Generating Music with Recurrent Networks

Yunsheng Li
A53086891
yul554@ucsd.edu

Yi Luo
A53091187
yil485@ucsd.edu

Xiyun Liu
A53099348
xil429@ucsd.edu

Siwen Yan
A53093655
siy043@ucsd.edu

Abstract

Convolutional Neural Networks assumes that the data points were IID and temporally independent. However, CNNs are unable to model some tasks where the data have temporal dependencies. For these sequential relationships Recurrent Neural Networks (RNNs) are used. In this assignment, the power of Recurrent Neural Networks is explored by using a basic RNN model to compose music.

1 Introduction

In the previous assignment, Convolutional Neural Networks are used to do several tasks. However, for all those tasks, we assumed that the data points were IID and temporally independent. CNNs are unable to model some tasks where the data have temporal dependencies.

For the problems which have time dependence, sequential relationships Recurrent Neural Networks (RNNs) are used. In this assignment, the power of Recurrent Neural Networks is explored by using a basic RNN model to compose music.

2 Method

To explore the power of Recurrent Neural Networks, in this assignment, sequential relationships Recurrent Neural Networks (RNNs) is used to generate music in abc format. We first train a basic RNN model using characters extracted from a music dataset and then un the network in generative mode to "compose" music.

Then we further investigate the parameters of this model. First, using different temperature in the softmax output layer and generate music pieces, we could get some insight about the relationship between temperature T and the performance of the result. Second, try changing the number of neurons in the hidden layer. Third, use dropout with p=0.1,0.2 and 0.3 to investigate how it influences the loss of training and validation. Next, we look into the optimizer and compare the performance between using RMSprop and Adagrad. Finally, by creating heatmap for a particular hidden neuron, we could get some kind of insight about how this particular neuron fires.

3 Results

3.1 Generate 6 sample music pieces

After experiment with T, the best music piece we generated is the first one using T = 0.5, 100 hidden neurons, RMSprop optimization, no dropout., which is shown in figure 1 and 2.

Here we shown all six music pieces below.

29th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

The two music pieces generated with T = 0.5 are shown in figure 1 to figure 4. Besides T = 0.5, we use 100 hidden neurons, RMSprop optimization, no dropout. It has obvious musical pace , sounds comfortable and a little familiar to what we have heard before.

```
<start>
X:33
T:Farcain
Z:id:hn-reel-78
M:2/4
Q:1/4=120
M:2/4
L:1/8
K:D
DFG EDC|cAGA d2ab||
fagf a2fd|~A3c d2cB|cdBA G2AB|1 ABcA GFEF|d2gf edAG|AFGB AGAB|c2 (3B^cde^fgg|abag d2
(3dc-|d2d :|
<end>
```

Figure 1: Music 1 with T = 0.5 in ABC notation

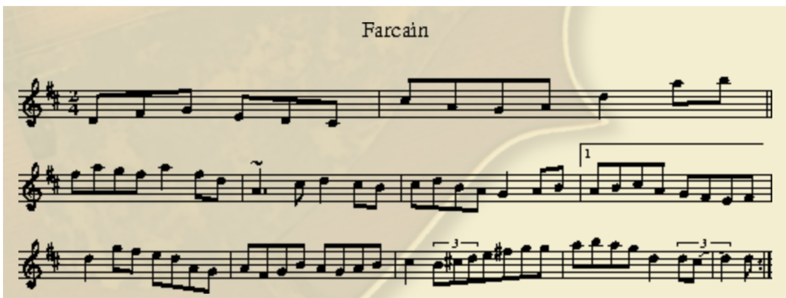


Figure 2: Music 1 with T = 0.5 in standard music notation

```
<start>
X:V3
T:Cailley als's Twon's ha di Drimn Fros is downy: The Nipele the Mithee a Bouscd jo- 1 -22
M:2/4
L:1/8
K:A
D2 (3fec d2 (3cAG E2 :|2 EGAG EGGF | E2G2 BAAF |
B,EG,2 D,ED|D2DE DE:|
|:BE/F/A/B/|(BAd) ed^c|(c>e 2feg|dBAF|d2ea f BcdB|A6 (3c2c a2bB||
<end>
```

Figure 3: Music 2 with T = 0.5 in ABC notation

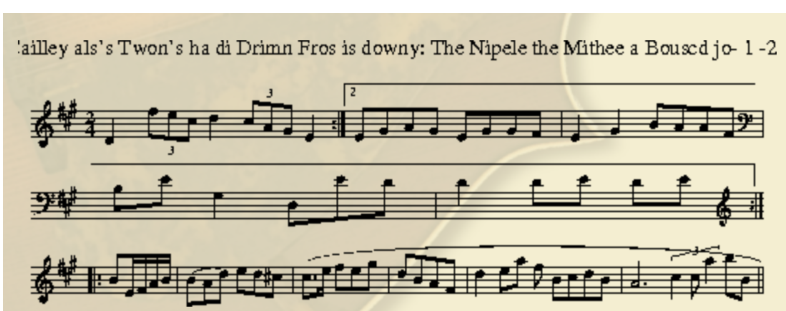


Figure 4: Music 2 with T = 0.5 in standard music notation

The two music pieces generated with T = 1 are shown in figure 5 to figure 8. Besides T = 1, we use 100 hidden neurons, RMSprop optimization, no dropout.

```
<start>
X:98
T:Ewnoge Wie: The Fablery S'oon With tonce#m7
A:Fasouis O'Keenapole
M:3/4
L:1/8
K:G
g>g|
d2 (3ABc (3cBd|edcA ADD2:|
|:(3dff fe dd|1 dAG A2D||
<end>
```

Figure 5: Music 1 with T = 1 in ABC notation

Figure 6: Music 1 with T = 1 in standard music notation

```
X:150
N:69
M:Ch
R:marnplaalld maslang.
Z:id:hornail
O:France
[M:8/4
K:D
~d3f ghze | fffe dfec|dcBA GFFA |
BGBd gfd2:|2ed c2|eeed ge^fd|.e2^c2 =Bd^cc | d4c2 | d/F/A/G/ AB (3cBA | GBcd Bggg | (3dcB
Bc af (3ega |
e3 A2A|dBd cAG|GEF BGE|FAB AFD|EFG A2d|eAA D2:|
```

Figure 7: Music 2 with T = 1 in ABC notation

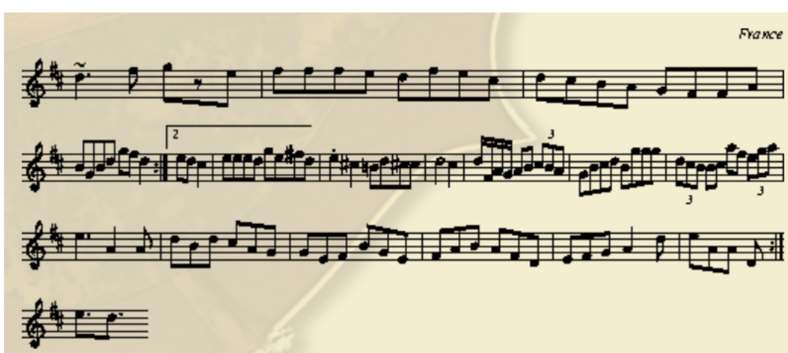


Figure 8: Music 2 with T = 1 in standard music notation

The two music pieces generated with T = 2 are shown in figure 5 to figure 8. Besides T = 2, we use 100 hidden neurons, RMSprop optimization, no dropout.

```
<start>
X:98
T:Ewnoge Wie: The Fablery S'oon With tonce#m7
A:Fasouis O'Keenapole
M:3/4
L:1/8
K:G
g>g|
d2 (3ABc (3cBd|edcA ADD2:|
|:(3dff fe dd|1 dAG A2D||
<end>
```

Figure 9: Music 1 with T = 2 in ABC notation

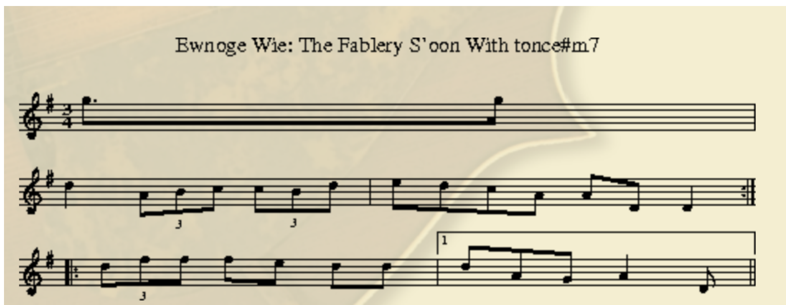


Figure 10: Music 1 with T = 2 in standard music notation

```
<start>
X:1245
T:Le wai dried S
L:1/8
K:G
d3 d2g|f2e fdf ecBc|A3A GE~D2||
:6d3b a2gf|
g2ba gbmaj
|:B/B>B/Bc | c2 dc d3 | c2z F2A | B2_"G"d3B2A | G4 G2G||
<end>
```

Figure 11: Music 2 with T = 2 in ABC notation

Figure 12: Music 2 with T = 2 in standard music notation

3.2 Training & validation loss vs. Number of epochs

For T = 1, 100 hidden neurons, no dropout and using Rmsprop, the training loss and validation loss vs number of epochs on data is shown in figure 13.

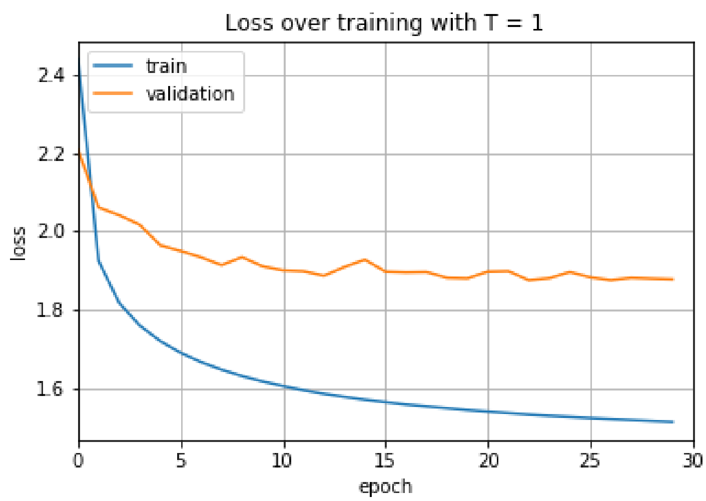


Figure 13: Loss over training with T = 1

For T = 2, 100 hidden neurons, no dropout and using Rmsprop, the training loss and validation loss vs number of epochs on data is shown in figure 14.

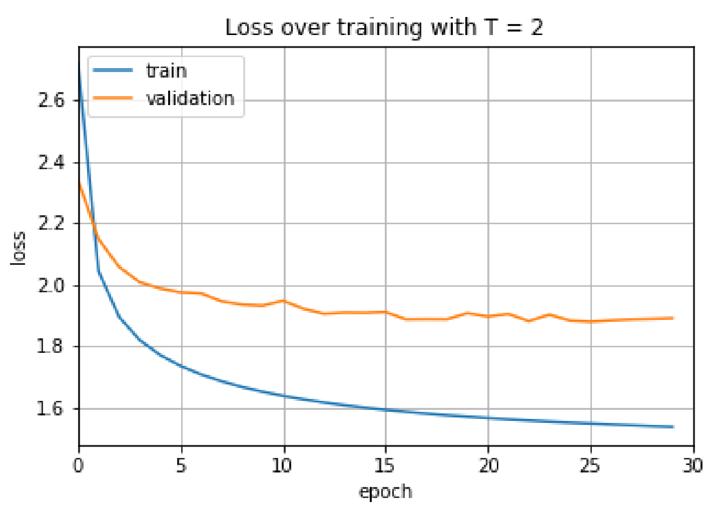


Figure 14: Loss over training with T = 2

For T = 0.5, 100 hidden neurons, no dropout and using Rmsprop, the training loss and validation loss vs number of epochs on data is shown in figure 15.

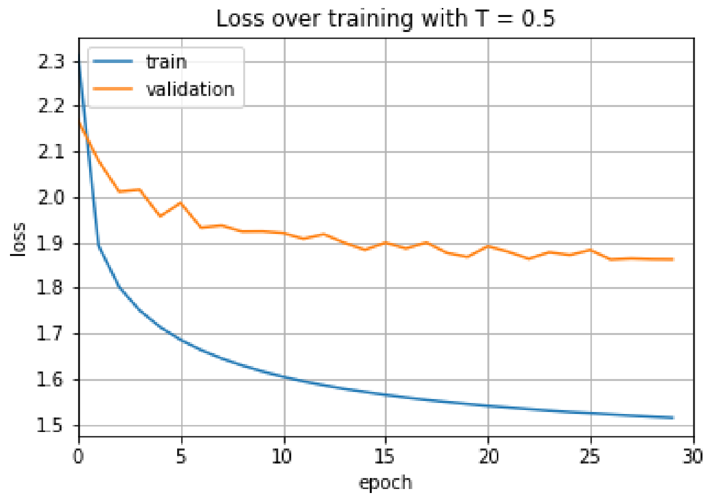


Figure 15: Loss over training with T = 0.5

3.3 Changing the number of neurons

For 50 hidden neurons, T = 1, no dropout and using Rmsprop, the training loss and validation loss vs number of epochs on data is shown in figure 16.

Figure 16: Loss over training with 50 hidden neurons

For 75 hidden neurons, T = 1, no dropout and using Rmsprop, the training loss and validation loss vs number of epochs on data is shown in figure 17.

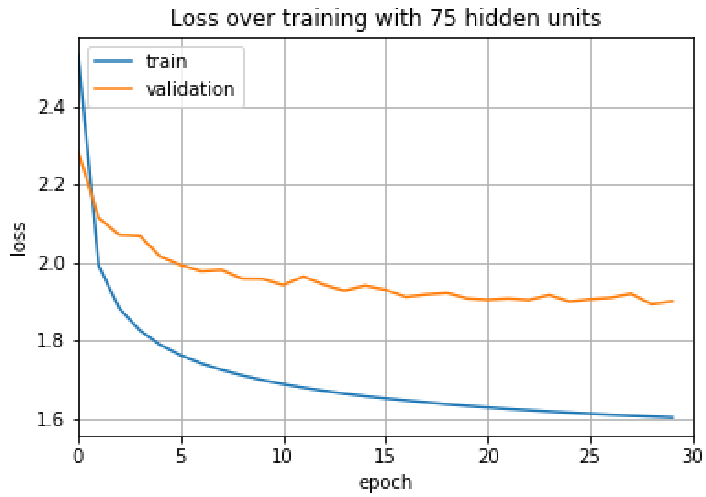


Figure 17: Loss over training with 75 hidden neurons

For 150 hidden neurons, T = 1, no dropout and using Rmsprop, the training loss and validation loss vs number of epochs on data is shown in figure 18.

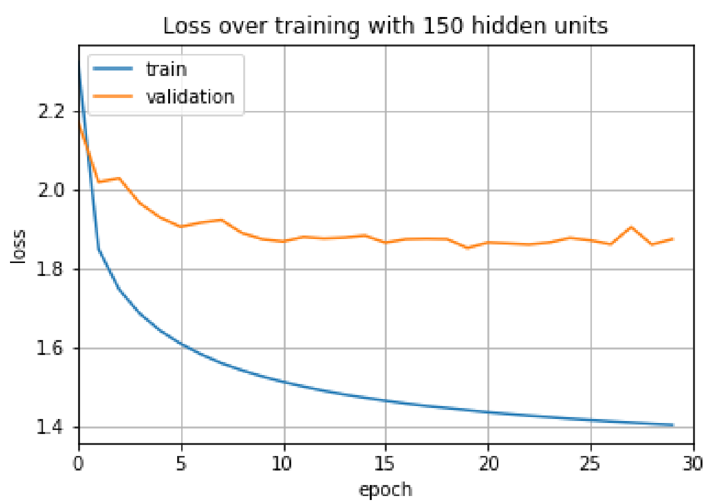


Figure 18: Loss over training with 150 hidden neurons

3.4 Using dropout

For p = 0.1, T = 1, 100 hidden neurons and using Rmsprop, the training loss and validation loss vs number of epochs on data is shown in figure 19. The generated music is shown in figure 20 and 21.

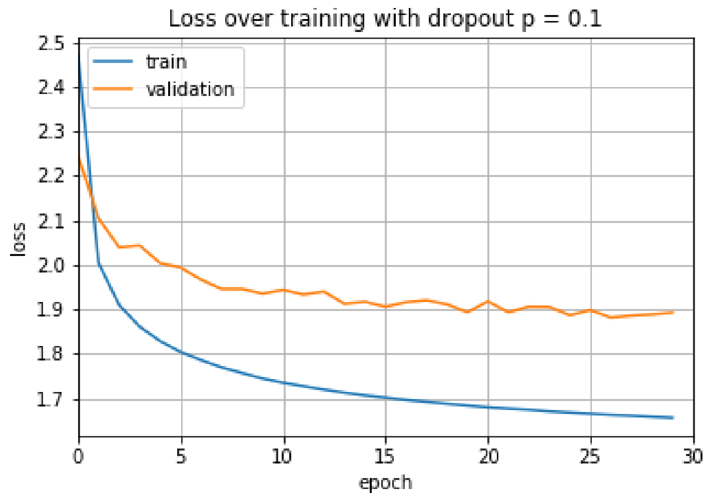


Figure 19: Loss over training with p = 0.1

```
<start>
X:3
T:O'Sc Maranc Ri-fer pertoh rneinel Benas-se
Z:id:hn-jig-31
M:C|
K:C
V:2
Ae-f2 f2d2 | d2dd) | (f4) | f4f | B>B | d3) ||
effge|edf e2e:|
|:fBAc d>ed | fef f2B | F2F G3d | e2d eag | fed ABd | cAG A B GF | d3 G2 | D2EF D2E2 |
A4-2 ec dc d2 | AG fe cB | dB cd|ce d>f | Gi Cf par'N?c ~f3ded|FA~B2 AGEE|dd:F2~A2DeedcB|
G2G2 G2B2|FABc AGBA|c2cB A2G2|B2B2|B>fd>B|cdce ffef|fecB decB|| (3cBc (3Bcd (3FAA|b
G2 D2|A4:|
<end>
```

Figure 20: Music generated with p = 0.1 in ABC format

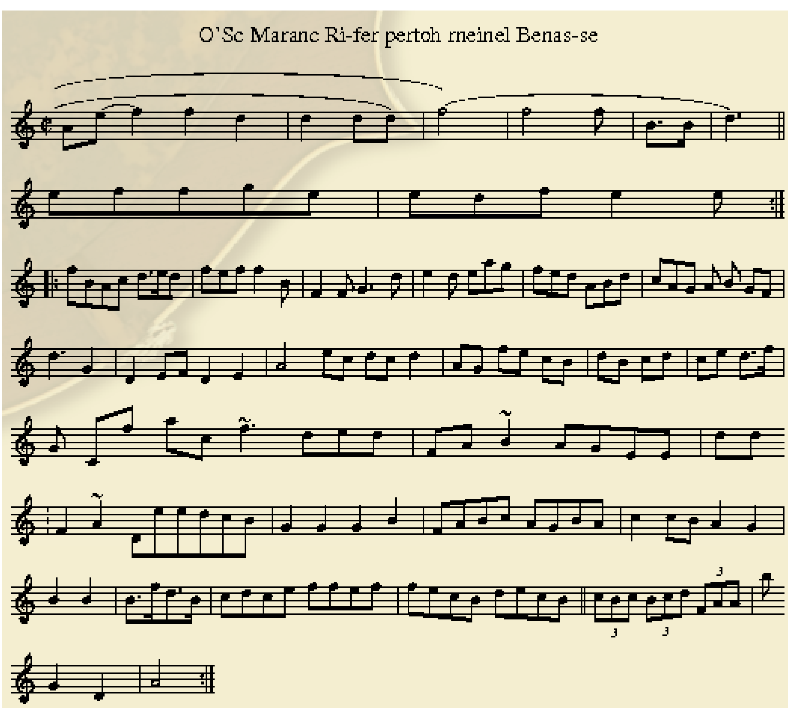


Figure 21: Music generated with p = 0.1 in standard format

For p = 0.2, T = 1, 100 hidden neurons and using Rmsprop, the training loss and validation loss vs number of epochs on data is shown in figure 22. The generated music piece is shown in figure 23 and 24.

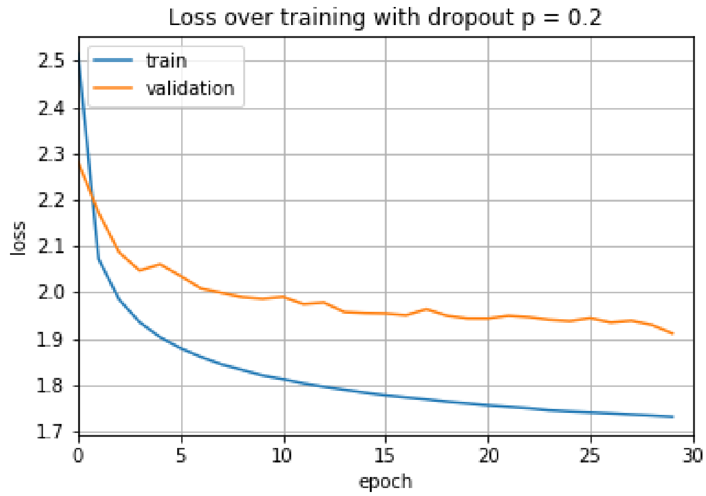


Figure 22: Loss over training with p = 0.2

```
X:69
T:Miche minh S|e2" Lench don mailit:
M:2/4
L:1/8
K:Gdar
~arsarast
R:Nor-jid H:.
Q:0/4e/2f/23/2e/2d/2e/2 Pc3z | BBBB | B4B2 | Dcdg fred2 f2d |
F2- A BAB |
c2B BAe | deB cBA ||
|: ~E3A BABc dBBG | DB2G2G ||
```

Figure 23: Music generated with p = 0.2 in ABC format

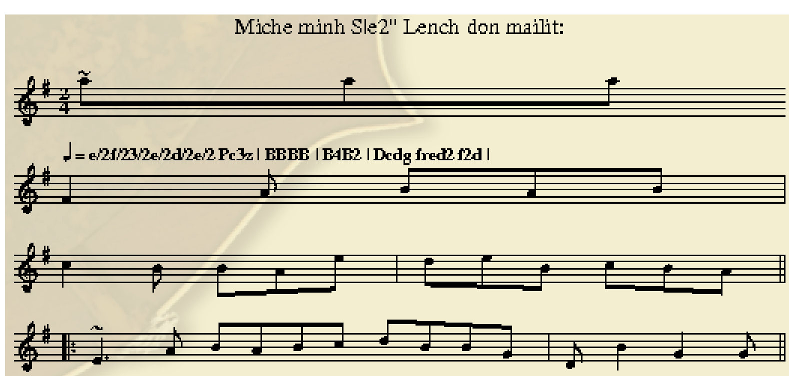


Figure 24: Music generated with p = 0.2 in standard format

For p = 0.3, T = 1, 100 hidden neurons and using Rmsprop, the training loss and validation loss vs number of epochs on data is shown in figure 25. The generated music piece is shwon in figure 26 and 27.

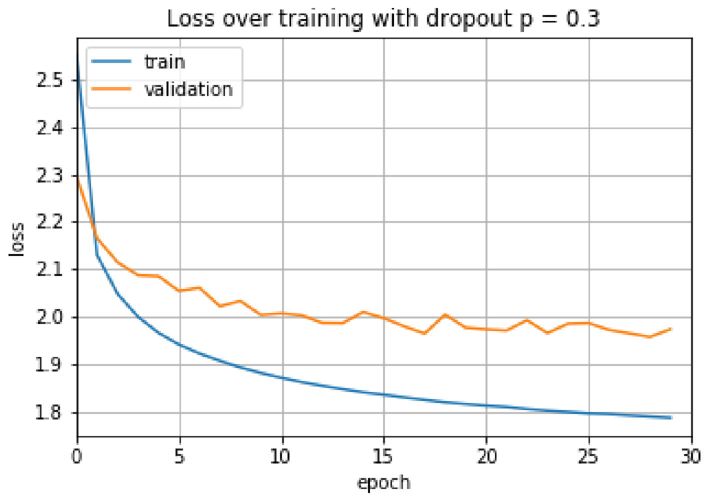


Figure 25: Loss over training with p = 0.3

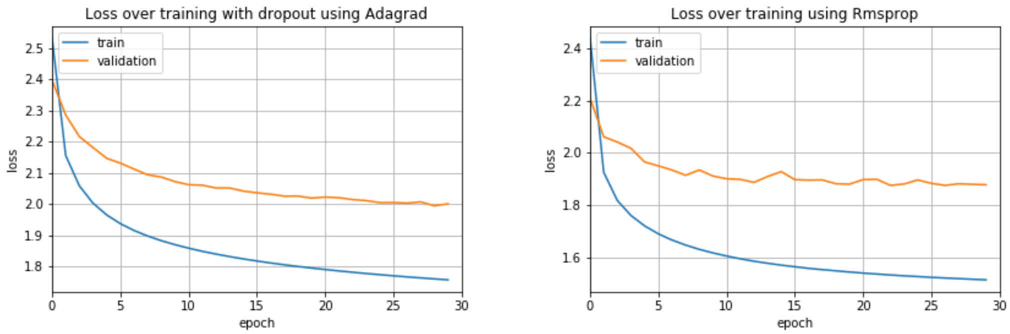
```
<start>
X:4
T:Trag <onwi
|:eA FFG/B/|Bd fe|ac/c/ BG|e2 d2 e>B ||
K:Bb
V:Franom G, La (3baral BThe Ried li ballon'strid Bd Bc GG BG | AG FA |
D2G2 | A4| A,GE | G2GE DFB2 |
(ec d2d| c e>d =e2 | F8 | (3FDF ~G3G||
|:ce F2GEF|d4c2 c|06-G2164
g:
3:|
<end>
```

Figure 26: Music generated with p = 0.3 in ABC format

Figure 27: Music generated with p = 0.3 in standard format

3.5 Using different optimization

The plot of loss over training using different optimization is shown in figure 28. Besides optimization, they both use T = 1, 100 hidden neurons, no dropout.



(a) Loss using Adagrad

(b) Loss using RMSprop

Figure 28: Loss over training using Adagrad and RMSprop

3.6 Feature Evaluation

For the second music piece generated using T=1, 100 hidden neurons, RMSprop and no dropout, we do forward propogation through the model's network and generate the heapmap for each particular neuron. The plot of one particular neuron (#12)is shown in figure 29.

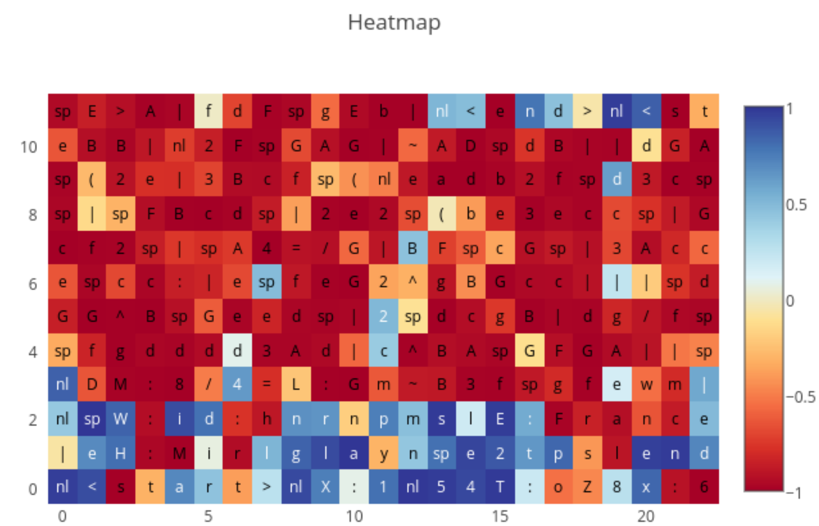


Figure 29: Heapmap for a particular neuron

4 Discussion

4.1 Generate 6 sample music pieces

The six sample music pieces are shown in figure 1 to 12. The hyperparameters are reported in the last section. From the experiment, we found that the music sounds more comfortable and has more musical pace when we use smaller T. However, we also found that actually when T = 2, we generated more reasonable music pieces than when T = 0.5 and 1. In the future, we could investigate why this happens by running more models with different T.

4.2 Training & validation loss vs. Number of epochs

The comparison between the final loss of training and validation is shown in table 1. Comparing figure 13, 14 and 15, we could see that the final training loss and validation loss are smaller if we use a smaller T. During generating sample music pieces, we found that smaller T could give us a smoother and comfortable melodies.

T	final training loss	final validation loss
0.5	1.5150973852157592	1.8628008056640626
1	1.5168706178665161	1.8788561856842041
2	1.5383447570800781	1.8907986346435546

Table 1: Final loss comparison between using different T

4.3 Changing the number of neurons in hidden layer

Comparing figure 16, 17 and 18, we could see that the final training loss and validation loss are smaller if we use more neurons. This is reasonable, since more neurons could capture more features from the input. The final loss comparison is shown in table 2.

Number of neurons	final training loss	final validation loss
50	1.7148263666915893	1.9460150500869751
75	1.6019527910995484	1.9000797046661377
100	1.5168706178665161	1.8788561856842041
150	1.4039436611175538	1.8749046572494508

Table 2: Final loss comparison between using different number of hidden neurons

4.4 Using dropout

Dropout is to prevent overfitting during training. However, our model actually is very simple and does not have many features to make it overfitting. Therefore it is not suitable to use dropout technique here.

Comparing figure 19 to 27, we could see that dropout increase the time to converge to a lower loss, decrease the training speed and makes the result worse in this assignment. With larger p, the final loss of training set and validation set both increase, the music quality decreases and it is harder to find good music pieces.

p	final training loss	final validation loss
0.1	1.656528591003418	1.8923998642730713
0.2	1.7309635577392579	1.9111485869598388
0.3	1.786046654663086	1.9730128372955322

Table 3: Final loss comparison between different dropout

From the figure we found that at the first epoch, the validation loss is lower than the training loss. This is because that dropout is used during training and the training loss is calculated based upon the specific randomly selected dropped-out network, which is much less likely to be well tuned than the validation loss which is calculated based upon the full network(with adjusted weights) on the validation data.

4.5 Use different optimization

Comparing figure 28(a) and (b), we could see that RMSprop gives us better result than Adagrad.

optimizer	final training loss	final validation loss
Adagrad	1.756101514892578	2.0004622744750975
RMSprop	1.5168706178665161	1.8788561856842041

Table 4: Final loss comparison using different optimizer

4.6 Feature Evaluation

From the figure 29, the heatmap could give us some kind of insight. We could see that this particular neuron fires for the header, which means that it is able to recognize header of music in ABC format.

5 Conclusion

In this assignment, we trained a basic RNN and generate music in abc format. We also investigated some hyperparameters and optimizor of this model. Several music pieces are generated.

6 Contribution of Each Member

We contribute equally to this assignment, we program and discuss together.