sequence, the recurrent neural network applies a recurrence formula at every time step based on the previous state of the current network and current input to produce an output and the new state. The same parameter and same recurrence formula is used at every time step.

However, RNNs are unable to ”remember” information for arbitrarily long times. Long Short Term Memory (LSTM) units are a variation of RNN which use cell states and gates to capture long-term dependencies in data. A typical LSTM unit consists of 4 gates which control previous information to erase from and new information to store in the cell state, the output to be produced and the update to hidden state. The cell state provides a mechanism for gradient to be backpropagated uniterrupted. Hence, LSTMs are able to resolve the vanishing gradient problem and learn long-term dependencies in data.

In this paper, we explore LSTMs and how they can be used for music generation. We conduct a variety of experiments with different architectures, optimizers, and sampling strategies to better understand the behavior of LSTMs in these settings.

2 Music Generation

2.1 Experimental Setup & Discussion

The experiments were conducted on the GPU support provided by the instructors for the course.

In order to recognize the best model, we ran various experiments on different input sampling techniques, more and fewer number of hidden layers and hidden units, GRU units, and several sequence lengths (patch sizes). We divide the data into appropriate training and validation sets, and follow all best practices. We describe each of the above-mentioned variations in subsections below.

It shall be noted that wherever we report ”loss”, we actually mean ”running loss” over epochs.

Here, we also state two models which are referred elsewhere in the report. Terminology for the definition of these models can be found in the subsections below.

Vanilla Model: This model consists of ”Random” sampling of patches with 30 sequence length, treating the input data as just one text file. There is one LSTM layer , with 100 hidden units. Learning rate for the model is 0.0001, and the model is trained for 500 epochs (or early stopped appropriately). Adam optimizer is used.

Best Model: This model consists of ”Every Tune” sampling of patches with 30 sequence length, treating the input data as just one text file. There is one LSTM layer , with 150 hidden units. Learning rate for the model is 0.0001, and the model is trained for 500 epochs (or early stopped appropriately). Adam optimizer is used.

It shall also be noted that we trained over 250 models in total for different variations of hyper-parameters and sampling techniques. This was necessary to fully understand the importance of various variables for training LSTM models for music generation.

2.1.1 Training Techniques

We extract the tunes from the training data provided and create a 80-20% training-validation split. Then we studied and experimented with different sampling techniques for training LSTM models. Particularly, we used the below three sampling techniques, where each of these explain the training for one epoch:

1. Random: Randomly choose 1000 patches of a fixed length [for example: 30] in each epoch, to train the LSTM models
2. Every Tune: we choose a patch of fixed size from all the tunes present in the training set in each epoch
3. Sliding over some tunes: we randomly choose a fixed number of tunes and do a sliding window with a fixed stride with a fixed sequence length, by conserving the the hidden state throughout the epoch

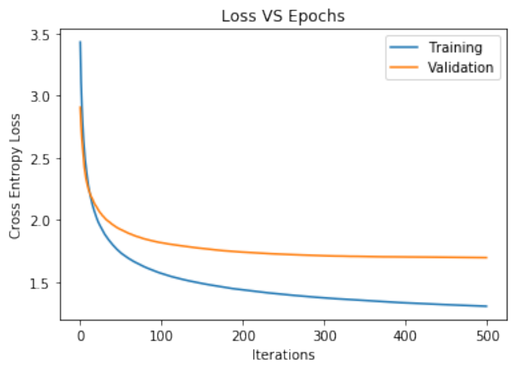
After experimenting with the above techniques, we find that “Every Tune” strategy works best, and has the lowest loss on validation set.

2.1.2 Convergence Criteria/Early Stopping

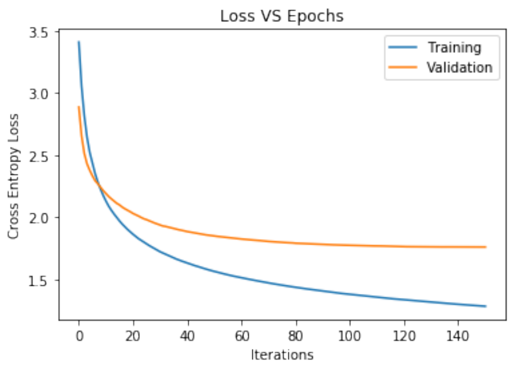
We limit the number of total epochs to be 500 for every model that we trained. Besides that, we also implement early stopping i.e. stop training when validation loss increases for 3 continuous epochs.

2.1.3 Variation in number of hidden layers

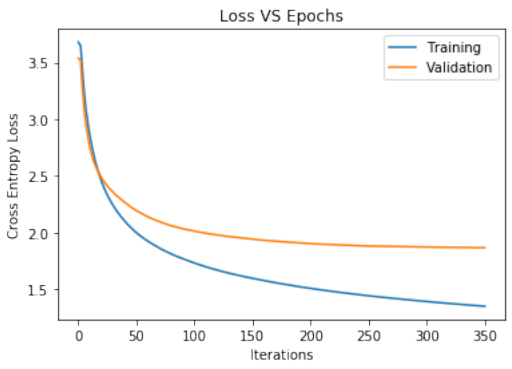
We study the variation of number of hidden layers on the performance of LSTMs for music generation. Below we plot and discuss the results obtained on trying out various number of hidden layers on the ”Best Model”



(a) 1 Hidden Layer



(b) 2 Hidden Layers



(c) 4 Hidden Layers

Figure 1: Plot of training/validation loss for different layers

It is observed that increasing the number of layers is not that helpful for producing higher quality tunes, and the converged running loss on both training and validation sets increases.

2.1.4 GRU units

In literature we see both the variants of RNNs: LSTMs and GRUs used on wide variety of modelling tasks which require RNNs to capture long term dependencies. In recent years GRUs are becoming a more popular choice among the community because it is a simpler model, with lesser parameters which trains faster than LSTM and it is easier to build larger RNN networks with it. Here we experiment with GRUs for music generation.

Model Architecture We experimented with RNN containing 2 GRU layers with 200 hidden units followed by a linear layer of size equal to character embedding. Dropout was used between the hidden layers as a regularizer and the optimizer used was Adam, with learning rate 0.001. [Rest of the details of training, evaluation and quantitative results are mentioned in Dropout section of the report.]

Observations In this task of music generation we tried both LSTM and GRU units, and found that LSTMs were giving a slightly better quality of music, which had more sensible long term correlations than GRU units, for RNNs with same number of layers. The reason this might be is because LSTM has more control over the cell state with more gates as compared to GRU units. Also, for the same number of layers and hidden-units, LSTM has more paramters than GRU unit based RNNs allowing it to learn better dependencies.

2.1.5 Experimenting with dropout

Dropout is used to prevent overfitting, by randomly dropping units and their connections from the network randomly. We experimented with using dropout on a network of 2 hidden layers. Multiple dropout rates of [0.1, 0.2, 0.3] were tried. In general, we observed that increasing dropout rate increasing the training time. With dropout of 0.2, we observed the lowest loss on validation set. More details about this are given in section 5.

2.1.6 Experimenting with curriculum learning

Curriculum learning is based on the intuition that it is better for the model to first learn simpler shorter sequences and then move onto learning longer difficult sequences. An LSTM network with 2 hidden layers having 100 units each was used in curriculum learning using Adam optimizer for weight updates. Sequence length is doubled after every 100 epochs during training.

It was observed that this model produced better tunes most of the time when compared to those trained on a fixed sequence length. The model converged in 330 epochs with training loss 1.4398 and validation loss 1.8940.

2.1.7 Experimenting with temperature

Temperature is used to control the exploration/exploitation strategy of the model while producing the output. It scales the logits before applying softmax. Therefore the probability of class i now becomes:

$$q_i = \frac{exp(z_i/T)}{\sum_j exp(exp(z_j/T)}$$

The character to be output is chosen from the probability distribution of all characters. When T is high, a softer probability distribution over classes is produced and therefore, the chance of randomly chosen character being output increases. When T is low, the character with the highest probability is more likely to be output.

We experimented with 3 values of temperature: [0.5, 1, 2] for the generation of music.

We observed that at T=2, the output produced was mostly gibberish and it was highly unlikely to be a valid tune, which suggests that this temperature is too high.
At T=0.5, the music samples generated were more likely to be structured and valid.
At T=1, out of all music samples produced, some were valid and sometimes the network produced random outputs. It was less likely to produce valid output compared to T=0.5. The best ABC notations were produced at T=0.5.

The ABC representations of the music generated at various temperatures is provided in the appendix. However, for T=2, no valid tune was produced and hence, the midi file has not been uploaded for that.

2.2 Music Samples

We evaluate the music samples generated, by the RNN using various empirical metrics as seemed suitable. In the music samples submitted we observe that the our tuned RNN learnt long dependencies between different notes, their scale and composition. It was also able to start a tune from notes which were predominantly used to start the tune in the dataset, and learnt to end the tune gracefully with the notes typically used to end the tune. It also learnt to generate music with multiple instruments, and music with different beat speed. We observed that the model primed with " $< start >$ " and with temperature values of 0.4,0.5,0.6 generated the best sounding music. Also the model often generates invalid midi files, in between generating some valid music files which is expected from the inherent rooted indeterministic nature of any generative class of model.

2.3 Hyperparameters & Model details

This best model consists of an LSTM network having the below characteristics:

Sampling strategy for training: ”Every Tune”

Sequence Length: 30

Number of layers: 1

Number of hidden units in each layer: 150

Learning Rate: 0.0001

Optimizer: Adam

Loss computation strategy: Running Loss

Max Epochs: 500

Early Stopping: True

We ran an exhaustive grid search on models with different combination of hyper-parameters, optimizers and sampling techniques. It shall be noted that we trained over 250 models in total for different variations of hyper-parameters and sampling techniques. This was necessary to fully understand the importance of various variables for training LSTM models for music generation. The model described above is the best model we found in terms of the converged validation running loss and has been marked in **bold** in Table 1.

3 Plots

The loss plot in 2 corresponds to our best performing model. It converges in 281 epochs with a minimum training loss of 1.276 and a validation loss of 1.741.

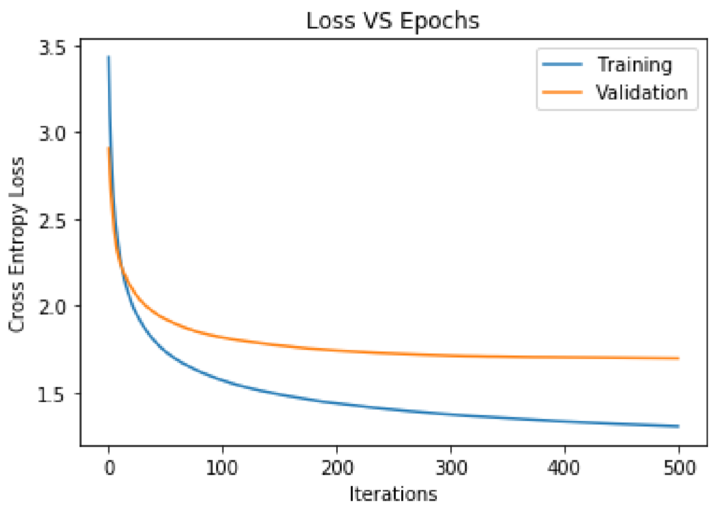
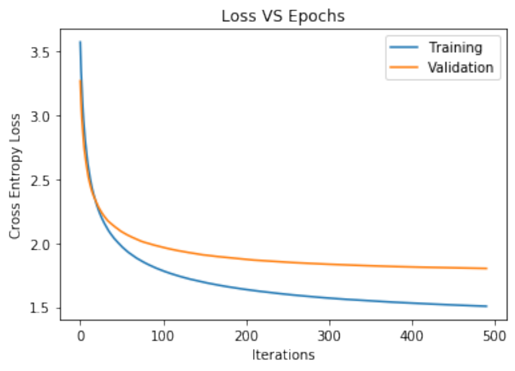


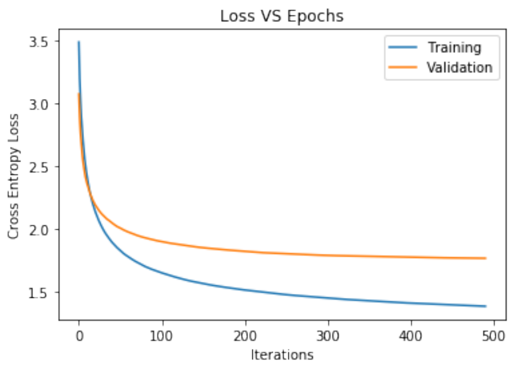
Figure 2: Loss vs. Training epochs

Discussion It was observed that the loss per sequence fluctuates a lot. This may be because sometimes the model comes across a simple, already learned sequence and sometimes it comes across a very difficult sequence. However, it was noticed that on an average, the loss is stable, as expected. Hence, to study the model during training and its convergence, a running loss over each epoch is used. All the plots make use of this heuristic.

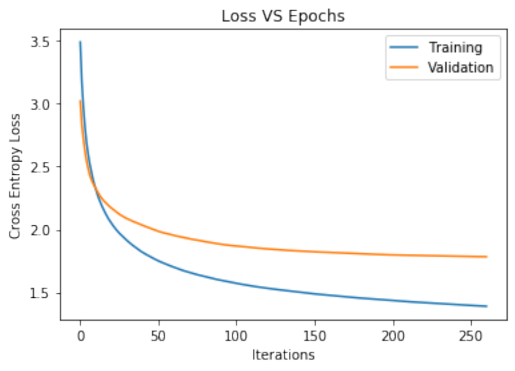
4 Varying number of hidden units



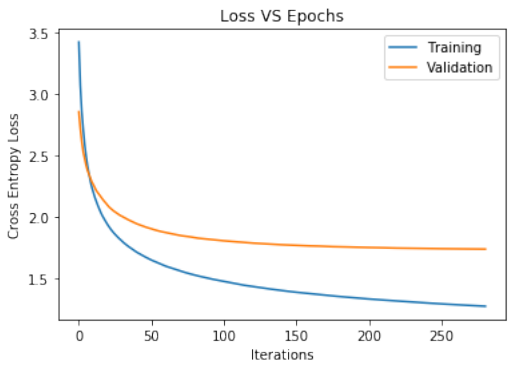
(a) 50 hidden units



(b) 75 hidden units



(c) 100 hidden units



(d) 150 hidden units

Figure 3: Plot of training/validation loss vs epochs with different number of hidden units

Discussion We experimented by taking different combinations of hidden units[50,75,100,150]. Out of these, the best model which gave the minimum validation loss on convergence was 1 hidden layer with 150 hidden units.

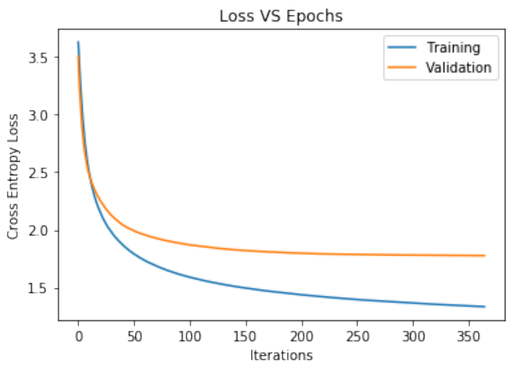
With increasing hidden units, we observe a drop in converged training and validation loss. This is as per our intuition, higher number of hidden units enable model to capture more variation in data as it introduces more parameters increasing higher degree of non-linearity in the network, and hence learns better internal representations. These results have been reported separately in Table 1.

As an extension, variation with number of hidden layers has been described in section 2.1.3

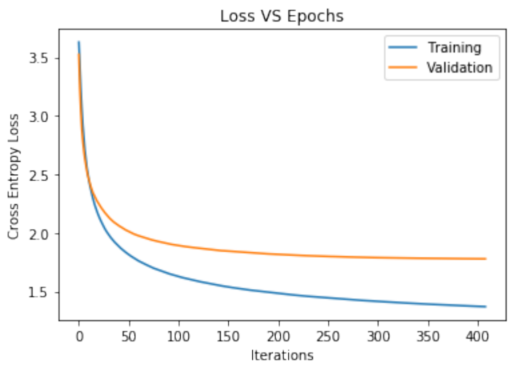
5 Dropout

Dropout is used as a means of regularization in the network and prevent overfitting on the data. It helps ensure that the network learns multiple independent and robust features of the data, such that reliance on any particular neuron for a feature is reduced.

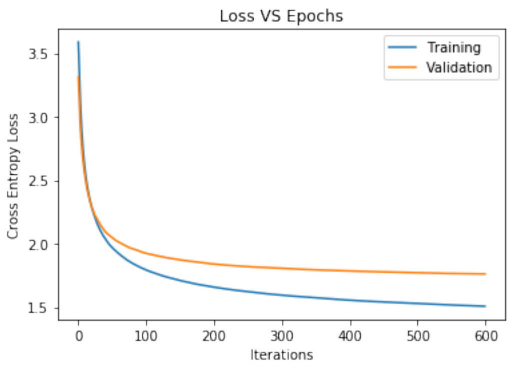
In a network with 2 hidden layers, we experimented with using dropout on the weights between the 2 hidden layers and between the last hidden layer and output. Multiple values of dropout were tried: [0.1, 0.2, 0.3]. The plots are given in Figure 4.



(a) p=0.1



(b) p=0.2



(c) p=0.3

Figure 4: Plot of training/validation loss with dropout using LSTM cells

Early stopping, with increasing loss for 3 epochs on validation set, was used for convergence. Increasing the dropout rate increases the number of training epochs. Training converges in 361, 405, and 600 epochs for dropout in [0.1, 0.2, 0.3] respectively. The training loss produced was [1.3864, 1.4022, 1.5063] and validation loss was [1.7912, 1.7495, 1.7608].

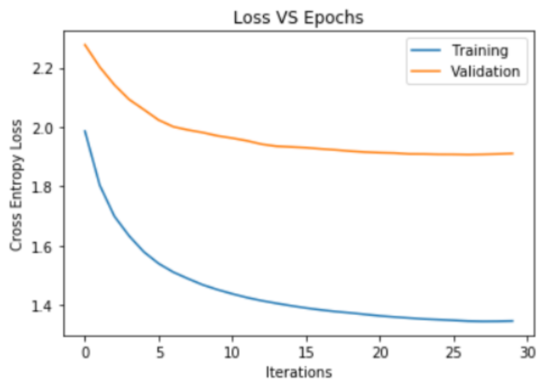
As we increase dropout rate, the training loss increases (at convergence) since the model is less prone to overfitting. For dropout=0.2, the validation loss is lower compared to dropout=0.1, while for dropout=0.3, it increases again. This implies that the model might be underfitting at dropout=0.3.

The notes for music samples produced for different values of dropout have also been added to the appendix. However, to our untrained ears, the music pieces generate did not sound drastically different. The difference with using dropout was more visible in the convergence speed and training and validation losses.

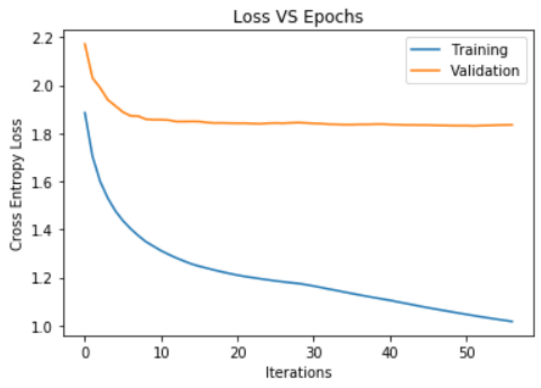
5.1 GRU unit based RNN with Dropout

Model Architecture and Training We experimented with RNN containing 2 GRU layers with 200 hidden units followed by a linear layer with outputs equal to unique characters in the dataset. Dropout was used between the hidden layers as a regularizer and the optimizer used was Adam, with learning rate 0.001

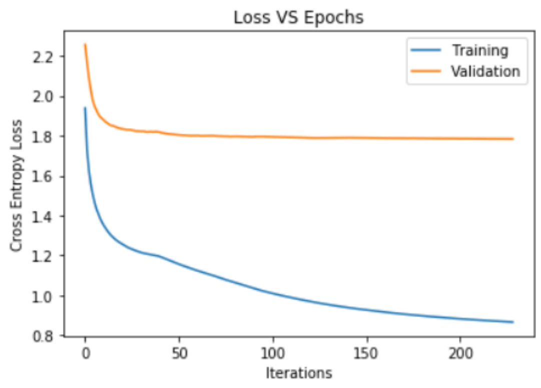
The training, validation set was constructed by doing a 20% split on the total number of tunes available. During each epoch of training we go through all tunes in the training set. From each training tune we generate a batch of continuous sequences of length 30 starting from a randomly selected start point. This batch is fed sequentially to the RNN while conserving the hidden state while training the same tune. After training on one tune is done, we reset the hidden state and train on the other tune, so on and so forth. Training is done till we see and increase in validation loss, which is used as an early stopping criteria.[the definition of epochs used in this section is slightly different from what is used elsewhere in the paper, hence the convergence epochs are on a different scale.



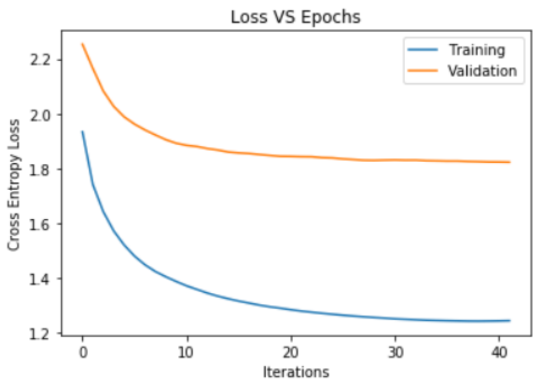
(a) vanilla



(b) p=0.1



(c) p=0.2



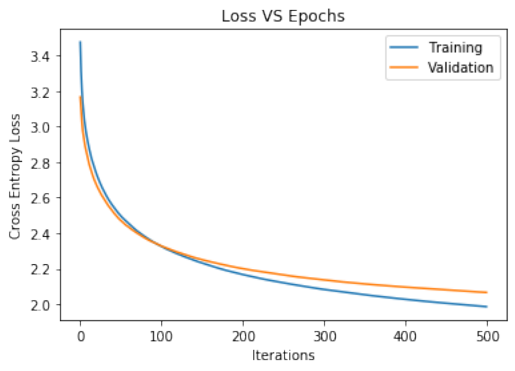
(d) p=0.3

Figure 5: Training/Validation loss vs Epochs for GRU with Dropout

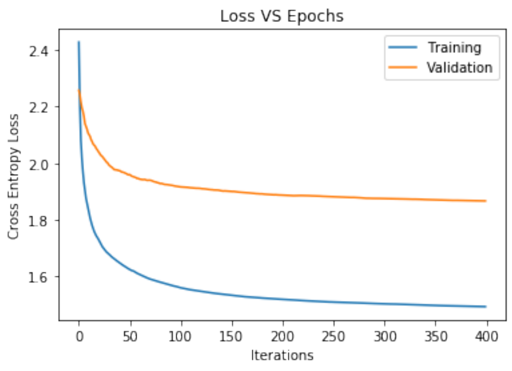
Discussion Introducing dropout in the vanilla version of the 2 layer network helped the model generalize better to the validation set as evidenced by Figure 5. We tried 3 values for dropout p=0.1, 0.2, 0.3 from which p=0.2 performed the best in terms of reducing the validation loss. With no dropout we reach a validation loss of 1.907, whereas with dropout this number reduces to 1.783 for p=0.2. The downside of introduction of dropout was that it increased the number of epochs required for convergence by some amount as evidenced by figures 5. We expected the music generated by model with better validation loss to be of a higher quality, but the music generated at all tried out dropout values seemed to have almost similar undertones. The model with p=0.2 gave a slightly more melodious music than the other two values of p, but the judgement is very subjective.

Generative models like this are quite difficult to evaluate, using validation loss does not ensure that the music generated will always be better. For same loss value we can get 2 types of samples, one with coherent music and another without.

6 Optimizer Evaluation



(a) Adagrad Optimizer



(b) RMSProp Optimizer

Figure 6: Plot of training/validation loss vs epochs with different optimizer

We experimented our model on Adagrad and RMSprop optimizers and observed slow training speed on both the optimizers as compared to Adam Optimizer. Adagrad performed worst among the three, reaching epoch limit and achieving worst training and validation loss among all experiments. Besides, Adagrad's output in terms of the tunes generated is not good. It generates long repetitive sequences. RMS on the other hand does produce good tunes, but again not as well as Adam.

RMSProp performed comparatively better than Adagrad, converging in 400 epochs and achieving 1.4939 training loss and 1.8669 validation loss.

Sampling Technique	Layers	H. units	Optimizer	Epochs	Training Loss	Validation Loss
Every Tune	2	150	Adam	151	1.2909	1.7609
Every Tune	4	150	Adam	351	1.3508	1.8663
Every Tune	1	50	Adam	491	1.5076	1.8032
Every Tune	1	75	Adam	491	1.3851	1.7673
Every Tune	1	100	Adam	261	1.3906	1.7839
Every Tune	**1**	**150**	**Adam**	**281**	**1.2767**	**1.7413**
Every Tune	1	150	Adagrad	500	1.9858	2.0661
Every Tune	1	150	RMSProp	400	1.4939	1.8669
Every Tune + Curriculum	2	100	Adam	330	1.4398	1.8940

Table 1: Performance compilation on various experiments

7 Feature Evaluation

An interesting way to look at how the LSTM is able to predict the music sequence is to look at each of the hidden unit activations during a forward propagation as shown in Figure 7. The color of the hidden unit corresponds to its activation. In Figure 7, blue corresponds to the most active unit and red corresponds to an least active unit. Most of these hidden units follow some pattern in their activation which helps in producing music.

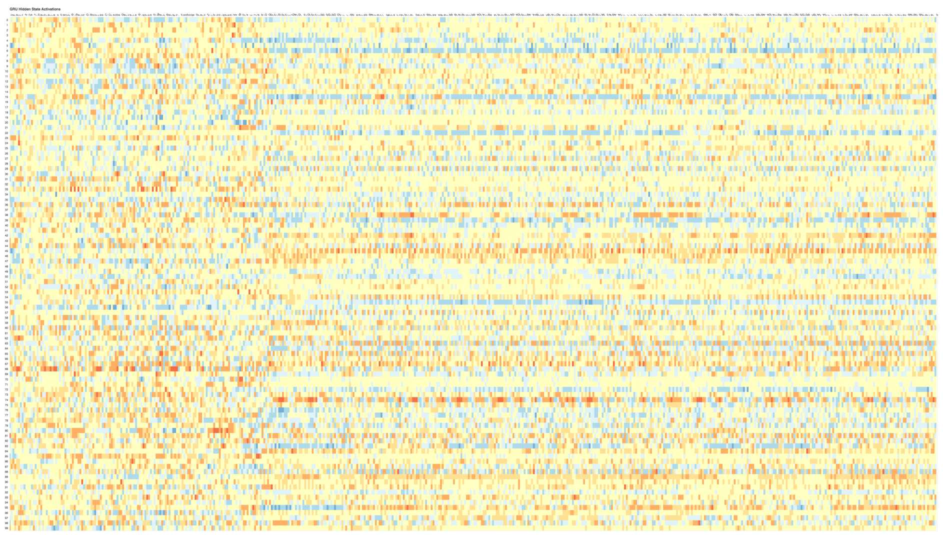


Figure 7: Activations of all hidden units

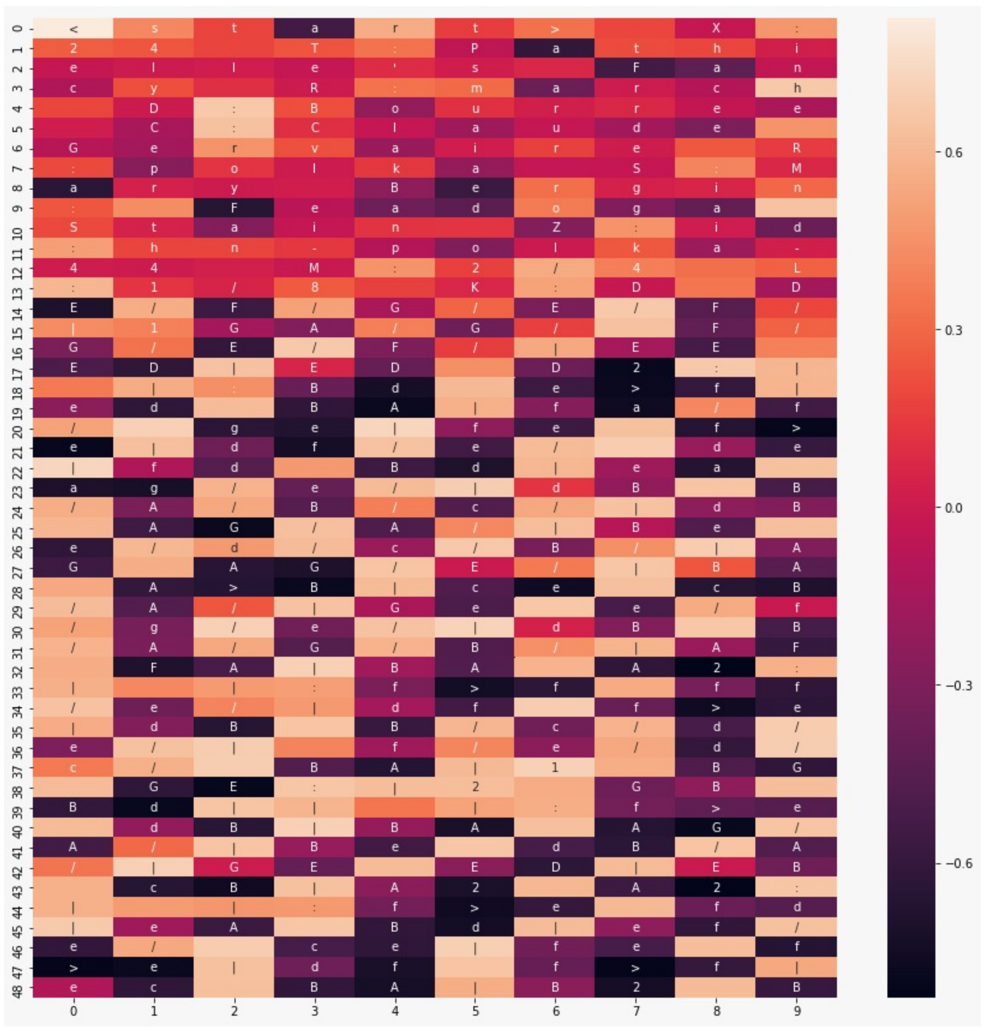


Figure 8: Activation of a hidden unit firing for spaces, newlines, | and /

The hidden unit in Figure 8 is fired for spaces, new lines, | and /. The lighter the colour in Figure 8, the higher the activation. This particular hidden unit is more active for the header of the tune and is relatively dormant in the rest of the tune except for characters mentioned above.

8 Conclusion

A recurrent neural network with LSTM or GRU units is capable of learning patterns and long term relationships between notes in a musical tune. LSTM units mitigate the vanishing gradients by allowing gradiets to flow back in time. This is proven by the music generated by the networks trained in this paper.

9 Learning Outcomes

Music composition is hard to do for a musically challenged layman, but being technically sound has its perks. In this exercise we learnt how to train a recurrent neural network to generate music. We experimented with different techniques of sampling and generating music and learnt about the tradeoffs involved in each of them.

10 Contributions

This assignment is a result of equal contributions from all the authors. Everyone pitched in to brainstorm on various points discussed in the report. All implemented their own versions of LSTM, which was later combined into a single source code to set up baselines for all other experiments. The work on different experiments was similarly divided equally. We request the grader to award equal points to all team members.

References

Following references were used in the assignment

[1]https://github.com/euler16/CharRNN

11 Appendix

11.1 Music samples for different temperatures

11.1.1 T=0.5

```
<start>
X:16
T:John Ryand (1589)
O:France
A:Provence
C:Trad Bon the Wood
R:jig
H:Also played in Gnoffiellan (1670-1738)
H:Also played in G
K:C
Gc BA|1 G2 G2:|2 B2 B2||
|:"B"B2 BA GB|"D"A2 A2 AB|"G"dd B2|d2 de fg|"G"d2 Bd cB|"F"A2 AB AG|"D"A2
     A2 dA|"G"dd dB AB|"C"c2 B2|"G"Bc BA|"G"G2 G2|"C"c2 c2|"G"dB GB|"G"dd
     dB|"C"c2 c2|"G"dB Bd|"C"e/2d/2c/2e/2 " b/a/g/f/|ed cB|"G"BG G2|"D"A2
     AB AG|"D"A2 A2 AB|"D"A2 A2|"C"c2 e2|"G"d2 z2|]
<end>
```

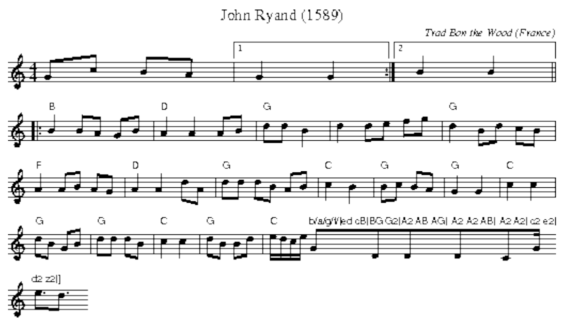

John Ryand (1589)
Trad Bon the Wood (France)
1
2
B
D
G
G
F
D
G
C
G
G
C
G
G
C
G
C
b/a/g/f/|ed cB|BG G2|A2 AB AG| A2 A2 AB| A2 A2| c2 e2|
d2 z2|]

Figure 9: Music representation for T=0.5

```
<start>
X:47
T:Farandole de Cannesters of The Mary’s
R:polka
D:Mairys of Tualy: Welsposest Stor #6
Z:id:hn-reel-36
M:C|
K:G
BA|BG~G2 AcBA|GABc dBAF|GEED E2:|
|:ef|ee ef/g/|af f>e|fe ce/f/|gb ag|fe e/f/g/e/|dB BA|FA AB|dB gB|1 BA AB
    :|2 BA AB/c/||
|:dB cB|AB ce|fe ce/c/|BA A2:|
|:Af fe|fd ef/g/|af f>f|ed Be|f/g/f/e/ dB|A>B cA|BA AB/c/|dB BA/B/|dB AF|
    DF/F/ GB|AB/c/ ec|fA cA|fA f>f|ed Be|fd ed/c/|BA AB|cB/c/ed:|
P:Version 2:
|:~g3 edB|def a2g|afd age|dBG ~g3|ega bge|~a3 fdd|efg eag|fed efd|ede gab
    |fed e2g|fed efg|afd efg|fed efe|dfe dBA|1 GED G2D:|2 GEE E2D||
|:efe edB|d2e fed|faa aga|bag e2e|ded dBA|efe edB|ded dBd|~g3 gfg|fed cBA
    |def ged:|
<end>
```

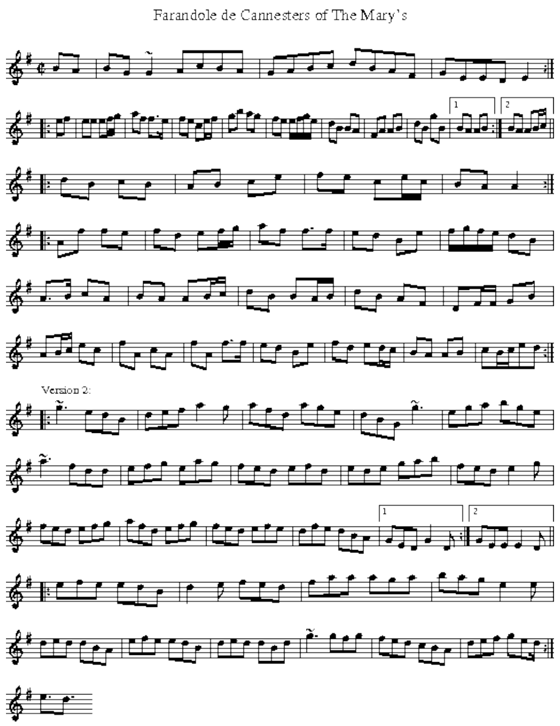


Figure 10: Music representation for T=0.5

11.1.2 T=1

```
<start>
X:1
T:Bourree
Z:id:hn-polka-92
M:2/4
L:1/8
K:G
GB ed/B/|AB e/c/A|1 Bd d>e:|2 cA AG||
|:ce ce|fa ag/f/|ec B>g|fa a2/a/|ba bc|ef cA|de/f/ ge|dB GB|dB cB/c/|d2 d
    >e|
fa ba g/a/f/g/|a>g fa|ge fd|ed dc|BA FA|AF FA|GF GB/d/|de d2|dB gB/c/|de
    f>e|df AB|cB cd/c/|BG G2:|
ed ef|ga ge|dB/A/ Bd|eA/c/ ed|c/B/A Bc|dg dB|ef dB|GB d>B|cg ec|1 Bd ec|
    d2 d2:|
|:de/f/ gf|ef gg|fe cB|Af fe|gf ec|ef/e/ dc|Bc d>c|BG Ed|BG B/c/d|e/f/e/c
    / BA|GA Bc|d2 de|fd fd|eA f>e|dB cA|de dB/c/|dc dc/B/|AG FD|B>A Bc|D2
     FA|eG/B/ AF|AF GE|DG dB/c/|dA GA|Bd ed/e/|fd eA|f/A/A fe/f/|b2 c’C|1
     "G"d2 dc BA|\
"G"dd/d/ dB Gd|"G"BA cA|1 AG G3:|2
Gd/B/ cB|DB/c/ dc|BA FA|cd/c/ A2|fd de/f/|gf e/d/c/B/|AF EF|ED D2:|
<end>
```

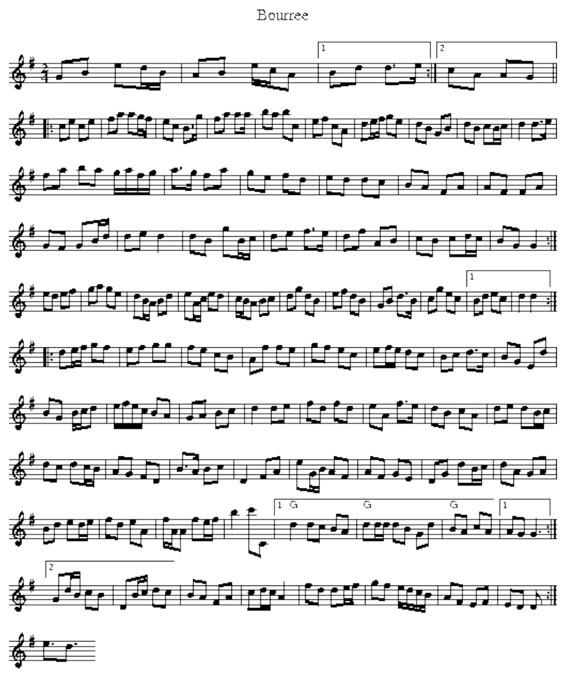


Figure 11: Music representation for T=1

```
<start>
X:24
K:G
BA GB|cddc BA|de/f/ ge|cA Ac|ec B2 c2|
e2 fe de|fd cd|ec d2|eB d>B|AB/c/ dB|A2 AB/c/|de e/f/g/e/:|2 ce BB||
|:zf/g/ fe|dg eg/e/|df/d/ dc|Bd cA|Ba e2|fe ga| ad fd|ea/c/ (e/d/)b/a/ fe
    |fa af/a/|ba fe|f2 fd|ec e>f|eg ge|dB AB|d2 dB/c/:|2 AG GB/c/||
|:dB cA/B/|Ae/f/ gE|FA BB|AB/c/ dc|BA cA|Bc de|fd ef|g>g dB|cA A2:|2 (3g/
    f/e/ de|fa e/f/e/d/:|2 ef/e/ de|fg ed|c/B/A ec|d2 d2:|
P:Version "C"d2B2 G2Bd|
[1 bgbg ageg|d2 (3Bcd ecde||
|:~f3d BdBd|gedB GED2|
dBdB d2 (3def|ge~g2 edBG|AF (3AGF G2:|
|:eg|ed d^c|dB ce|dB cA|dc Bc|BA A2:|
|:ce ec/e/|fd Ad/f/|fe ce/f/|gg fe|gf ga|ba ge|dB BA|BA F2|gf/a/ ge|fd f>
    e|df AA/c/|1 B2 g2:|2 cA A>B||
|:ce/f/ ed|ef ge|fd fd|ed f/a/f|ed c/B/A/d/|f/e/d ee|fd eg/f/|eb ef|g2 ge
    |dB BA/B/|dB GB/d/|af/e/ ce|af fe|e/e/d/B/ cd/c/|BA FE|1 E2 DE:|2 D2
    DE||
|:~G3 AB/c/|dg fd|ed d/c/B/A/|
D2 B2|ga/g/ ec|fd f>e|dB cd|ed cB|Ad c/B/A|GA Bc|BA F2|D2 FA|D2 FA|Be/e/
    fe/c/|cB/c/ ec|dB dc:|
<end>
```

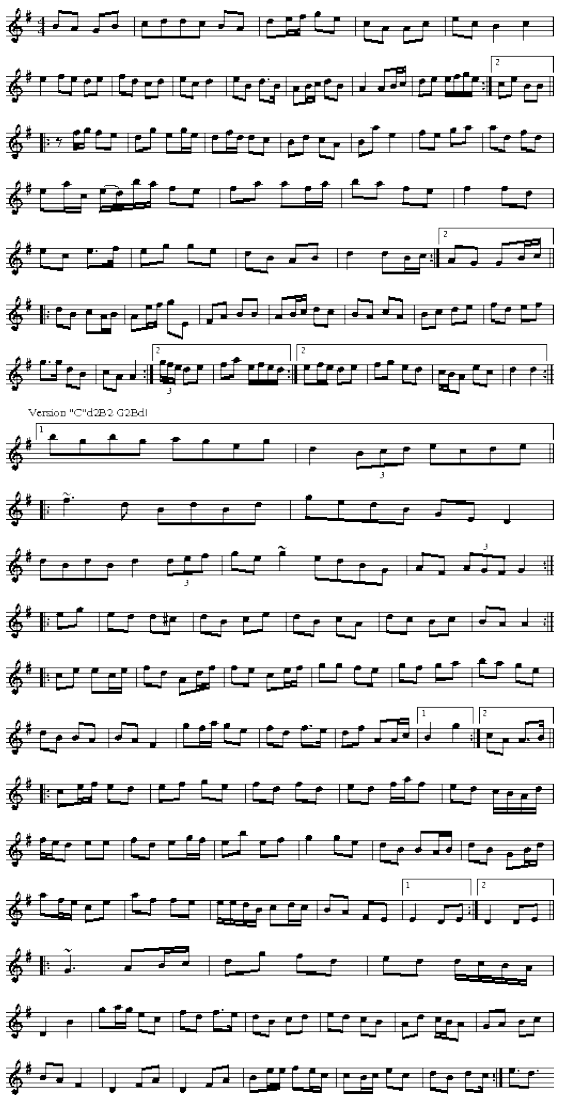


Figure 12: Music representation for T=1

11.1.3 T=2

```
<start>in: CMul’a
T:Walss Dame J0, I Toucydrel[Q:1/46o"ao]cor.ck
H:miloe,˜Zhy The. Cieergyd"An^KBAAInay:
G2B B/^A/c/gh:|2 GA A>G | GD ˜F3 |:|
| g>a | eg |
.f2b>
Jin’s
. c=_)ucfidg...a
P:Ix[Q:moj6#-d2ig
dd,d AfB|g2 eic.|g e2 G2|e4 cBz|G f/e/d (>9 I.vor
R:ahchehoy’s
R:honnaps
O:Bradley Polka"-lh/|- edce | g2ca c/e/fB=G |[g BA cB|AB Gd/e/|fe fe|a>f
    f/B/df|ae f/e/d/c/|=BB BA|A>B FB|
a4 b>ci| (ba/2b/b/b/E| "(m!’h(eHn!u[K:Aa+got"R+ranjhe
M:2/4
L:#3
I]|:gb|a/c Ab ge|fg EcleG=B|
ADAd >F^Fd|b2fA f>BA|Bbf eg^f|B>f ed|cg G2|
ed/G/ (c/B/A/c> a|rit et/fe-win, /ipi)#55
D:Cate3,, On \2BfirhICrephenol.?y="3dds:des"C
z <orc.cathay the4,
W:o>ov n6] |B2B2A2B2 (4-g,U 1E2(0k.c) ztaMtt
d)AG GGBd |
<end>


<start>"Teft"F>e g/f/|el ed ce/e/|
|:Ad B |AA/E/ FG|AB h/c/G/A/:BCRcd2.
Z:id:hn-r/z
H:Also Illka
R:Scottish: Llwa Hiltsuua
Z:un in Stoztladok, TcEtofe"3
C:Vessiss's
T:cit Cobon
R:polka
D:Nahury Blan, Tute Es| A. S.
Z:id.
R:radhe McNapait BEdhRaptek Cowns, The
T:EcitgighaFFr?+C7
Z:als Lotch Ec.ny, Bras (:593d vori".:P-70-06-2tif 1933
D:Jary?e (Cagle=sin eng>, #Thrrrg.0: The The|ec Hltle.
.:Bf.Cb |B2GBGB ed^cd|fa (3f=eA !a.g/|+ats"{cd]F/A/G/|E2 E2::|:(BlA8zGB\?
    ecRauD
was d arth, The
T:S\'roishna\eB AFCh
(~B3|~E3 GED|
Eede dFGD/B:|2 Bdce2f|ged^cea|=aab2 E=cGz||[BAgb'a b:hrinoverit:go+c_kneq
    !1{aessmeCt2FAly Farey@on": 'liis ~ord, she wai
",2b)(phred'sh'an:
T:MoDvas
D:DTGRELIm6it2"Crku't" nf
D:matt Moderny"E.z4: Iotwise Noiss. Zne Ramy'n
R:piceSiadk,
R:pel
I:af c/^g/ac) _[4nuxb2A0B> F|]
P:Variations:stinfot@Oufe."C:mec
O:Francy
Z:b_a he, The
R:reelid:hn
D:Flyash:uncen CECLFEG
K:EG"AB gz "F"B/F/|la" "Am"c2 e2A/f/:|fd ec cg|a a a|1
Ff ecA f z)z|
^e|"Gm!z)2z"AB^cA|\
e2d
(ffppe+A,2ar:"^c21g^=ef|2degec|c=^c^e|{ggf^c|B2AA:|
|:FFIfa eGAe|fefb ag|fdcA F2DG:|
V:
(3ef=c fe|d2 F2|fd fe|df af|g>b cB|AB/c/ (dc/B/|Ae e/d/e/f/|gFA4:?"B/A/g/
    d/|=.G>FGB|A>B cd|c2 ^fe-fg/c> AG|c3 dcd| ef/2B/oae ::LH2bh:Aqgieh=&
    Cooveifgsy
R:Totry, Sweud, 94 (d Polka, #118ri-ati"dus.gasse:~e,2nd !2z2G EEE|BGG ~
    E3F/G/|FE E>c|BE AA/G/|FD D2:|
<end>
```

11.2 Music samples for different dropouts

11.2.1 p=0.1

```
<start>
X:60
T:La lass of the Word, The
R:reel
```

```
D:Mary Bergin: Feadoga Stains 1.
Z:id:hn-reel-35
M:C|
K:G
Bc|dB A2|BG AG|FA AB|cB AB|cB AG|EE E2:|
|:ef/e/ dB|BA A>B|cB BA/B/|dB BA|1 BG G>A:|2 E2 E2||
|:de f>e|dB BA|B/c/d e2|fa ef|ef/e/ dB|BA AB|cA B/c/d|e>f ef/e/|dB BA|1
    BG GB/A/:|2 BG G2||
|:ef/e/ dB|BA A>B|ce e>f|ed Bd|e>f ec|BA Bc|d>e dB|AF FE|FE FE|DF AF|AB
    cE|ED D2:|
|:A>B AG|FA A2:|
|:e>f ec|BA Bc|de f>f|eA cB|1 BA A2:|2 A2 AB||
|:de/f/ ge|dB B/c/d|e2 d>e|fe e>d|ed cB|AF FE|D2 D2:|
|:d2 d>e|fd fe|fe d>e|fe fa|gf ed|gf/e/ dc|BA FA|B/A/B/c/ dB|BA B>A|Bc BA
    |B2 B/A/B/c/|dB BA/B/|dB BA|1 BG G>A:|2 G2 G>A||
<end>
```

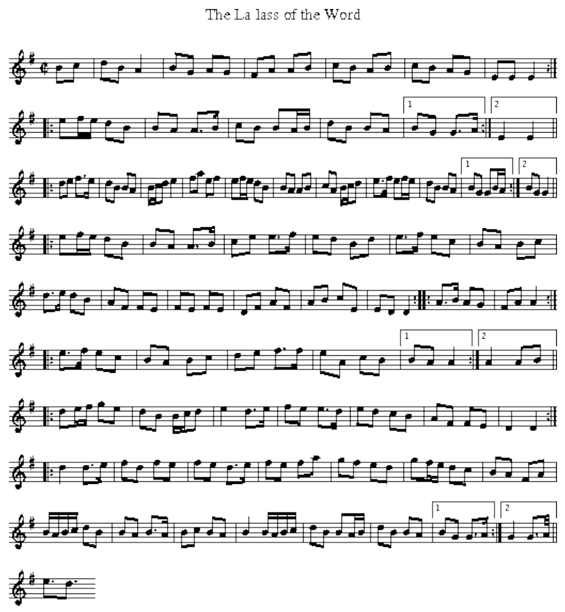


Figure 13: Music sample for p=0.1

11.2.2 p=0.2

```
<start>
X: 4
T:Mart to Sheer Moring Deans in G
Z:id:hn-polka-23
M:2/4
L:1/8
K:D
dc BA|G2 A>B|AG ED|D2 G2:|
|:e>d ed|ed d>e|dB BA|d>B AG|FE D2:|
|:f>e de|fe dB|AB ce|fe fe|d2 de|fe d>e|fe dB|A2 AB|cB GB|AF EF|AB cB|A2
    FA|G2 G2:|
|:f>f fe|fe fe|de fe|dc BA|AG ED|EF GG|AB AB|AG ED|D>E EF|A2 GE|D2 D2:|
|:B>B dB|AB AG/F/|EF AB|G2 A2:|
|:ef/e/ de|fd ed|BA AB|de fe|de fe|de fe/f/|ge d>e|dB cB|A>B AG|FA FA|cB
    AB|cB AB|ce fe|dA d>B|A2 AG|G2 G2:|
|:f>e dB|AB AB|dB BA|BA AB|AF GE|D2 D2:|
|:f>g ad|ed ef|gf gf|ed ce|fe d>c|BA AB|ce ef|ed ce|fe fe|fe dA|BA AB|Ad
    de|fa af|ec d>e|fd ef|ed cA|BA GE|G2 AG|FA AB|AF FE|D>E DE|FA FA|de
    fe|dd dB|AB/A/ Bc|de f>e|dB AB|AF DA|BA GA|BA BG|A>B cA|FA G>A|BA GA|
    BA GA|BA BA|BA BA|GB BA|BA GB|AB AB|AB cB|AB/A/ GE|E2 D2:|
|:a>g fa|gf ed|cA cB|AB ce|fe ef/e/|dB A2|GA BA|GA GA|Bd dB|A>G FG|AB AB|
    ce ef|fe fe|d2 de|fd dB|AG FA|BA GA|BA BA|GB GB|AB cA|
G2 AG|FA G2:|
|:ef/e/ dB|A2 AB|cd ed|cA AB|AF GB|AB cB|AF DE|DE GE|AB A2|BA GA|BA BA|GA
     BA|GA AB|A2 AB|AB BG|AB cB|A2 G>A|BA GB|AF ED|EF GB|AF EF|AB AG|FA
    AB|cA FA|GB BA|BA AB|A2 AB|AF ED|FE D2:|
|:fe d>e|fa ag|fe ef|ed ce|fe dB|AF FA|A2 A2|AB/A/ GE|D>F EE|D>E EF|AB AB
    |cA AB|AB AB|cA AB|cB AB|AF FE|D2 GE|D2 D2:|
|:f2 d2|df ed|c/B/A GB|AF AB|cB AB|AB AB|AB GB|AF FE|D2 D2:|
|:f>g fe|de fg|af f>e|dA GA|BA GA|BA AB|cA AB|AG AB|cB AB|cB AB|AF AB|A>G
     FA|DF AB|cA BA|GB BA|BA G>A|BA GB|AG FA|GB B/c/d/e/|dB A2|Bc BA|GA
    BA|GB GB/A/|1 E2 E2:|2 GA G2||
|:ef ge|de fe|de fe|dc BA|GA B/c/d/e/|dB BA|BA G2|BA BA|1 GF G>G:|2 G2 G2
    :|2 BA GA||
<end>
```

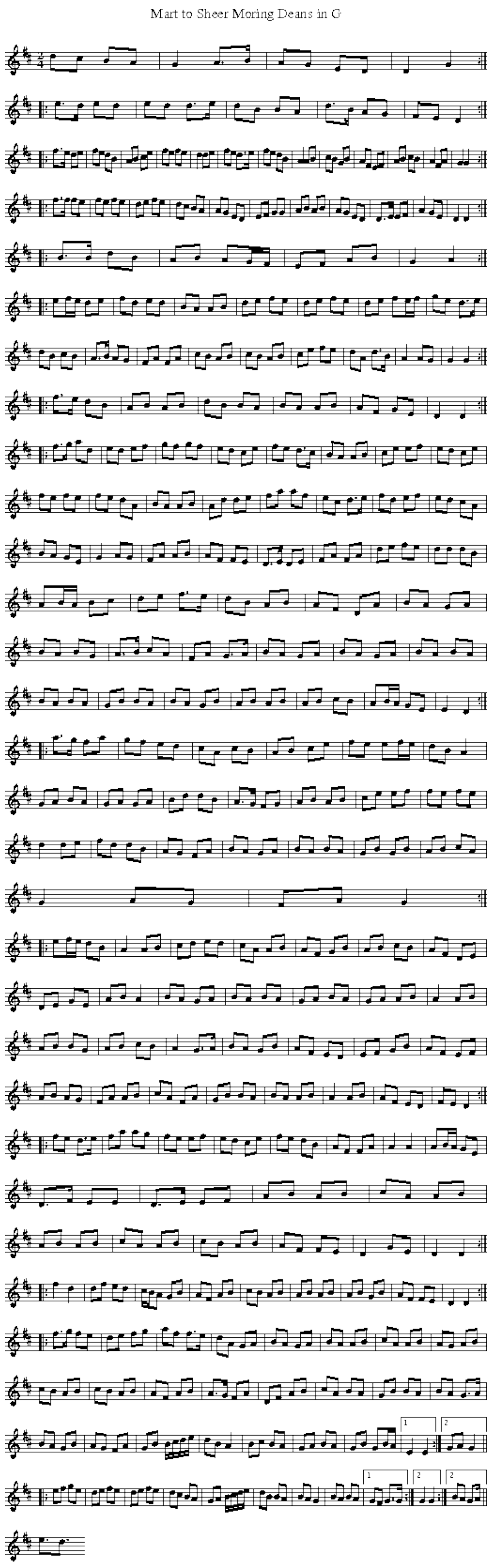


Figure 14: Music sample for p=0.2

11.2.3 p=0.3

```
<start>
X:15
T:O'Carolan's Wilthe Sing, The
R:reel
Z:id:hn-polka-94
M:2/4
L:1/8
K:D
AF FD|E2 E2:|
|:A>B cB|AF FA|A2 AB|cB/A/ GE|DF DF|AF DF|AF DE|FA D>D|FA DE/F/|GB B/A/B/
    c/|dB B/A/B/c/|dB BA|BA Bc|d>e dB|AF DF|AF FE|DF AB/A/|GB BA|G2 GB|d2
     d/e/d/B/|AF FA|GB/A/ GE|EF E2:|
|:fe e/f/g/e/|dB AB/A/|GE EF/G/|AF D2:|
|:eg fe|df/e/ dB|BA AB|ed/B/ AF|AF FA|d2 dB|AF AB|cd/e/ fe|dB AB|ce ed|cA
     BA|ed e>d|ef ed|BA AB/c/|df Ac|BA BA|GA/G/ FE|1 DE DE:|2 AG G2||
|:ef/e/ dB|AF AF/A/|BA BA|BA Bc|d>e fd|ed BA|de/f/ gf|ed cA|BA AG/A/|BA
    AA|BA GE|DF AB|cB AB|cB/A/ GF|AB/A/ GF|AB A2|B>A Bd|ef e/d/c/B/|AF D2
    :|
|:ed d>e|dB B>A|Bd ed|Bd e>d|ef ed|e/f/e/d/ ed|ee fd|ef ed|cB BA|BG Gd/d
    /|BA AA|BA A2:|
|:fe/f/ fe|fd cA|BA AB|cB cB|A>B AF|FE D2:|
|:d2 d>e|dB A2|ef/e/ dB|BA A2|d>e fd|ed ed|ef ed|ea ag|fe ed|cB AB|cA AB|
    cB AB|cB/A/ GE|FE DF|AF FA|BA BA|GE EF|GB/d/ ec|1 BA A>A:|2 dB A2||
|:fe fe|d>e fd|ef/e/ dB|BA AA|B/c/d/e/ fe|dB BA|d>e dB|AF FA|BA AB/A/|GB
    B/A/B/c/|BA AB/A/|GE EF|GF GB|cB A2|BA Bc|d2 dB|AF EF|1 D2 DE:|2 D2
    D2||
|:de f>e|dB cA|BA AF|GF/G/ AB|A/G/F/E/ D2:|
|:d2 dB/A/|GE ED|EB B/A/B/c/|AB AB|cB/A/ GE|DF D2:|
|:ef/e/ dB|AF AB|cB/A/ GE|EF GB|AB cA|BA Bc|de fe|d>e dc|BA BA|BA AB|A>B
    cA|BA AB|AF AB|cB BA|BA AG|FA AB|cB AB|cB/A/ GE|DF AB/c/|dB BA|GB/B/
    AB|AF EF|DF A>B|AF AB|cB AG|AB/A/ GE|DE FA|AB/A/ GE|EF FA|BA AB|cB/A/
     GA|BA AB|cB/A/ GE|DE FE|D2 DB|AF/A/ Bd|AF DF|AB/A/ GF|E>E Ec|BA AF/A
    /|dB BA|Bd dB/A/|GE EF|DF AB/A/|GE ED|E2 EF|D2 DE|D2 DE|D2 DE|D2 D2:|
|:cA AG/A/|BA AB/A/|GE EF|GE EF|AB/A/ GE|1 DE DE:|2 D2 D2||
<end>
```

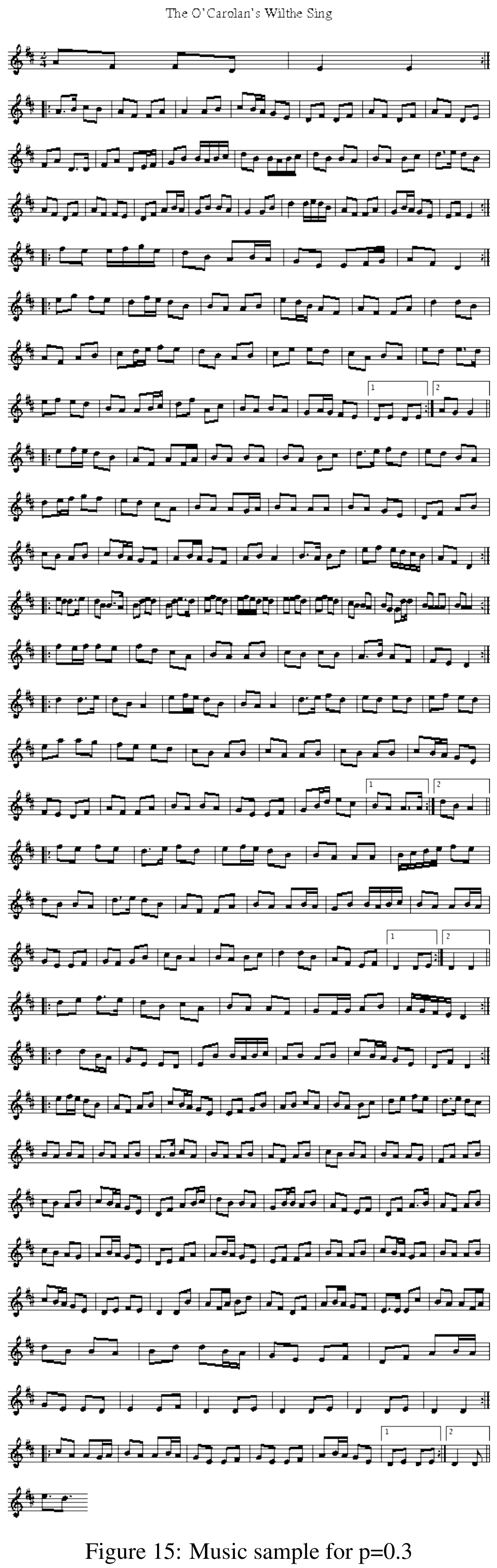


Figure 15: Music sample for p=0.3