CSE 253: Neural Networks and Pattern Recognition Winter 2017
Generating Music with Recurrent Networks

Swapnil Taneja (PID: A53219137)
Department of Computer Science and Engineering
University of California, San Diego
San Diego, CA 92093
swtaneja@eng.ucsd.edu

Saksham Sharma (PID: A53220021)
Department of Computer Science and Engineering
University of California, San Diego
San Diego, CA 92093
sas111@eng.ucsd.edu

Kriti Aggarwal (PID: A53214465)
Department of Computer Science and Engineering
University of California, San Diego
San Diego, CA 92093
kriti@eng.ucsd.edu

Prahal Arora (PID: A53219500)
Department of Computer Science and Engineering
University of California, San Diego
San Diego, CA 92093
prarora@eng.ucsd.edu

Saicharan Duppati (PID: A53221873)
Department of Computer Science and Engineering
University of California, San Diego
San Diego, CA 92093
sduppati@eng.ucsd.edu

Abstract

In this assignment we use the trained RNN model exploiting the temporal relationships of the characters present in the sequence. We train the model with the target as the next character in the sequence for 200 epochs. The RNN model automagically recognizes the temporal relationships of the characters in the sequence and generates features that can be used for inference predictions. The best part of the training is that we do not pre-process the data in any form. We even let the $< start >$ and $< end >$ tags to be present in the file. Our model can even store this relationship , thereby ensuring proper formats of our target output file. Needless to say , it can even generate proper headers for the music file.

Introduction

In this paper we explore various configurations in RNNs for the purpose of generating sequences of music. The Recurrent Neural Network has a simple sequential model with SimpleRNN as the first layer of the model. We then add a dense layer with softmax activation on top of it . We start with a batch of sequence having length 25. The core reason that recurrent nets are more exciting is that they allow us to operate over sequences of vectors: Sequences in the input, the output, or in the most general case both.See figure 1 .The difference between $return\_sequence = True$ and $return\_sequence = False$ is that in the first case the network behaves as in the 5th illustration (second many to many) and in the latter it behaves as the 3rd, many to one. We proceed with $return\_sequence = true$ as we wish to exploit the batch size with keras functionalities.

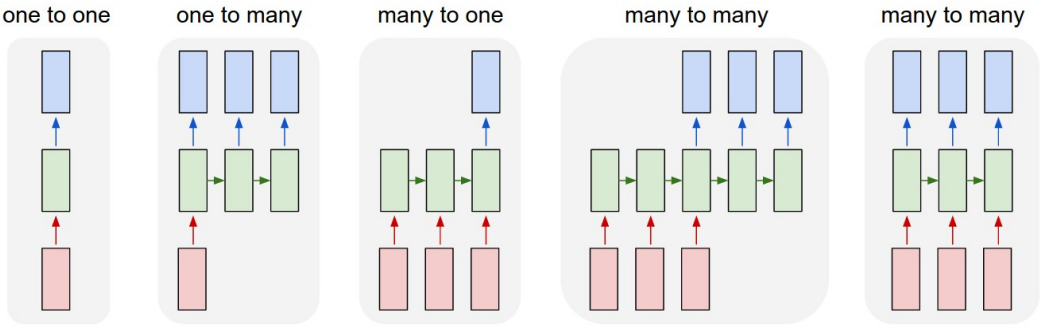

Figure 1: Various RNN configurations

a. Temperature

The ABC notation and the music representation generated from http://mandolintab.net/abcconverter.php are as follows:

- Temperature = 1
  The ABC notation for the sample 1 is shown in Figure 2
  The music representation for the sample 1 is shown in Figure 3
  The ABC notation for the sample 2 is shown in Figure 4
  The music representation for the sample 2 is shown in Figure 5

- Temperature = 2
  The ABC notation for the sample 1 is shown in Figure 6
  The music representation for the sample 1 is shown in Figure 7
  The ABC notation for the sample 2 is shown in Figure 8
  The music representation for the sample 2 is shown in Figure 9

- Temperature = 0.5
  The ABC notation for the sample 1 is shown in Figure 10
  The music representation for the sample 1 is shown in Figure 11
  The ABC notation for the sample 2 is shown in Figure 12
  The music representation for the sample 2 is shown in Figure 13

The tunes have been uploaded in midi format on vocareum (see folder PartA).

Discussion of Results:

- We generated 2 samples for each of the temperature parameter and observed that the samples generated for T = 0.5 were melodious and had the sequence of musical notations without skipping of notes.

```
X: 3
T:Sheme eewdond mamin sineloh, The
R:part>
X:12
T:Hterr.
W:Scott cadr
R:harnpipe
C:A
Z:idesher'i captheat
R:hornpipe
D:Kevin Hailledc Mor i-lowa rea hertion Slanpyre mout tune Me luviervou'solmene afBosser tanna Seme-
R:Boy Po?l, bian: O'Dany doom, The Mc Dumes (FDD Ed)F|E2A2F2 G2A2E2 F2DFAB | A2Ac ABdc/8=13b2 a2aff|ad2c ecde|fded ~g3a|g2gf dBBG|(3ccA la gandan
R:nillime
Q:38
T:Vipe de Arols & Cranat escicso #418
Z:id:hn-jig-29
M:6/8
K:D
BA | BAGB G3 | BGB c2z | e3 A2z | ABA G2E:|2 B2G C2A |1 B,2G,2 G,8::|
B4 | | Ed/2 ed ce |
f3gg2f2 | edcdBBG | B2 "F7"AG|"[D3c/e/d/c/|dA B/2A/2 E3|ABG AGG|G3:|
|: faa ade | g3 f2e |  d2+b)fa fe dc | B2 B3/2A/2 G,2|EE D2:|
A,|DED DED|1"e"c c/2e/2 | ed/B/c/B/B/B/G/ A/B/A/G/ Bd/d/ cB | +"c2m"c3B | A2B2A2 | A4Fz |]
```

Figure 2: Temperature = 1: ABC notation for Sample 1

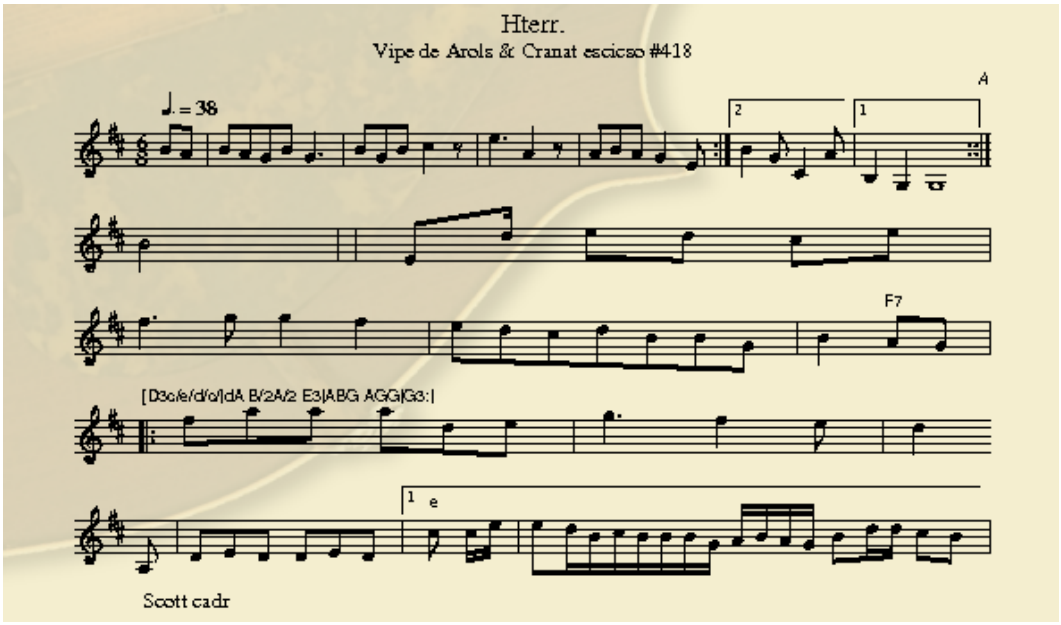

Figure 3: Temperature = 1: Music representation for Sample 1

- We also observed that as we increased the temperature, the music was slower and skipped a few musical notations.
- Also, we observed that the tempo of the song was different for different temperature parameter.

The **hyper parameters** are as follows:

- Number of neurons in the hidden layer = 100
- Optimizer used = Adam
- Loss function used = Categorical cross entropy
- Mini batch size = 25
- Number of epochs = 100

b. Train and Test loss

Figures: 14

```
X:63
T:Boly'2
Z:Trend oveus vous #25, Che)
O:Drvews dlelavin tit Mon be la
O:Raborrignd:
Bede g2g:|
|:gfaf bag|fdB c2e |1 d3B BAGB | ^F/A//F/E/G/ | BB G/B/G/g/|B3 A/A/f/e/d/2c/2 |
d2d cBF | B2d | [d2BG][BB]"C"B[B2c2B2 | "G"Bc | "Em"Bc | "G"d2 | "C"Be/d/c |
ef gf | e2e2 f2d2- | B2d2 (3cB> ||
|:  d2 GE B2 FB cz/d/| "D7"BA zB B2z2 |: c^cc ABFE | FCAF GEF,D | D,,2B,D,C, | C,CTh, The
R:adgrianl
O:France
A:Provence
C: Upolka-59
M:C|
K:G
A2 E2:|
|:ABBA B2 G2|B3B BAGA |:AGAG E2GF||
```

Figure 4: Temperature = 1: ABC notation for Sample 2

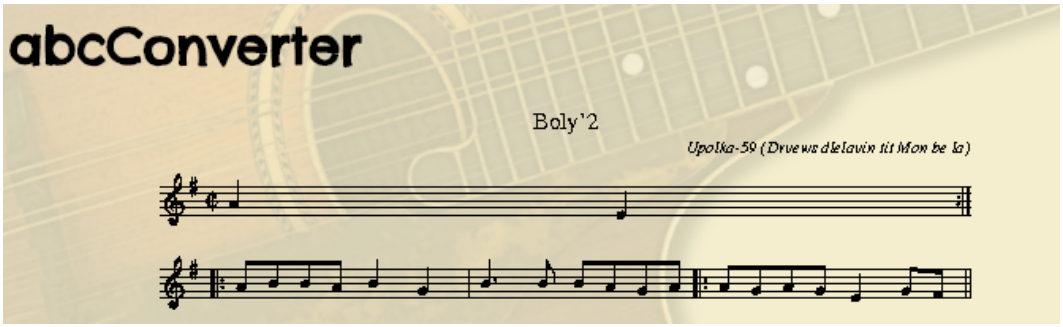

Figure 5: Temperature = 1: Music representation for Sample 2

- Figure 14 shows the variation of training and validation loss with epochs run on neural network with 100 hidden units.
- We observe that the train loss decreases with number of epochs. As expected, we see that validation loss is higher compared to training loss as at the time of validation it sees relatively new data. This reaffirms that neural network is learning on the data.
- We also observe that the validation loss roughly decreases with number of epochs, but with a slope less steeper than train loss.
- Validation loss doesn't exhibit a smooth trend like Train loss. The frequent variations in the validation loss can be explained because while calculating validation loss, we are testing on the unseen data.

c. Number of hidden neurons

Figures: 15,16

- Figure 15 shows the variation of training loss with varying number of hidden units between 50, 75, 100, 150
- Figure 16 shows the variation of validation loss with varying number of hidden units between 50, 75, 100, 150
- We observe that the training loss shows the same trend as we observed above for every choice of number of hidden units ie., training loss goes down with number of epochs

```
X:12
T:Horvik Hath
Z:id:hn-jag-2
M:2/4
M:C|
K:G
A2e dBA|FEA BAG|DAA BAB |Afc Bcd|dcA D2A|MC2|G6:|2 G2 EE|1 cA BA/F/ | G2A2G2 | ~B2B/e/g/B/|fg/e/ ee|e2 e/|a/b"G"e/2:|2 a e2 fe|da/e/ a2 a2g|g2g|ffg|Daf|eAB|cdc|defe d2
ffd|BE~D2 D2:|
|:ddBA G4:|
|:gf|dBAB dFAF|(D2E2 A4 :|2 A4:|
|:~A2BA GG:|2 E2 : F2DE|D3 | d2c def | (e2a>g f3| c2c2c2) z2 |]
```

Figure 6: Temperature = 2: ABC notation for Sample 1

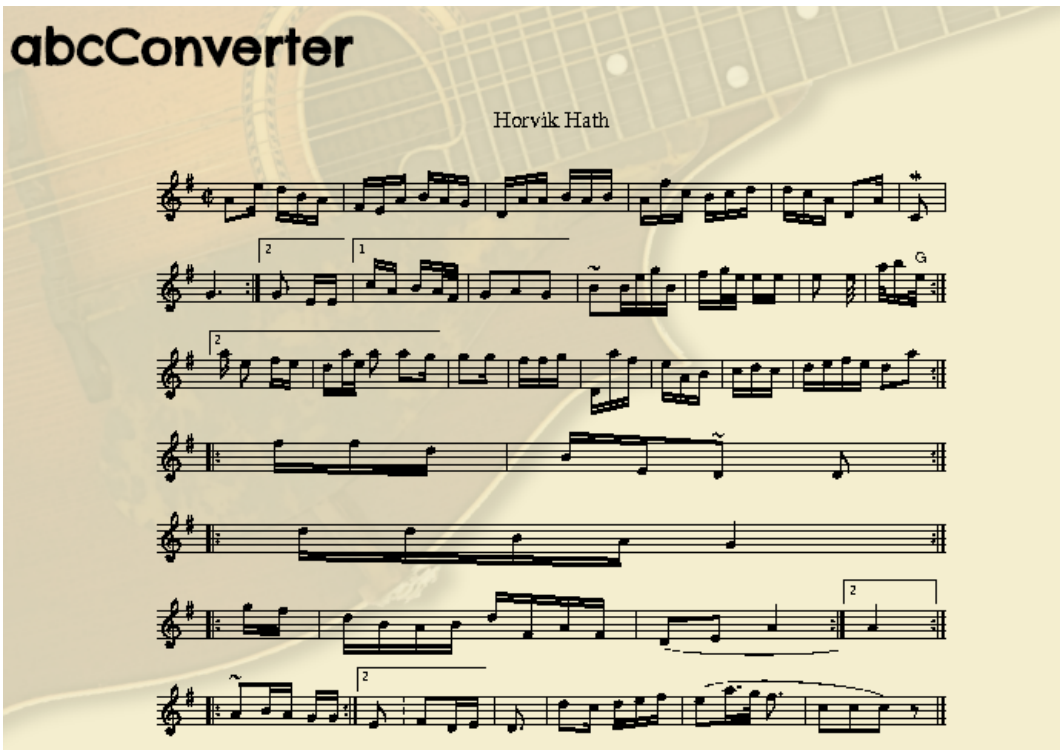

Figure 7: Temperature = 2: Music representation for Sample 1

- Secondly, we can also observe that the decrease in training loss increases as we increase the number of hidden units. In other words, training loss of 150 is better than 100 than that of 75 and so on.
- We observe that the validation loss shows a similar trend as we observed above for every choice of number of hidden units ie., the loss goes down with number of epochs (Except for the case when number of neurons is 150, we see a slight increase after 50 epochs which can be explained as a case of over fitting)
- A more interesting observation is comparing Validation loss across different number of neurons. We expected that validation loss to follow the same trend as train loss but our results show otherwise.
- Firstly, we see that validation loss is less in the case of 75 neurons and increases for 100 neurons, which explains that with 100 neurons we are likely to over fit the data than when we have 75 neurons. The graph shows that the sweet choice of number of neurons is 75
- Secondly, we also see that in the case of 150 neurons even though it has the minimum validation loss initially, it quickly increased after 50 epochs and started to perform worse than 75 hidden units. The reason again, can be attributed to over fitting the overwhelmed network with lots of parameters.

```
X:1
T:Tu dhi Ae the mang
b:01 fndedes=rn,>, B,,0,G
Z:Trad.
R:dolka
Z:id:hn-majis c/2ner
C:--6
Z:Pour toute observation mailto:galouvielle@free.fr
M:2/2
L:1/8
K:G
B2 d2 dA | cdce | dc^c cA | E2GE2Bde | d2dB BBGA | BAGG FFGA|cEF DGA|AFDE F2D|FAF Gcd:|
|:~a3/g/|
g2 dB/d/|!aine! ||
P:variat ouviellend"
IIadA Rranx.ur tou bour-17
M:C|
K:D
GF|AB GA|A>A B>B|AB/G/ GF|ED ED|EFA c2A|1 A2G G2A:|
```

Figure 8: Temperature = 2: ABC notation for Sample 2

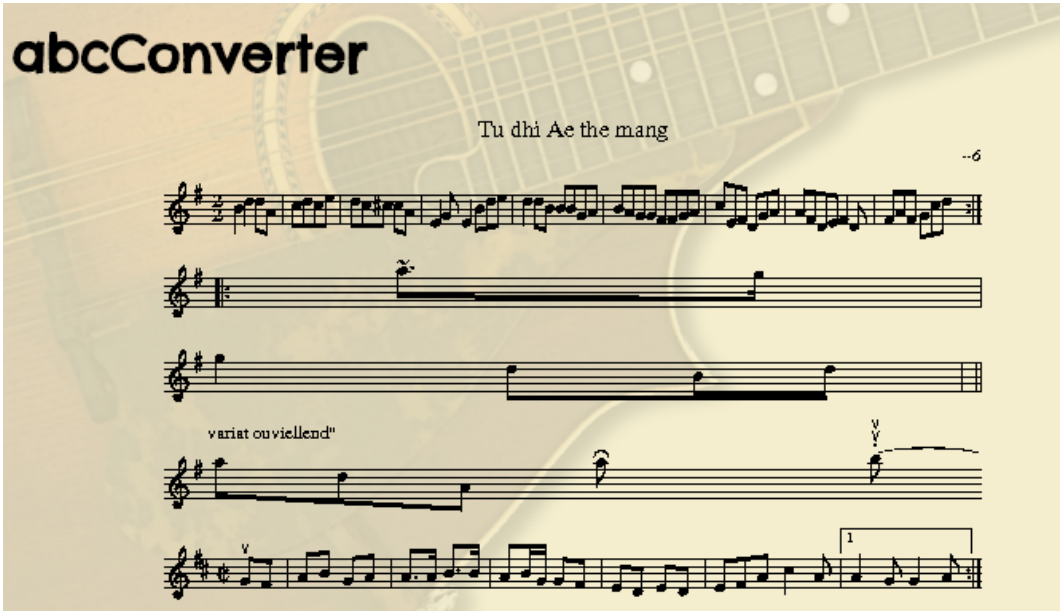

Figure 9: Temperature = 2: Music representation for Sample 2

d. Dropout

The dropout technique is a data-driven regularization method for neural networks. It consists in randomly setting some activations from a given hidden layer to zero during training. Repeating the procedure for each training example, it is equivalent to sample a network from an exponential number of architectures that share weights. **Figures: 17,18,19,20**

- We added a drop out layer after the Simple RNN layer in our model. We used the same model for calculating the training and validation accuracy.
- We observed that both the training18 and validation20 accuracy decreased as the drop out was increased.
- We also observed that the training17 and validation20 loss increased with the increase in the drop out rate.

```
X:31
T:Lither Saul? ba-de vies Fwastlaz.
Z:id:hn-aitambe- t/1/e/B/d/ c2 | d2 f2 :|
K:G
F2|A2 (3 BG |
B2- B2 : \
T:Fastien-ti-ne du rat Mazurk, bs's enstrim's Enassous"ompurre
O:France
A:Arleed:|2 Bd F2 DED:|2 B2A2 ABAG|E2AG G2:|
|:gabc a2gfec|BAAG AG (3Bcd | ~G3F ABcB|1 2GGB d2ef|
g2ec c2GA | ~A3 cde | ffa g2ga bag, | e,2E2 D,,4- ||
L:2/4
L:1/8
K:G
FG Bd|ce ef|d2cB cd|c2 d ec | A2B2 | B2A2 | B>BA>A | G2 | B2d2 | B3 :| "G"AB/2c/2|"F"Ad ed|"Bde G>d|c2G FGA ||
```

Figure 10: Temperature = 0.5: ABC notation for Sample 1

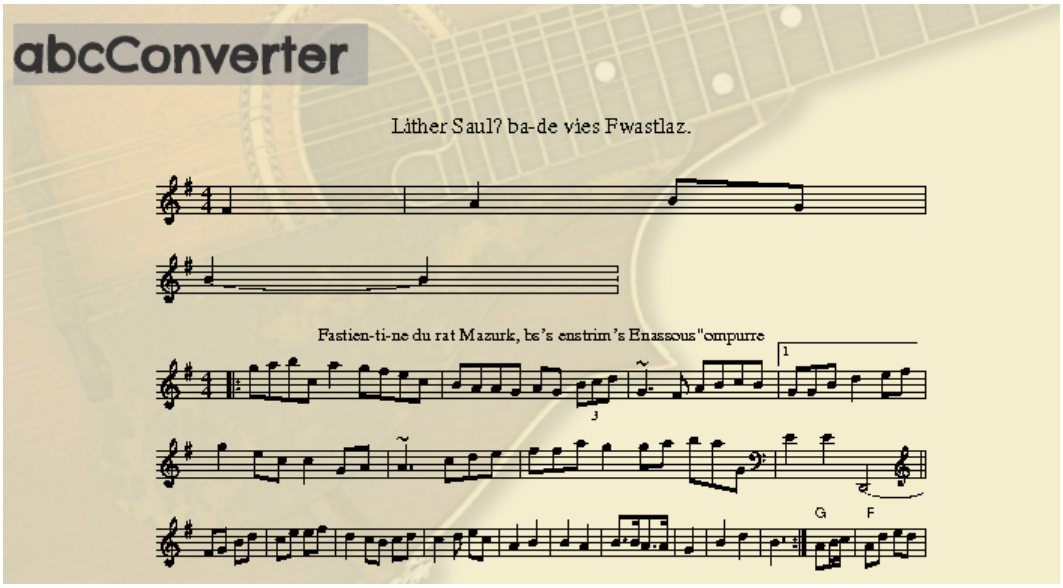

Figure 11: Temperature = 0.5: Music representation for Sample 1

- We observed that the time to run an epoch started varying with the epochs. The epoch were run in 16-19s while with out dropout, the time to run an epoch was mostly consistent. Though, we found that the dropout did not significantly affect the training speed. It still took us approximately the same time ($\sim$17-18s) to train our model for an epoch.
- The observed results were quite close to the results that we got without dropout.

e. Optimization techniques

Figures: 21,22

- We have plotted training 21 and validation loss 22 for the following optimization techniques: Adagrad, Adam, RMSprop
- We see that Adam and RMSProp have comparable performances for our application than Adagrad. We can see the similar trend for both training loss and validation loss as observed in the previous part. Adagrad performs poor convergence due o slow learning rate. Adam seeks to reduce its aggressive, monotonically decreasing learning rate. Instead of accumulating all past squared gradients

```
X:67
T:Laich ED D Sanemmen Bonsuatire carlorvation mailto:galouvie tu la Borritt tu tammob turrier parth
H:Seed fndr Afrait Gz Stotse du tance l'ane G. Chasan-__i Notton Grtsite
O:Frit eecorrig? toumoutiond]A2 gdB^c|dffg a2cB|d2c2 c2B2|c2A2 F2|GGFE cBdG|
A4BG d2:|
w:Burkos th cinas
R:polka
R:polka
Z:id:hn-polka-99
M:C|
K:Aminst
D:BA Brad(cdd) (3bafa f2gf|gefe g2 gdBB | FABB G3 | d2dedc | cfc | B3A | F6 | F2CF | c2e2 |
{a}x16 | cB=B2AG | B2 A2FF | B6- | F2B2B2 ! !brealatponk og Corste #27, dommonn:F
Z:Transcrit et/ou an pos pleend's cale fringliny
O:Provence
Z:Transcrit et/ou corrig? par Michel,
Z:Transcrit et/ou corrig? par Micorn-ti-- tal?o] \
d/d/f/d/|g2 fg/B/|B/cB/d/ cc|d/A/c/d/ Bd:|
|:FA AAd|c2c eec|dce c2B|cBd e2d|ef e/g/ e2 | c2 cz | f>f ee fd cd | d2 c !+!BF)d| ded g2g | f2f e2d | G2c B2 :|
|: fde fd | B3-B3 |  z2B>B | F2AG | F2G BGA | g3 bge | d2B GGA | cBc dcA | "D2D>E G/d/B/d/ | cB FF | G6 AF|FA A2|: [B2c] "C"d4 | e6=B, E3 :: B2B3B | G3 E2ED |
cBdc FGAG | ~E3 DDE | G3G | A2G2 | A2cd e2cd | G6 d2A, | G,,EG, ||
T:BGid Dasthen-"Mofncoim
T:Lastion-hornpipe
C:Br
```

Figure 12: Temperature = 0.5: ABC notation for Sample 2

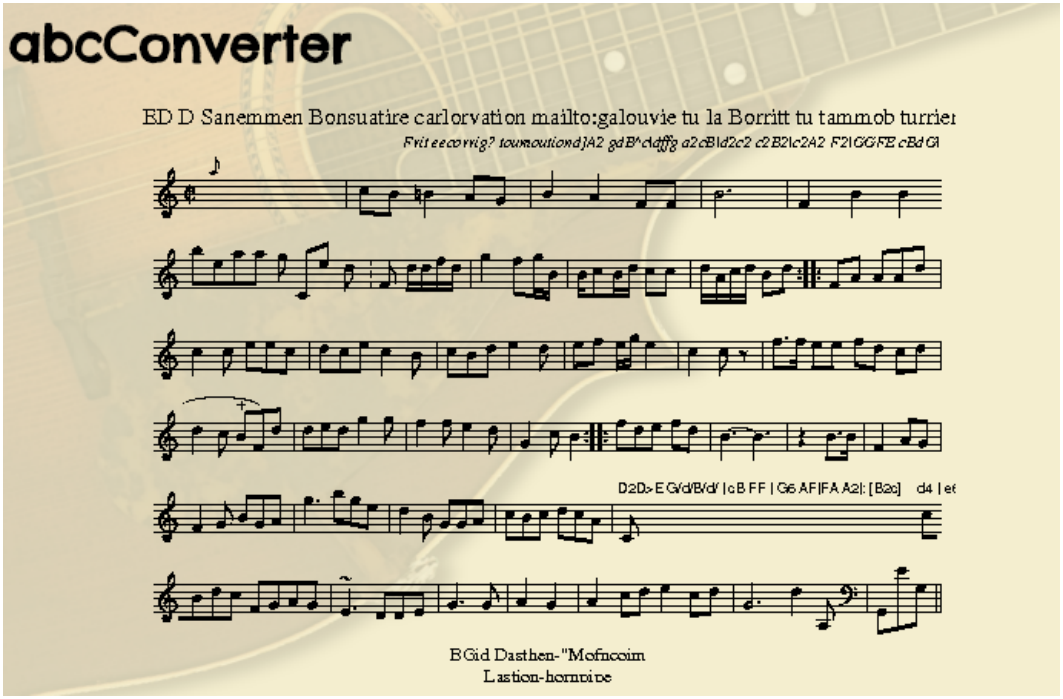

Figure 13: Temperature = 0.5: Music representation for Sample 2

f. Feature Evaluation

We observed the activations of a neuron for a single sequence and observed the heatmaps as can be seen in figure 23 . The heatmaps show us a pattern in which the music of a song show similar variation in the heat. That is, these delimit some notion of music that we as music novice find difficult to interpret. The heatmaps provide us a medium to get better insight into the functioning of the RNN model .

1 Learning and Results

We learned that RNN possesses great power to not only predict next character in the sequence but can also generate music at its own behest without much triggering. This shows us the enormous capacity of RNNs to do feature engineering across time. The natural algorithms like BPTT take care of the shared weights learned across time. And hence provide us a learned model.

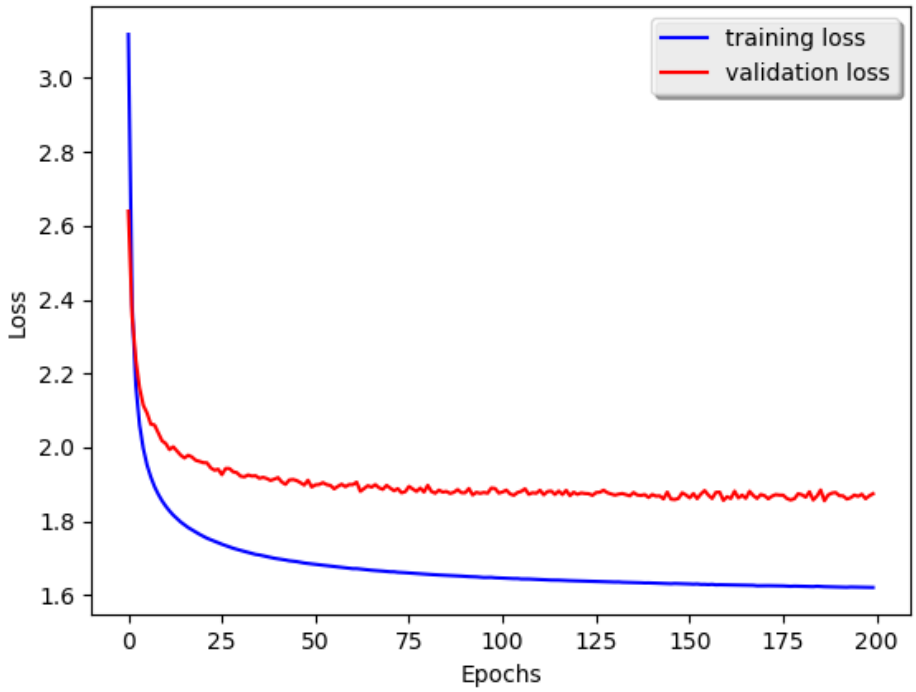

Figure 14: Train and Validation loss variation with epochs

Table 1: Contribution by each team member

S.no.	Name	Contribution
1	Swapnil Taneja	Code implementation and experimented with feature evaluation
2	Saksham Sharma	Code implementation and experimented with temperature parameter
3	Kriti Aggarwal	Code implementation and experimented with dropout parameter
4	Prahal Arora	Code implementation and experimented with the number of neurons
5	Saicharan Duppati	Code implementation and experimented with different optimization techniques

2 Contributions

We made sure that most of the work is equally distributed and shared our individual contributions for the team assignment, so that each of us understood the latters work. Following are roughly the contributions made (see Table 1.):

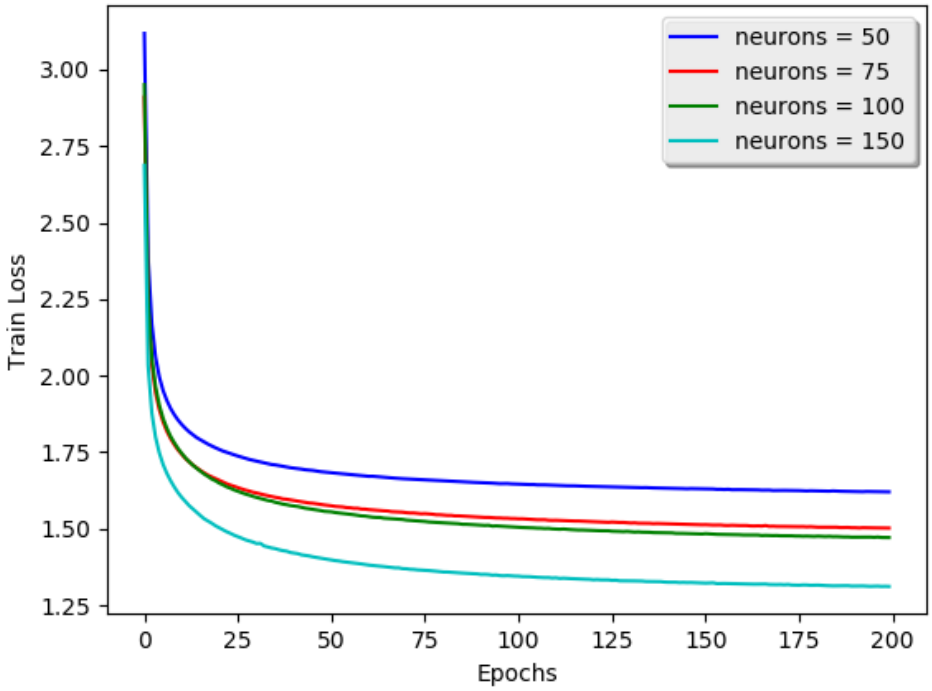

Figure 15: Train Loss for various number of neurons variation with epochs

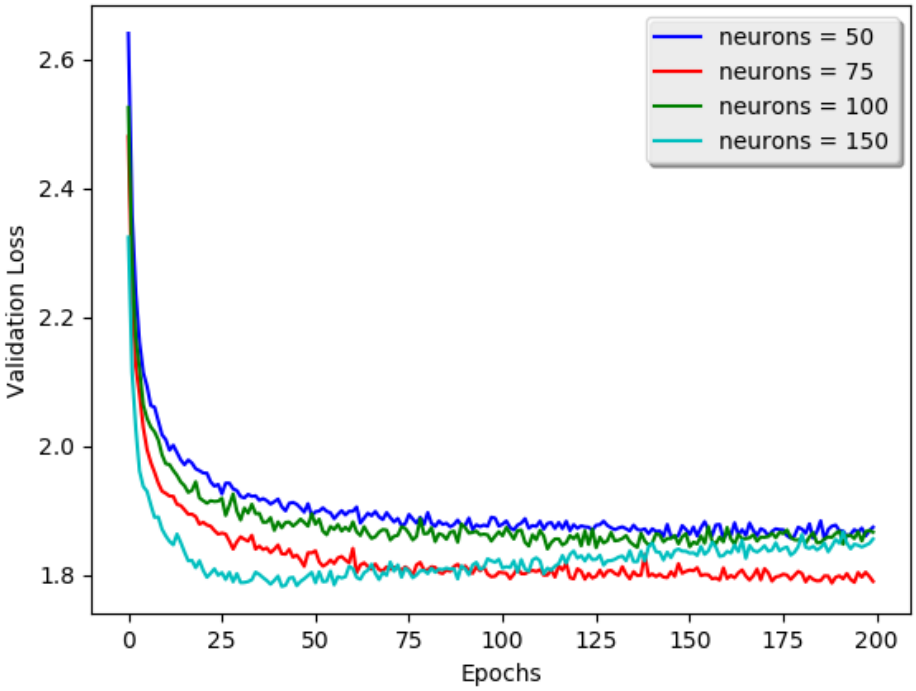

Figure 16: Validation Loss for various number of neurons variation with epochs

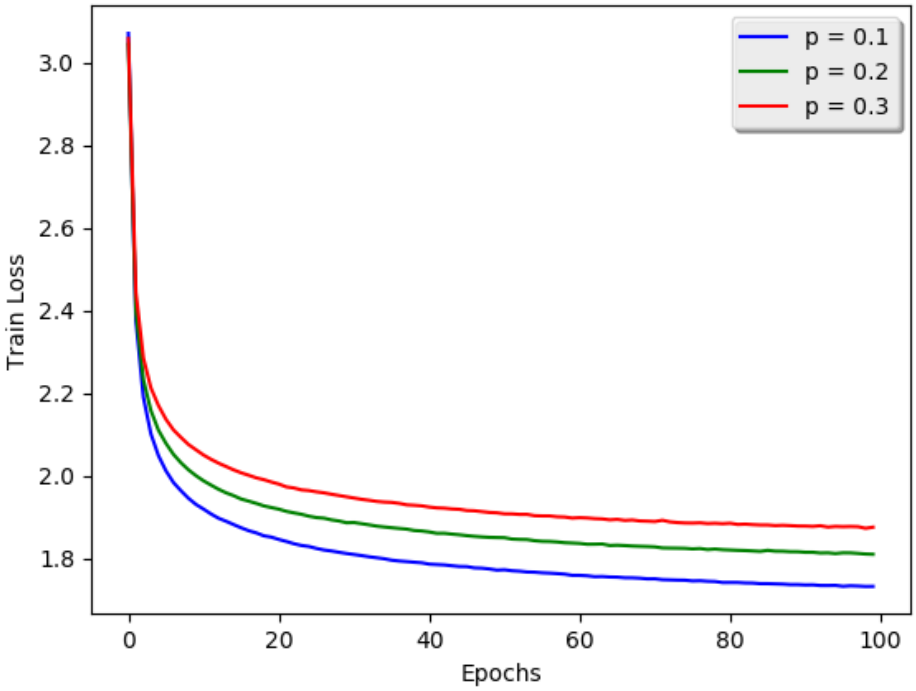

Figure 17: Dropout: Training loss vs number of epochs

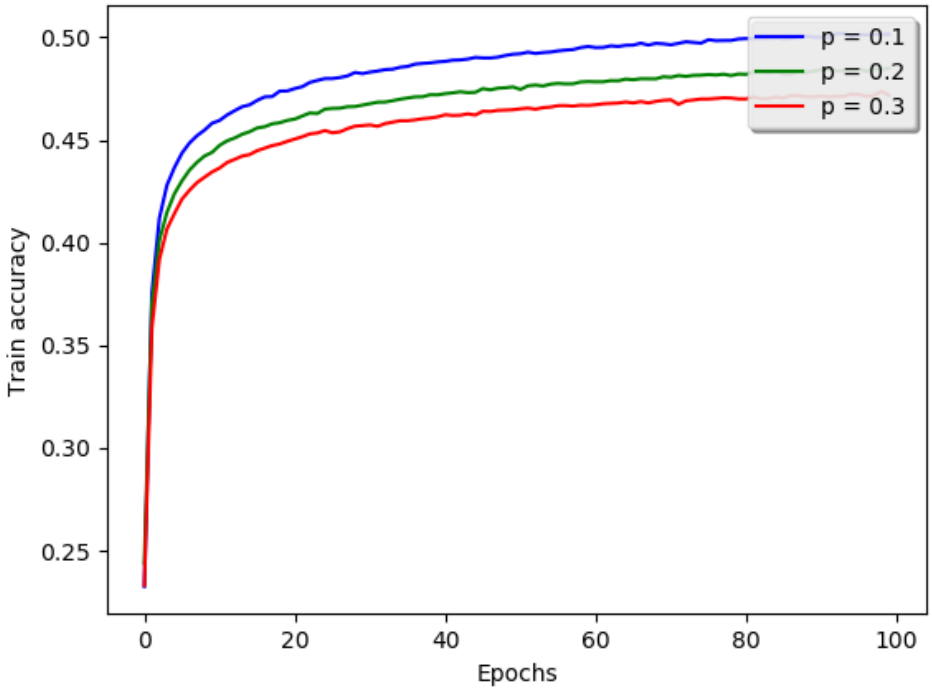

Figure 18: Dropout: Training accuracy vs number of epochs

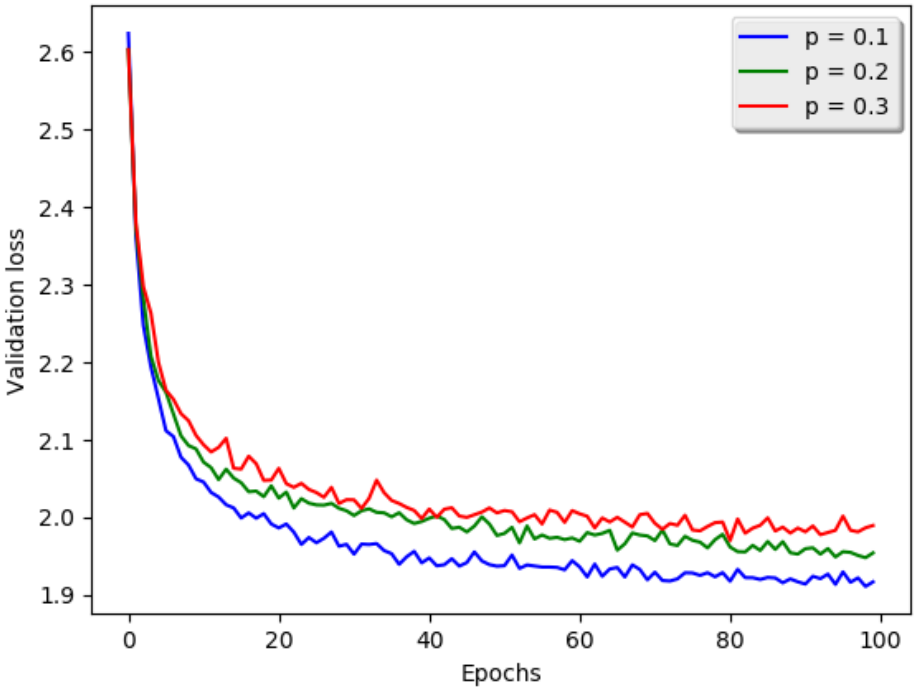

Figure 19: Dropout: Validation loss vs number of epochs

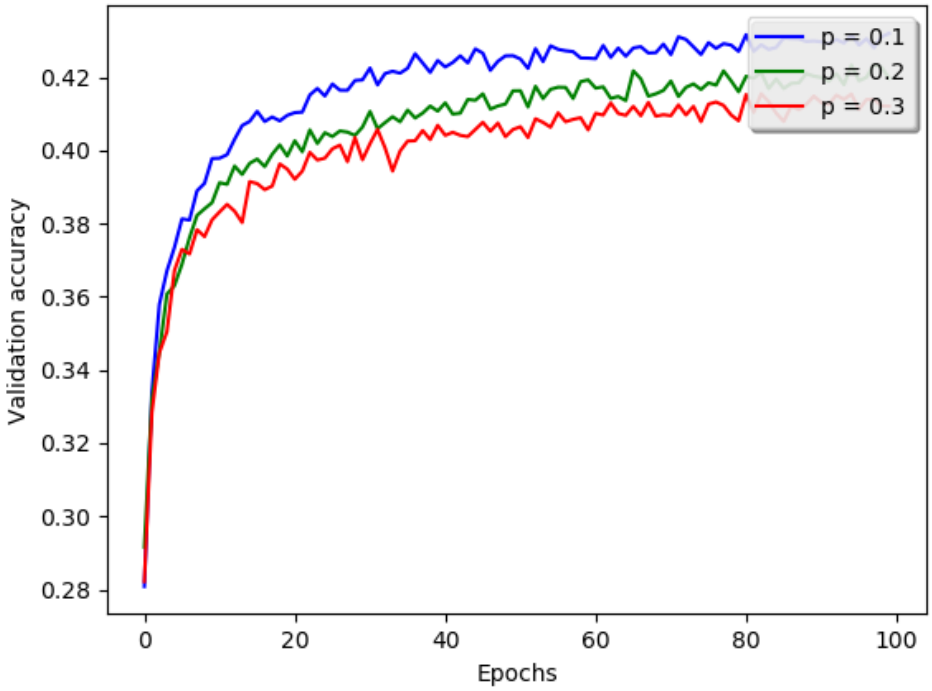

Figure 20: Dropout: Validation accuracy vs number of epochs

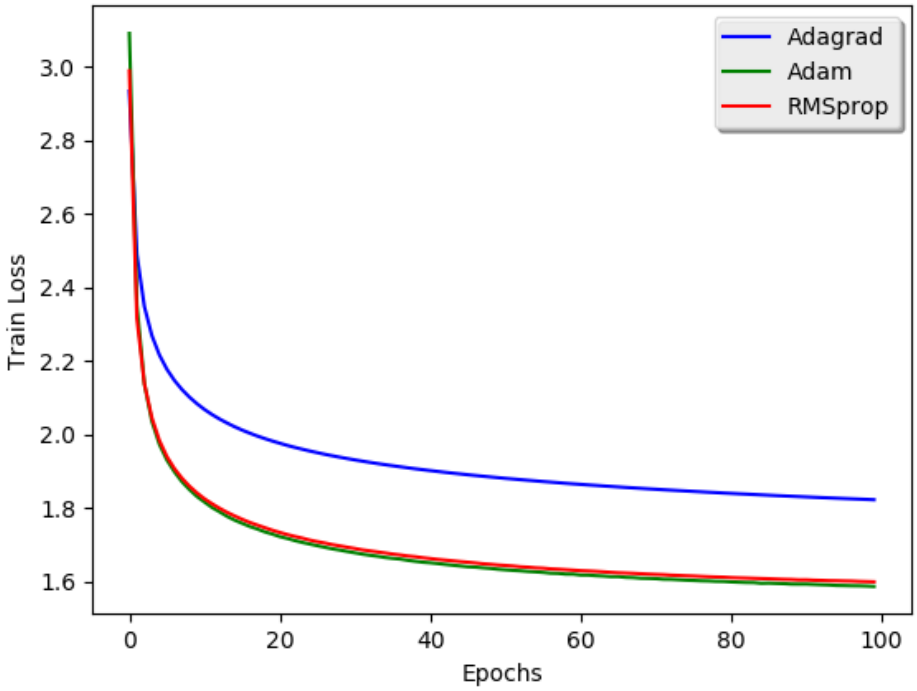

Figure 21: Train Loss with epochs for various number of neurons variation with epochs

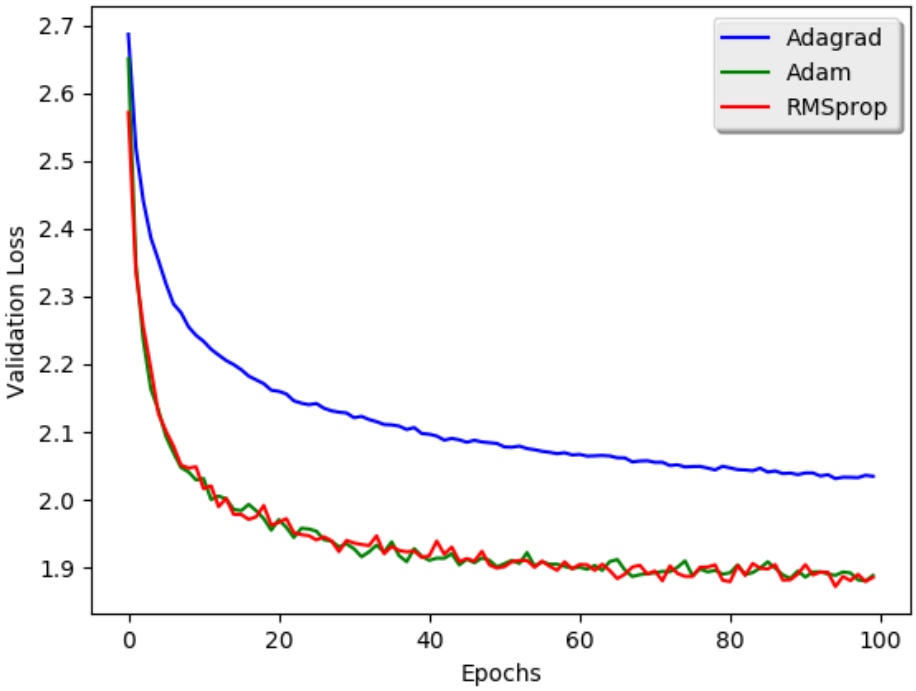

Figure 22: validation Loss with epochs for various optimization techniques

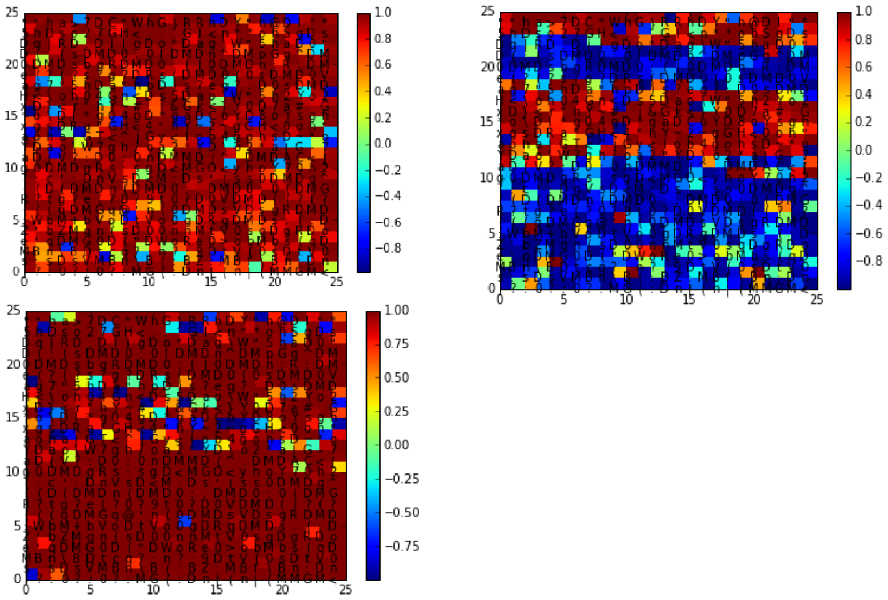

Figure 23: Heat maps for a various hidden neurons