CSE 253: Neural Networks
Winter 2018
Homework Assignment 4:
Generating Music with Recurrent Networks

Jiawei Li(A53226117) jil206@ucsd.edu
Zhiling Liu(A53226055) zhl385@ucsd.edu
Deling Li(A53108804) del078@ucsd.edu
Xuefeng Shen(A53070578) xus009@ucsd.edu

Abstract

To generate a good piece of music, we need model to understand what is good inside a music. To learn such sequence data, RNN is the most powerful architecture. With the help of Pytorch RNN module, we implemented RNN models with different sizes, types, and dropout values, as well as other factors like batch-size, optimizers, etc.. And finally, after training by Adagrad optimizer with batch-size being 50, a RNN containing one LSTM layer with 150 units, gave us the best validation loss, 1.617 and many beautiful musics.

1 Generate Music With Different Temperatures

In order to get the best music according to our model, our team decide to do this part in the end, so we can get the best value for dropout, hidden unit and optimizer, we are going to discuss how we get these values in the rest part. The result we get is dropout=0.1, hidden units=150, numbers of layer=1, type of layer=LSTM, learning rate=0.001, batch-size=100,optimizer is Adagrad.

To prevent the case the our model may stuck in a loop or the music it generate is too long, we set a limitation that when the character of the music reach 1000, it will stop. Therefore there may be some music that doesn't contain ¡end¿, but the rest 6 music are guarantee to be playable.

The six music result based on T=2, T=1, T=0.5, is shown below.

```
<start>
X:1
T:Brigans in A, #34)
T:Marche
R:hornpipe
Z:id:hn-reel-75
M:C|
K:D
A2 AG|G2EF D2A | BGBGB d2d2 e2 | g2 e2ga gef|gfed cAFA|B2B B2A G2B:|2 A2 G2|d2A2 d2B | dcde | (3edcB AGE | G2GA BcdB | d3ed e2d
2 | e2c2 edc | dedB AGA | B2Bc dcBAG | A2Bc B2B2 | B2B2B2 | B2B2B2 B2B2 | BBBBBBBBBBBB | d2B2 | B4B2 | c2c2B2 | ABcG Bcde|~g3a
afe|d2B2 BAF | G2G2 G2 | G4 ||
<end>
```

Figure 1: The ABC format(1) for T=0.5

```
<start>
X:1
T:Peny's That Radhaire Ginas (1924)
O:France
A:Provence
C:Trad.
Z:id:hn-hornpipe-24
M:C|
K:D
A2 Ac BA|GA Bc|d2 (3efg agfe | d2d2 d2 | e2d c2A2 | AGF ED | G4 | F2 EF | A2 A2 | G4 | B3 | B2B2 | B2B2 B2 | B2B2 B2z2zB | B2B2
B2 | B2B2B2 | d2 d2 | c2c2 | A2G2 | d2A2 | B2B2 | B2B2 | B2B2 BB B2B | B2G2 | d2 d2 | e2f2 | (c/=B/c/d/ | f2f | e2 e2 | e2 fd |
(c/=B/A/|
BABc def|ede fed|B2z2|
B2 B2 c2 | B3 | cBc cB | B2B2B2 | G2 B2 | B2 B2 | d2d2 | c2c2 | B2B2 | B2B2 | B2B2B2 | B2B2 | B2B | d2d2 | c2A2 | B2B2 | B2B2B2
B2 | B2B2B2 | B2B2 | B2B2 | B2 d2 | egab gbb | faf g2 | edcB | d2B2 B2 | B2B2B2 | c2c d2 | BAG G2 | B2 B>c | dcde | f2d | c2 A2
| A2G2 | Gz2 | B2B2 | B2B2 | B2A2 | B2B2 | d2 d2 | c2 BA | B2B2 | B2 c2 | d2 dd | cdc A2 | AA/B/ A2 | F2 G2 B2 | A2 A2 |
(3Bcd ede2 | AdAG A2Bc | B2B2B2z2 | e2ed Bd | "Bb"G2 B2 | B2B2 | d2c2 d2 | c2e | fgf edB|Ad^c dcBA | d4 ede | ~f3 edc|1 AGF G2:
|
<end>
```

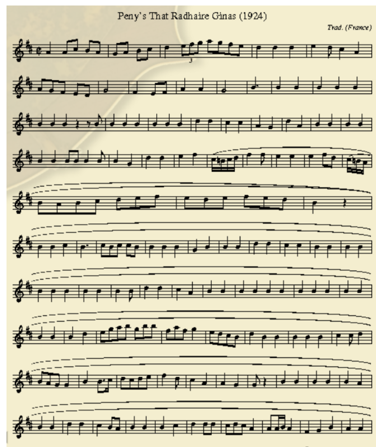

Figure 2: The ABC format(2) for T=0.5

```
<start>
X:24
T:Ailto:galouvielle@free.fr
M:2/4
L:1/8
Q:1/4=1150
L:1/8
K:G
D2 F/F/E/E/A/E/ |
G2F/2G3/2c/2|d/2d/2e/2|d/2e/2=e/2f/2d/2 | d/2e/2 f3/2g|f2f2 bag|d2B/2B/2 A2e2|e2c2cB|AF G/4G/2A/2 B2|d2d2|e2e2e2 e2f|"G"F/2^F3/
4B3/2A/2c/2 c2|
[1 A,G|GD/F/ A/B/ c/B/|
e2eg ef/e/ | f/e/g/e/ d/d/ dB|1 cd/e/ dd ed/2=B/c/d|
A2 AABc|BGB2 | d2c2B =B2B2 | B2BB B3B B2 | B2E2 | c4c2d2 | c/d/e/f/ |
fg f/g/e/d/ cB | B/B/B/c/ d2e | d2B F2 :|
P:Variations Pi_I, #53
R:polka
H:(158-195
M:2/4
L:1/8
Q:1
C E | ABG FGAF | Bcd efe dBG ^FGAFA | GFD G2g ||
a2ge df f2ge | dedB eAB | BAG AFA | AGA |1 B3c2 d2c|d2gd d2c2|
eAA BAG | (B2A A2F2 | FAGE FBA | (ceGA cecec|_Bc A>B | A2B2 GBG | F2 D2 |: G2 G2B | cBG cB | BA2 | G2EF E2D EFG | F2G2G2 G2 |
(3cc(3B/c/d/ gg/f/ faf g2g>e | d2g d2e | g4d4 |
GFGAG F2F>G G2AG | FGE G2F2 | AFD/2FD | A2B2 || G^FG G4 | Bd^c2B2 | BddB | B6 | ABcAB | AGE Fd=EG | AdfdB | c2c2 :|
P:var
E E2Bg | g2bf efge g2FG | d2d efAf | e3cd edBd|cBcA B2B | B2G G2 AB | ccBA dedc | AABc A4 B3
```

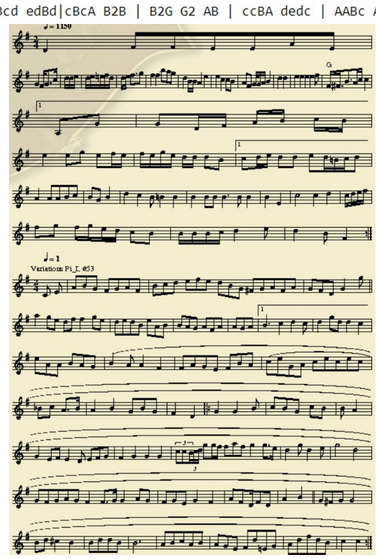

Figure 3: The ABC format(1) for T=1

```
<start>
X:14
T:Couther", #30
L:1/8
K:Em
b3/f/d/ e/A/|
AABAG A(f2)f2|d2Bc BAF|1 E2GD G4:|
"F7"AGFA | B4 G4A2|dedB dBdB|ccAF GABd|fddc (3F(3GED|
fgaf gG|(3FA d2dfd|"G"GB BB||
P:Variations
"C"c=Bd/2d/2c/2 | B3|c2A BB|| c>d B2 | c2z2 :|
/2B2z2B2 | ABc dcdB | f>e d2d | {c}udd cB | cd~e2 d2Ac_B | Gc:|2 ABAB | c>B c<ABc |
cdec dB|GAB G2G | ~E3 ~a3f|edce | e3d | dfed cF~F2|B6 | BBBdBG G2GB | B2BG B>B GABc (3defd | c2dd | c2A2 :|2 GAB B2AB | cdec (3
ABf | ega g2fe|2d2d2|dfdB d2A|1 efgfe fedce | ceAc ded2 | d3z | c4- | d3-f2z z cd|efd cBA|B2F4:|
P:Variations:
(c/d/)(3B/c/c/d/ Bc|B3-|B>B Bc|dc A2 :: x3 | ED2D (3c/=G/F/ G2||
<end>
```

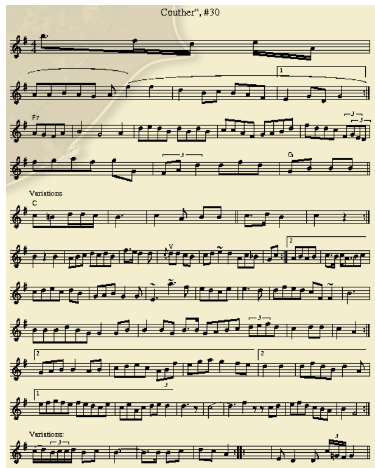

Figure 4: The ABC format(2) for T=1

```
<start>
X:16
TfHowpy8G, Losqict:m|
K:A
(=Bfd/g//|:afcE)g2atrhans
id2"G4F3.f|d2bid>G2A2 B2G/2B/2  | "D7(dfd=e4- AGFd=c!bbbb] ze/2 f/a/f/ 'g23_>g_>g a(f2!b2]f2g2f4AF|G6e:|2 !te3ga>b>uate),2 ] .f
0)
g2d/2)3/B/>^A||
B2|-E2 e=df :|[[Pa3t}!ceg ({fluxcalotie +rixBd, #39[N,2"JRtP:me{cqa_obfa C\batT
dO~BG a3D[F ABcB:|
g6 |
I7"i][Gd6B2EB BABd|e<^=u[Ec/c/: fAg)|.f2'Le/
V:Cse's p\'bl-ff..LdxF}(c G2E EB ^A4|"FD "F#F]8]
K:F
"A_d Ac'}[GA (e)^cd e2 fg|gf:|
<end>
```

Figure 5: The ABC format(1) for T=2

```
<start>
X: 1
T:Cransi maily Cother tountain Hornpipe, The
T:Pilindo
M:6/8
L:1/8
K:Bb
c3d | e2 g>d |1 ed cB | ce e2 |
B2 GFG | =GG)GF EABG|B2A2 c4- | G2B cBA|GBA GE (3B^c | defe d2Bcd :|2 B4 z^c|dcB A4|
w:Staint serry ua Mheas.
Z:id:hn-air-soch-8
M:6/8
K:G
B3 cde|"C"e2c2 d2e2|c2d fgf|
dg/2f3/2f/2 f3/2_e/2d/2e/2 c2!+!d2 de|f2d2 fd b2|e2c2 c4 ||
|: dgdc f2 ||
|: "FI"F#2"Gm7"d2e|ab/a/g/ a2d |
d>B AF|G/B/B/ A2:|2 "D7"A2|"C=e_e|\
FD GB|de|df f>g|fa fg|ag/f/ ge|=e/f/e/f de|c/B/c/ Bd|ce ed:|2 G2 B>A|G/2 E3/ Fd|"G"B>A g2|e>d d/e/d/e/d/|ef d2|ea af|de/d/ ce |
Leec | df/e/ dB | B>B cd | BAG A2 :|
w:Il?o
K:Bb
GBdB ||
<end>
```

Figure 6: The ABC format(1) for T=2

We found that when T=2, the generated music sounds much more random than the others, and also it will have higher probability to fail in converting to midi format.

For T=1, the generated music looks great, but when we reduce it to 0.5, the pattern of the music became relatively fixed, some times most parts of some music are repeated characters, this could be caused by restricting on the random sampling when we reduce the T, which suggest that T=1 could be a good option to generate the music, but we also got very beautiful music when T=0.5 or 0.3.

2 Loss and Accuracy

In this part we present the final model with optimal parameters after long time of tuning, and the figure of loss and accuracy for training and validation data is shown below.

You can see from Fig.11, during the training, the validation set over-fits very quickly, while the training loss just keeps decreasing smoothly, in order to prevent the generation music from being too similar to the training set, we used early stop(when validation loss keep increasing for 6 times, the model will stop training).

You can also see that, for both loss and accuracy graph, the validation set is better than the training set at first, we think the reason for that is during the first few epoch, the model didn't learn enough information about the music format, so the character it predict based on the previous character is pretty bad, due to the large number(90%) of the training set, it will have much higher probability to have a worse performance on training set than validation set, but after few epoch, the model learns enough information, and the character it predict become more similar to the pattern of the training set rather than validation set, so the loss decrease much faster than validation loss.

There is a very clear curve on validation set, we think it's because the size of validation set only occupy 10% of the total data, which make it more fast to over fit since the size of training set is 9 times bigger than it.

The lowest loss for training and validation are:1.261 , 1.617; The highest accuracy for training and validation set are:61.8% , 52.1%

The training loss could get lower, since in this case we only trained for 50 epochs, and only 90% of the data were used as training set. According to the graph, we have reason to believe that the potential accuracy could be higher than 70%, if we used the whole data.

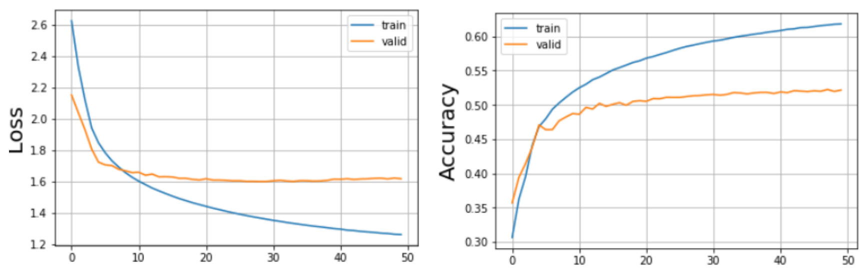

Figure 7: The Loss and Accuracy for training set and validation set

3 Numbers of Hidden Cells

We tried different numbers of cells inside one LSTM layer for 50, 75, 100, 150. Fig.12 shows their results, from which we can find that with cells getting more, loss decreased from above 2.2 to 2.0 at some epoch when cells are 150, however, after that low point, validation loss displays a strong over-fitting problem, although it is still the lowest one, 2.17, at the end of training.

Increasing complexity of model is as known as a good way to learn hard pattern with side effect of over-learned training data. Our experiments completely demonstrated this principle. More cells inside LSTM or GRU layer, more complicated feature can be extracted from input characters, and most importantly, more 'memory' from previous sequence can be reserved because of a large size of hidden states.

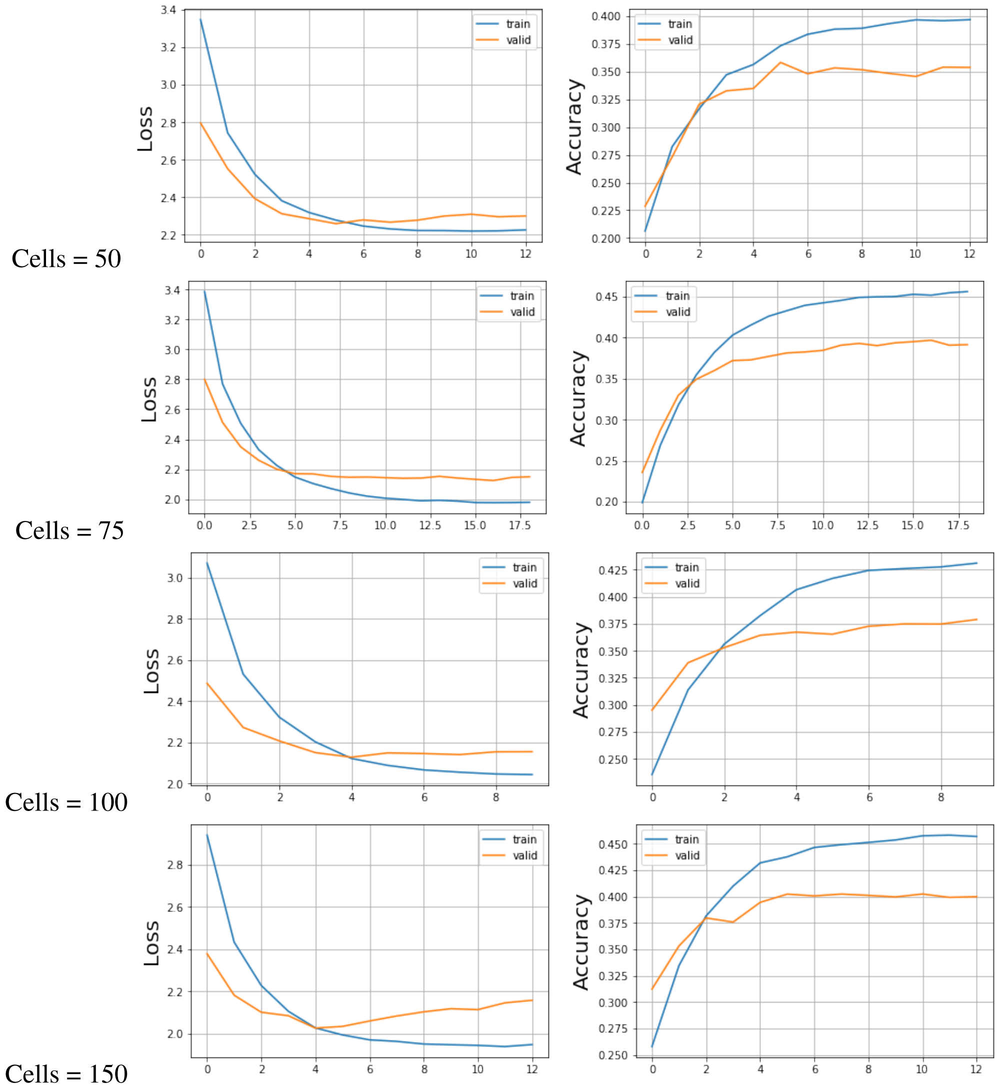

Figure 8: The Loss and Accuracy for training set and validation set for different numbers of cells

4 Dropout

Dropout ignored neurons during the training. And these ignored neurons won't be considered during forward or backward propagation. More technically, At each training stage, individual nodes are dropped out of the net with probability 1-p, so that a reduced network is left. And the reason why we needed dropout is to prevent over-fitting.[1]

```
<start>
X:27
T:Trilorgiog aire Let Hitiontive.fh
Z:id:hn-reel-16
M:1/8
K:D
E4|Ha DE ::
c2|A2GB|1 GABc|efgd|cAGF|FGAGF|EGA|BAGF ~E3 ~E3:|2 D2B2 BE|GE DFE|DEFG|
Adf dfge|dcdc Bcde | efbaf|ed g2 aga||
|:E4d4d4 g2 | egef |1 efd d2f | d2d2|
(3d2 A>B e>d dgeg :
g2e2|1 (3efgd d3/2 |
d2 AdAF|1 AG>A2|AGEGA ac'|egegef|
"F2 | B4-
f3ba boud'egag Bed|1 BdgG|F
AGB FGG GAB|
e4Da :|
<end>
```

Figure 9: ABC notation and sample music when dropout = 0.1

```
<start>
X:416
T:Fomo, nalledar
D:DogIjig? pe slans  Stevick
H:Also #61.on Bulang.2
R:Dan Bor
T:Rard sord Crolanatbindhielllla)
M:2/4
L:1/8
K:Acdd (3adB| A>B B/G/A/ |
~f3gfegf e dc||2BA|dcBA ABGF|~B3B ABd cBd|e2ed2g | e2f3/2 gag|eddzf gafg|
FGFG cBf|g2f2e fede|fedefgb fa ed^cf:|2 gB~e3|fB~fd | c>d d>e |
c'2c2e||
|:(Bde).dec|1 +b"ef gbga (3Bde|fd efg|f2eef (3gfged|
D2edcd|BAc AFFA |1 G2AB|(3A
e2fg gf edc|dfeBAG|
V:Goolle uk
Z:id:hn-hornpip Starlane, Thilledret Mar
C:Areathe
Z:id:hn-raelles
R:hn-21
M:9/8
K:A|1 Bc/A/B/|BG G>B|de dB||
P:varka
S:Gha-lan Alawig? pe-88
M:C|
K:AD
|:G2 AFA Gcde|fAe2g gge agede|fdGA^FGA AGA AFD|1EFDE AGAce||
<end>
```

Figure 10: ABC notation and sample music when dropout = 0.2

```
<start>
X:149
T:Proste.
M:C|
K:G
G/(3/ge/d/d/ ece | e/d/ fe|G/e/) | g3a ged/B/A/G/B/|
|:Bde ecA "E7"dcd|egeg/.G||
w:ceol do Chonne.
S:"Tedervation G?mellite.
H:Seant (B<~E3A/2G3/2 A3/2G2|
P:Brance
W:(Jabion Ginc
Z:Thocor
W:Jidande de McKevence
Z:Tranielle@frea ba Met@htion ceilla dilly at Malle
R:holka
C:Tinelle@fr
M:C|
K:D
AGFE EDD|g3 ~A3(F3 G2A2)A2|_g2B c2z2z2:|
~a (3ecd | Bd Bcd|edB|GBAF DDFA|1 d2g2 bagabge afg | fed ded ~g3 | efgfae eg~g4g2|edc GE E2:|
|:Bcd2 d2de gece|d6 ! |
c'aga gfg|a2gfg agf | ff e2cdf|edfd (3edcBA=A|1 GEFGE ECEF GAF|AddB "C"G2FA|dBG FDFABc | ~AFAB|1 B2G | E2E DDD2:|
<end>
```

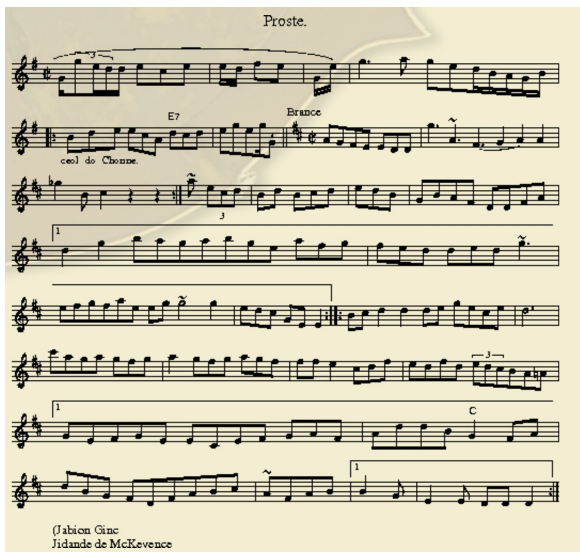

Figure 11: ABC notation and sample music when dropout = 0.3

Based on the summary table and each loss and accuracy plot, we can get a preliminary conclusion: validation performance gets worse with bigger dropout rate after dropout=0.1.

dropout rate	0	0.1	0.2	0.3
accuracy	40.2%	42.7%	41.8%	41.3%
loss	1.99	1.89	1.92	1.94

Table 1: Validation Performance for different dropout rates

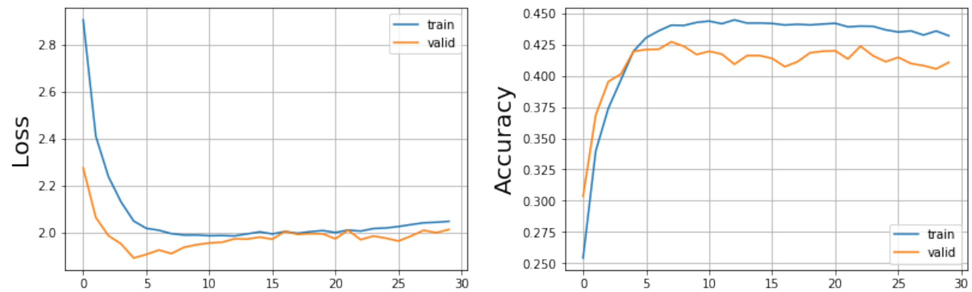

Figure 12: Loss and accuracy when dropout = 0.1

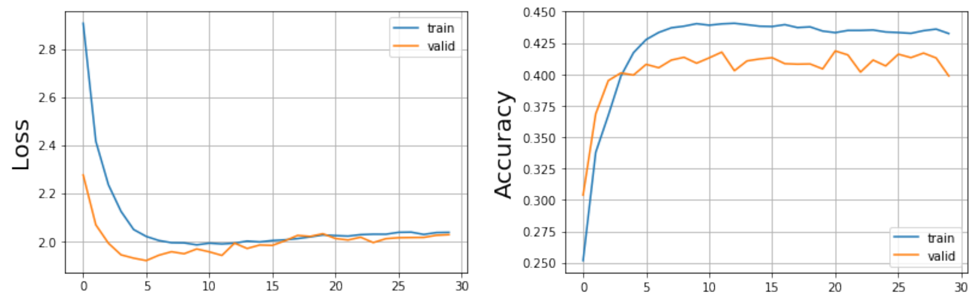

Figure 13: Loss and accuracy when dropout = 0.2

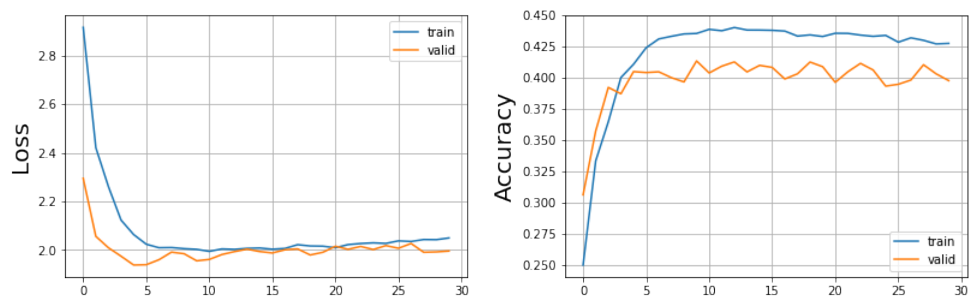

Figure 14: Loss and accuracy when dropout = 0.3

The reason why no improved performance was observed after 0.1 dropout rate might be our model weights are initially evenly distributed, namely, no super strong nor weak weight. Thus a small dropout rate is enough.

As for over fitting phenomenon at similar epoch for all dropout rates, because our maximum iteration limit is big and we applied a loose early stop criterion, our model is intentionally a little over-fitted. Other reasons could be too large learning rate and too small training set.

There was no big difference in running time among these three dropouts. They all cost 58 68s to train the model for each epoch. Therefore, the dropout didn't significantly affect the training speed. Since the result was better when we choose dropout = 0.1, then the next steps all based on this dropout rate.

5 Different Optimization Methods

After comparing with three optimizers, we realized that the Adam(loss = 2.1) and RMSprop(loss = 2.2) don't have very significant differences, however, based on the same learning rate(0.000005), to see the pattern of each loss clearly, we choose to set threshold to 5(it will stop when the validation loss keep increasing for 5 epoch), so we can see a clear pattern of how each loss changed, it seems that Adagrad(loss > 3) offered relatively worse results, but we can see its loss is still decreasing, while other two have already over-fit.

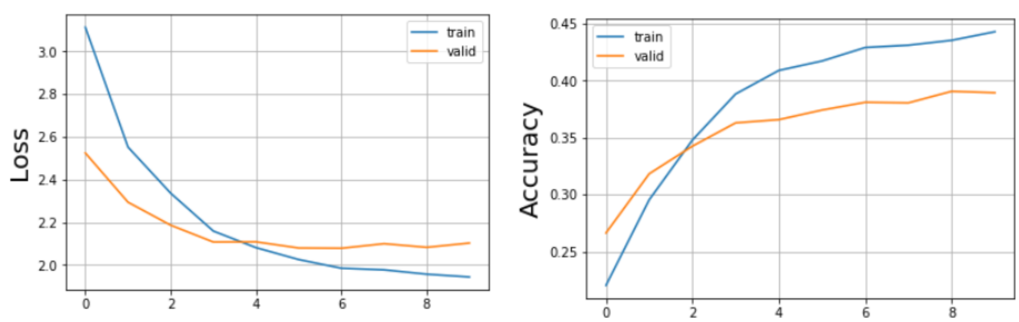

Figure 15: Loss and accuracy when optimizer = Adam

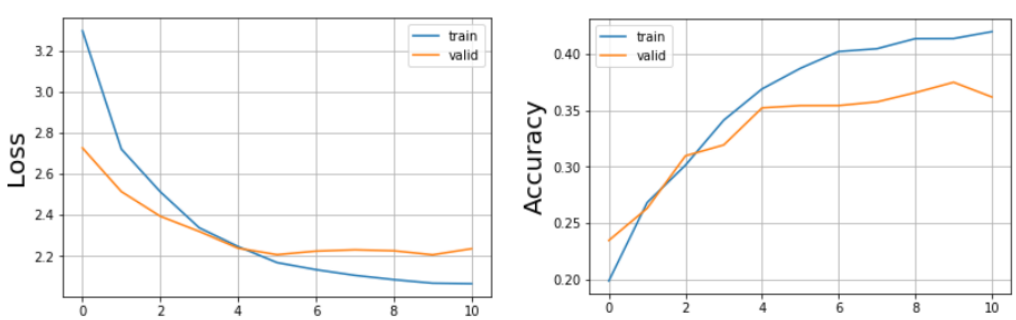

Figure 16: Loss and accuracy when optimizer = RMSprop

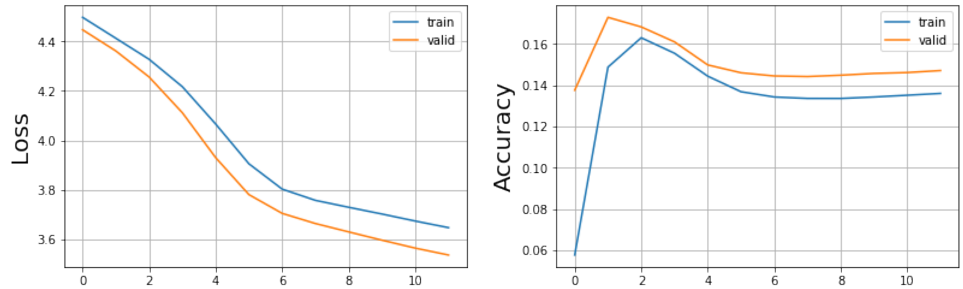

Figure 17: Loss and accuracy when optimizer = Adagrad

Then we changed the learning rate(0.001) for Adagrad optimizer, it achieved much higher accuracy(0.46) and lower loss(1.8).

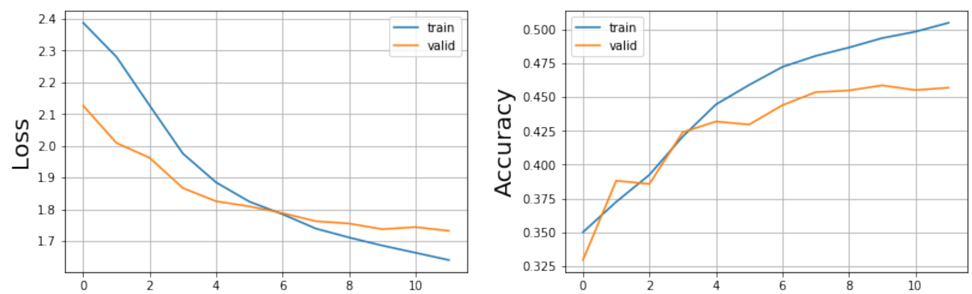

Figure 18: Loss and accuracy when optimizer = Adagrad and learning rate = 0.001

However, theoretically the performance for Adam should be better than Adagrad and RMSprop, in our case, Adagrad perform way much better than Adam and RMSprop, this may due to the parameters we chose, since till now, we have already change the value for hidden unit, dropout, batch size these may affect the result, it may also affect by the characteristic of the data itself, but so far we still can't come up with a very specific reason about what may cause this.

6 Feature Evaluation

In this part we will show the features by performing the heat map from the model. Basically we are showing the activity value for each cell when it generate a new character. In our previous step we already generate a music with 1000 character, here we will use the heat map coming from this music. It doesn't have ¡end¿, since we only need 1000 characters, so the generator will stop when the size reach 1000.

According to the previous step we choose 150 as the best hidden unit number for our model, so there are 150 heat maps, each of them have different patterns, most of them doesn't provide a very clear pattern about how they are activate during the generation, luckily we still find a heat map that contains many useful information.

The ABC format and the original music format is as below:

```
<start>
X:24
T:Ailto:galouvielle@free.fr
M:2/4
L:1/8
Q:1/4=1150
L:1/8
K:G
D2 F/F/E/E/A/E/ |
G2F/2G3/2c/2|d/2d/2e/2|d/2e/2=e/2f/2d/2 | d/2e/2 f3/2g|f2f2 bag|d2B/2B/2 A2e2|e2c2cB|AF G/4G/2A/2 B2|d2d2|e2e2e2 e2f|"G"F/2^F3/
4B3/2A/2c/2 c2|
[1 A,G|GD/F/ A/B/ c/B/|
e2eg ef/e/ | f/e/g/e/ d/d/ dB|1 cd/e/ dd ed/2=B/c/d|
A2 AABc|BGB2 | d2c2B =B2B2 | B2BB B3B B2 | B2E2 | c4c2d2 | c/d/e/f/ |
fg f/g/e/d/ cB | B/B/B/c/ d2e | d2B F2 :|
P:Variations Pi_I, #53
R:polka
H:(158-195
M:2/4
L:1/8
Q:1
C E | ABG FGAF | Bcd efe dBG ^FGAFA | GFD G2g ||
a2ge df f2ge | dedB eAB | BAG AFA | AGA |1 B3c2 d2c|d2gd d2c2|
eAA BAG | (B2A A2F2 | FAGE FBA | (ceGA cecec|_Bc A>B | A2B2 GBG | F2 D2 |: G2 G2B | cBG cB | BA2 | G2EF E2D EFG | F2G2G2 G2 |
(3cc(3B/c/d/ gg/f/ faf g2g>e | d2g d2e | g4d4 |
GFGAG F2F>G G2AG | FGE G2F2 | AFD/2FD | A2B2 || G^FG G4 | Bd^c2B2 | BddB | B6 | ABcAB | AGE Fd=EG | AdfdB | c2c2 :|
P:var
E E2Bg | g2bf efge g2FG | d2d efAf | e3cd edBd|cBcA B2B | B2G G2 AB | ccBA dedc | AABc A4 B3
```

Figure 19: The ABC format for T=1

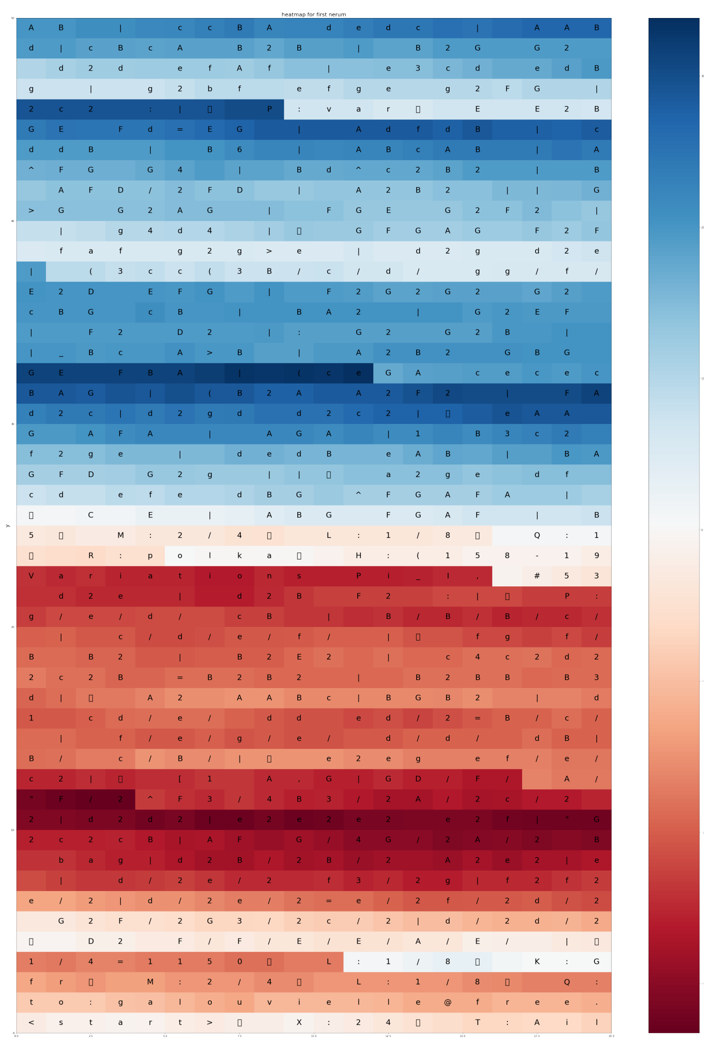

Figure 20: Heat map

The heat map showing here is coming from the 21th cells in the model, red represent negative value, blue represent positive value, the characters are aligned in the order of bottom to top and right to left.You can see that, our model generate this music in four parts ”header,body,header,body”. And the heat map actually have a very clear separate part between positive value(blue) and negative value(red). By checking the character along that middle area of the separate line, we found it’s ”Variations Pi-I, 53”. According to the ABC format we showed before, this is exactly the line between the first body and second header in the music. We can also find that, the white area at the middle of these two part actually contain the character for the second header, when we check the first few line at bottom, we also find that the lighter part corresponding to the first header.

According to these information we found, we can have an assumption that, the neuron we found separate different part of the music by having different activation, and also when it comes to header it will have less activation, but will have higher activation when it process to body, every time when our model generate to a new part of the music, this neuron will have a jump of activation value. Which will be shown as a lighter line or a dark line in the heat map.

The music format for this ABC format is shown below:

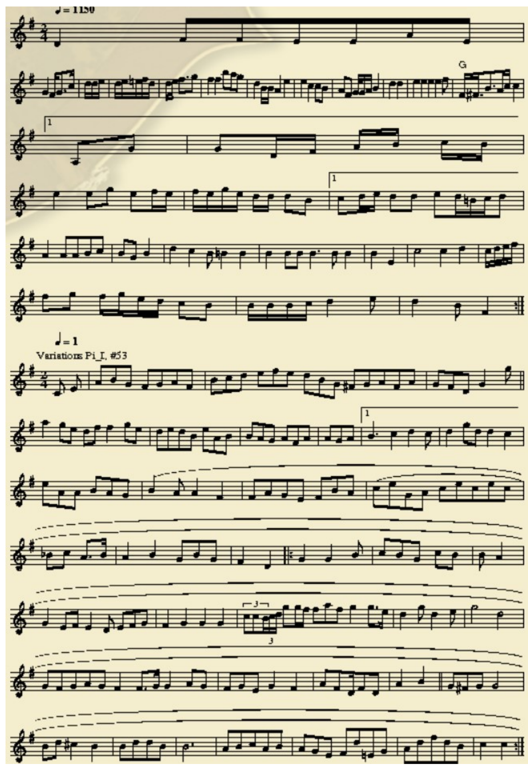

Figure 21: The music format for T=1

7 Contribution

Actually, for this project, every person in our group contributes a lot. Every one tried to complete problems and at the end we combined all results together and nd the best one to show in this report. Therefore, the contributions of each member are almost equally.

JiaweiLi: Designed the pre-processing part for data. Helped design the model and test tune parameters. Also wrote corresponding part in this report.

Zhiling Liu: Designed the generation part. Helped build the model and test tune parameters. Also wrote corresponding part in this report.

Deling Li: Designed the training part for data. Helped build the model and test tune parameters. Also wrote corresponding part in this report.

XuefengShen: Helped construct the model and tested different optimization method and summarized their performance.

References

[1]https://medium.com/@amarbudhiraja/https-medium-com-amarbudhiraja-learning-less-to-learn-better-dropout-in-deep-machine-learning-74334da4bfc5/