CSE 253 PA4: Character Level RNN for Music Generation

Brandon Ustaris
Department of Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093
bustaris@ucsd.edu

Brian Preskitt
Department of Mathematics
University of California, San Diego
La Jolla, CA 92093
bpreskit@ucsd.edu

Brian Wilcox
Department of Electrical and Computer Engineering
University of California, San Diego
La Jolla, CA 92093
bpwilcox@ucsd.edu

Derek Tran
Department of Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093
dtt018@ucsd.edu

John Clara
Department of Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093
jclara@ucsd.edu

Abstract

For this assignment, we explored the power of Recurrent Neural Networks. This was done through training a basic RNN model using characters extracted from a music dataset that was provided to us. After training, we ran the network in generative mode in order to ”compose” music. We were shown how RNN's are able to process sequential relationships like a pattern of musical notes one after another. This is contrasted with our previous assignment dealing with Convolutional Neural Networks which assumed the data was IID and temporally independent.

1 Training Network

In order to train our network, we first had to familiarize ourselves with the music data given in ABC format. After we figured out how to properly read in the data and give it to our model (the structure of it is described below in the Generating Music section 2a), we were able to train our network.

2 Generating Music

(a) The first task was to generate 6 musical pieces. Two at $T = 1$, two at $T = 2$, and two at $T = 0.5$ where T is the temperature parameter. Higher temperatures will cause softmax

outputs to be at closer probabilities to each other making our model less deterministic. Lower temperatures will cause softmax outputs to be farther in probabilities from each other making our model more deterministic. Our model is described as follows (parameters, etc.):
- 3 layer network recurrent neural network
- hidden layer with 100 units
- softmax output layer with cross entropy loss
- sequence length of 30
- temperatures of 0.5 to 2

In general, many of the songs generated from our model were not in the correct format (the biggest problem being that 'K:' would be missing from the header). However for the songs that were correctly generated, we could see slight differences between the songs generated at different T values.

Earlier we mentioned that lowering T will cause our model to be more deterministic while raising T will cause our model to be less deterministic. If we take our songs generated at $T = 1$ in figures 1, 2, 3, and 4 as a reference point, then we can see this pattern actually occurring.

At $T = 2$, we had more malformed songs because now the model considers all characters at more equal probabilities to be the next occurring character, which means incorrect characters are more likely to be selected compared with the model at $T = 1$ or $T = 0.5$. Also if we look at the songs generated at $T = 2$ in figures 9, 10, 11, and 12, we can see that they have a lot more variation in the rhythm and beats. In a sense, the music being generated is not following the general music style shown to the model when we trained it. We could think of these songs as the more unique, strange, or innovative songs.

At $T = .5$ for the songs shown in figures 5, 6, 7, and 8, we can see characteristics of the songs hinting that the model has become more deterministic. The model now selects the next character favoring the most likely characters a lot more than others. This is obvious through the songs that are generated as the music scores show songs with notes that are very close together and barely vary at all in terms of note lengths. This makes sense since in music, runs (a sequence of notes that are very close together) are very common while notes of the same length tend to occur after one another more than any other note length pairs. So this time with a lower T parameter, we can think of our songs as the more generic songs.

```
X:1
T:KnG
E>E c/2B/2 e2|
c,/M:1/16
K:G
aG|ecf6|
(Afdc) (3Bge |
aaga gdeg | gathe,o(3f ceed efdi | e2e2 a2dB | B)GB BAGF | C2G2 EAFG
| A2Bc dcAB | d3d d2d2 | G4 ddBd | c2Bd dFG2 | FE,2B,2 B2 | "C"f2 cec
|| cc e2 AG /4/A/ A/B/G2|e2d fg2 |f2 g2 e>d |
T:fc !erefB lai>. sa3hel
R:hn ~Calonee B2Bllass, dfepeel
D:Perist
```

Figure 1: Music text for first song generated at $T = 1$

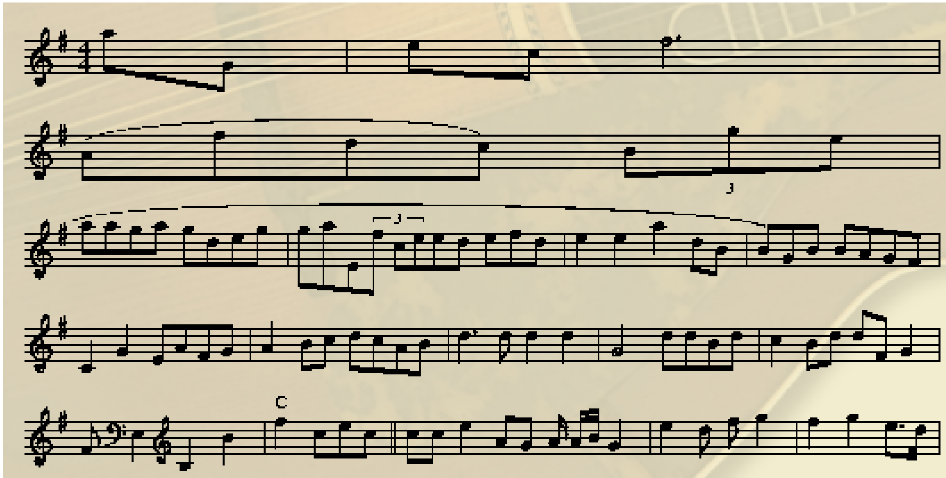

Figure 2: Music score for first song generated at $T = 1$

```
X:MO2 fthins Mo ter Pist ons neto horr, tz-lenfoll Etie
R:Branle be ,p pan ur ConO:11ta20
M:6/8
K:C
V:1
cA|FG de/f/ bged dBc||
F
2B3 | GFB |
B2B B>(c>c | c2B B3 BGG |
ad- ddf=ef | d4 dBAF | G2e2 edF2 | BcBc Bcdd | G2 EDFd | edbaf | daB2
| medc2 | .BdcB | G2F2 D2 :|
```

Figure 3: Music text for second song generated at $T = 1$

Figure 4: Music score for second song generated at $T = 1$

```
X:10
T:Dimnsedre Bouseoueboige
7:|AGBG FGFz|2BcB2B|A2F1 FG,2G,2e|dAF|A2Bgfd|B2A2 B2cB|B2o,>eM(e |
(e>d) ce | ef ef | dB/c/ d/e/g/T/dBc O BA || (=6 cB | '2a2 de |
(f2e)- edcB |
A2AF F2fe | (3ded BA | BcBA | FAGB GBAG | E4/D/ G/G/G/B/ B/A/d/B/
c/d/e/e/ | cB AG EG/E/ | B3c | cw=e | edc | (B2)
f2 | Z:id areatlliB,e3A, fe Gl
|:GEA B2e|c3G ABdd||
gag>d edBd|eefe (3dBB dBdd|cBcA FGEF|F3 de dB|Adg3f3d/2/4:|2::
r2A Bdd | BAc BcA | BBG A2f | e3 ecd |d2dg e/fg | f2A BF :|
P:Varonei-tol"o7tKeelt (oncHnA Mo]
C,Cnc2tid2znts:
D:Dopallenc hes Riche
R:folka
,4B, - 2f gat4-le FAFlle ded2 RdickeRon C 1647
Z:id:hn-1
M:g/4
K:Em
FGB f2ef|g2af B2dB|1 c4=F2D>A (3BAB | BGGG FGFG | FGEG d2AB | BGFA
F2ED |
AFA c dB2d | d2A2 | BBGB | GzKed | B2 GB B2 |B "GA>B AG GE/2 | GGd
fce | gte | fdB Bz Ad | fd c/d/B/c/ d/c/B/B/ .G2 | F2 F>B d2f | cAc
B2B | (G"GFh BA G2 | g|fag dBB | B=BAB Bcd |
fde ~g2 | f4 edBG |
cAAB B2B2 | B2AA BcdB | GAFB A2 : cB AC | E2-2EG>O2 :|
```

Figure 5: Music text for first song generated at $T = 0.5$

Figure 6: Music score for first song generated at $T = 0.5$

```
X:a2
M:3/4
K:D
P:Aveab
|:AG|DAF GED|DEF E3F|D2G2 F2GG|AFAG a/cde|dAB cAG|AGA FGB|Bge fed|Bae
e/dB|eaf aff|fgf gfe|dea ged|caf gfe|~a3 dfe|~g3 B3:|
```

Figure 7: Music text for second song generated at $T = 0.5$

Figure 8: Music score for second song generated at $T = 0.5$

```
X:1 nathe8
R:B4ripi, The
O:France
M:2/4
L:1/8
K:F
"faggree | !breatg/2|gege gfe|e2d e eld, M,AB,,G, | A,2A,2F G2F2 |
~G3 e2d | ddc Bcd | cd ef ee |
BB c2 B/ cB | e2g |bag2 ||
3B2 | BAB2 Bd | = Adc AGd | BFE fAD | e2f A2f |d2c2 | f2(gf 50d |
edcd'2 e4d | f2g2 (3agf | (eLef ed | e2 g(g(3ga |:gBf g2f>e | c4 2B
dB | B2 G2 ef | fpB/A/B/A/F/cf
```

Figure 9: Music text for first song generated at $T = 2$

Figure 10: Music score for first song generated at $T = 2$

```
X:3"
K:G
(3Acd |
Ac dcB | cBA AFA |
GFA d2c | B(AB/=B/c/c/d/) | gf gf |: d2 d2 e/2B/2|C.d/.e/g/g/a/g/
.g.gs Pour torse ChaiE|~Eg~a2|gdfe dBGA|eGAG AGFF|
3DFE EGGE|1 GEE E2D :|
): fa cond 2zury ees Favoth!D FEone, The
R:rec ette t(Rauder.sA .
dCoe/2 bemApiad
":)urcd, The, stree c'airathorie oneldy Muzurka
Z:Transcritte:|ogdA dfd|e2g efg|aAf gfg|e3d fffg|agfe dffB|dBAG
EGGA|^dC.d2 BBge|d2c2|B4B2d2dc|e2d2 e2d2|c4-c 1Bc2 6efe | A4 AF D2
:|2 AGF AAF a3 |edc2
GTf4 | dABA e2Bc | dcdc A2BG | GG^G A3 AFE | FED E
```

Figure 11: Music text for second song generated at $T = 2$

Figure 12: Music score for second song generated at $T = 2$

(b) For the model we trained in part a, we plotted the training and validation loss vs the number of epochs. The plot is shown below in Figure 13. As is the normal case, the training loss decreased very fast initially but slowed down at later epochs (around epoch 5). We stopped training at 16 epochs because the validation loss had actually reached a small value at epoch 13 and only got losses above that for the next 3 iterations. So to avoid overfitting, we stopped our training and took the weights of the model found at epoch 13. Though in general, the validation loss moved up and down over the epochs, demonstrating the difficulty in accurately modeling/predicting the structure of our abc notation songs.

Figure 13: This plot shows training loss and validation loss vs number of epochs for 100 hidden neurons.

(c) For this part, we varied the number of neurons in the hidden layer from 50, 75, and 100 neurons. Below are our figures for training loss and validation loss vs number of epochs. Figure 14 shows loss for 50 hidden units, Figure 15 shows loss for 75 hidden units, and Figure 16 shows loss for 150 hidden units. While training, we saw that increasing the number of hidden units increases the training time of the network. But it also improved our results because as the number of hidden units increased, the final loss decreased.

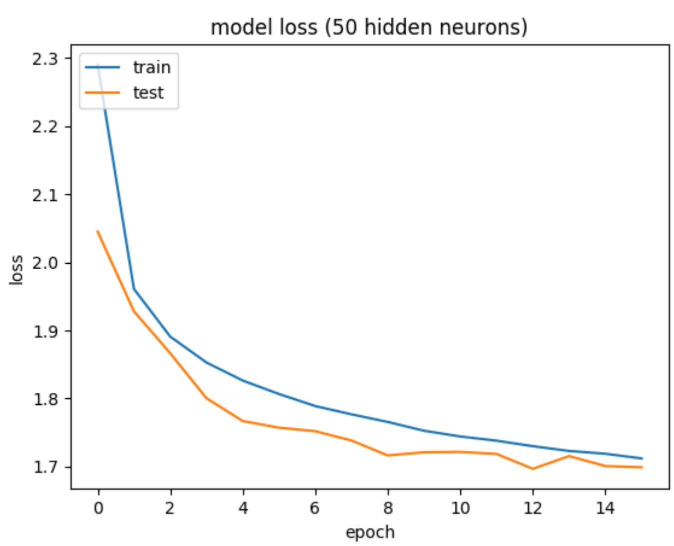

Figure 14: This plot shows training loss and validation loss vs number of epochs for 50 hidden neurons.

Figure 15: This plot shows training loss and validation loss vs number of epochs for 75 hidden neurons.

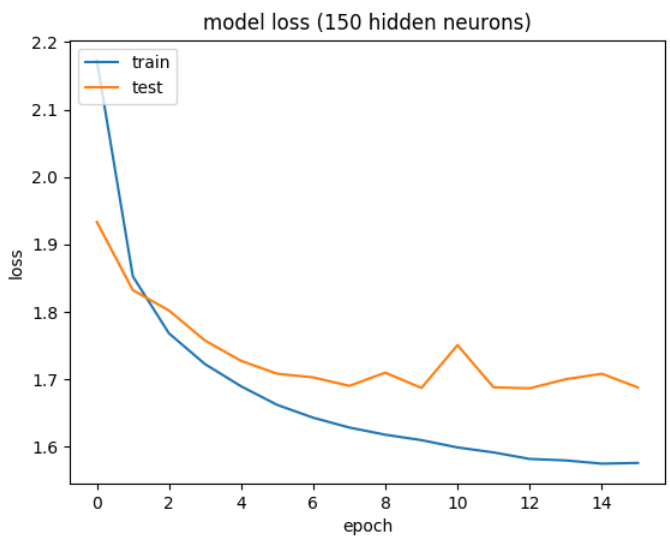

Figure 16: This plot shows training loss and validation loss vs number of epochs for 150 hidden neurons.

(d) Next, we varied the dropout value between p = 0.1, 0.2, and 0.3 and plotted our training loss and validation loss vs number of epochs. Figure 17 show the loss for p = 0.1, Figure 18 show the loss for p = 0.2, and Figure 19 show the loss for p = 0.3. After plotting all this, we generated music for each of the dropout values. Figure 20 shows the generated text while Figure 21 shows the music score for dropout p = 0.1. Figure 22 shows the generated text while Figure 23 shows the music score for dropout p = 0.2. Figure 24 shows the generated text while Figure 25 shows the music score for dropout p = 0.3. From what we saw when adding dropout, increasing dropout increases training speed. From our graphs, and generated songs, it does not affect the results much. The loss increases a tiny bit as dropout increases but the change is not too great.

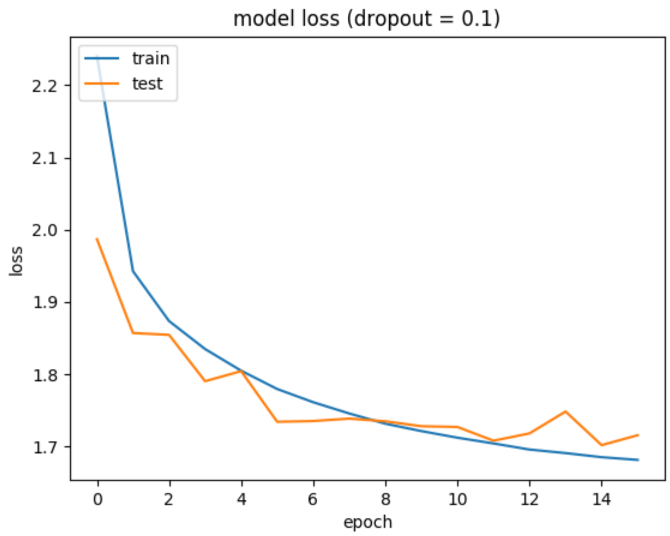

Figure 17: This plot shows training loss and validation loss vs number of epochs for dropout p = 0.1.

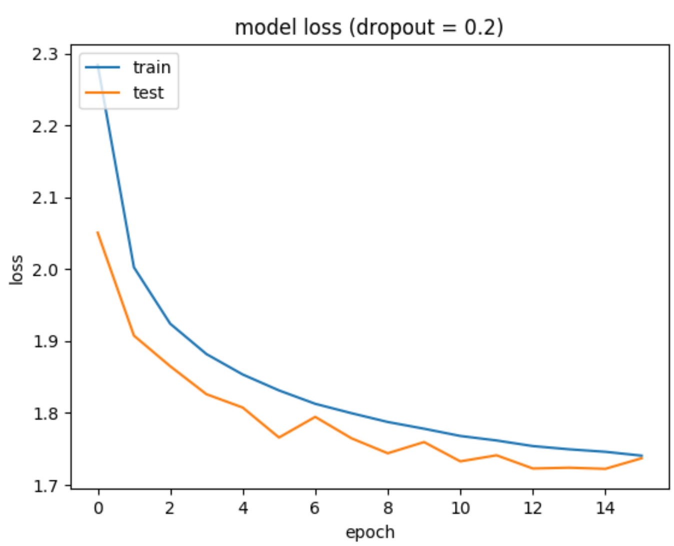

Figure 18: This plot shows training loss and validation loss vs number of epochs for dropout p = 0.2.

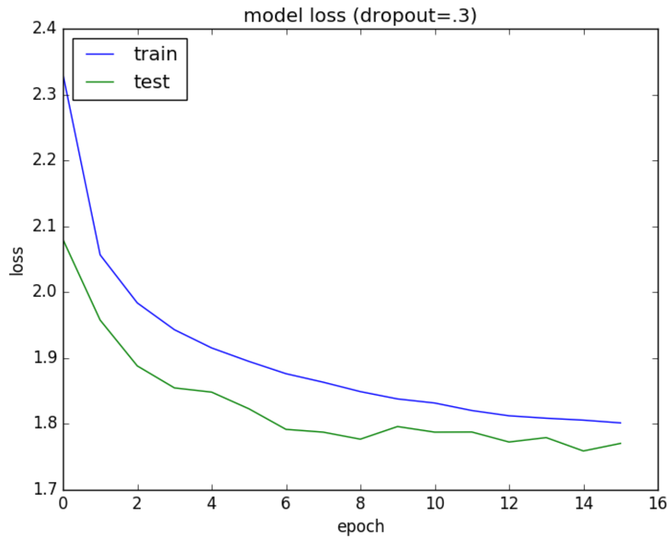

Figure 19: This plot shows training loss and validation loss vs number of epochs for dropout p = 0.3.

```
T:Ranea Brong aich 2ith : ke.fr
Z:id:hn-r
R:tarolSe
Z:ithansselanen dus otheriol ay Lesfint
X:3"
M:C|
K:DG~G2 BGcG|BG~G2 Bdgd|BG~G2 BdcB|cd (3edc dd|ffdc decd|feeA |
ABcc dB F2 | fd e2 ff |
e4 | e2 ed d=BG |BA Bc fgff | ddcd (3edc :|2 c2c>d c>ed |: dBA a2g |
fef gfe | dcB edc | (3ddc ef gedc | d2c2 ]e2f2e2 | cdc2B |
cBd B e d2 | (d>dc) | e4f2 agge | f2g2D2 | g2e2g2 | f4g2 | ef~g2 ae |
ecdB AcBc |16Bf>d eggee cad |(3 ddc cdcG | dccA BF22 | B2G2G2 |
D2G2G2 | c2c2c2 | c2d2e2 | c>d B>A G2 | B2c/d/2 f d/2B/2 A/B/A/B/ |
dcd2 dABA | d>de d2c | d3d | e/dc/2d/2c/2 |ec | AFG | FAGB | G>e g>d 2B |1
B2 B>B BA | B4B2 | B2z2B2 | BGB>c | B3/c/c/e/ | d2e2 | a4 | f2eg |
e3G | (e2c>a | e2ec2d | A3 BcA | B2G G2D | A3G | D2 DE E2 :||
```

Figure 20: This is the generated music text for dropout p = 0.1.

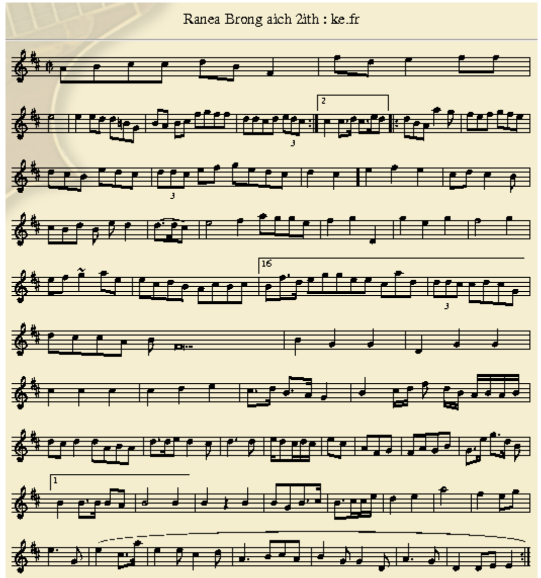

Figure 21: This is a picture of the music score for dropout p = 0.1.

```
X:9
T:Huntingtone Cassrid 99 t eh on cedlle Ja tory al O'C solle
R:Bircka
Z:Transcrtt ot ch pany af ou Jan
s:Tard GeeZ Trans
o:CamhrenC2llADFDan er Kehnar Lam sor
h:"Traplared in 6ra KehnpRalla
D:Cernein  sheraun, peerio.heina Dinsee rond Hour tom Datheine rid glle ouRyenPed
Z:id:hn-palle
P:strite
T:Caj15
Z:Pour toute observation mailto:gasou:inacret in BollaThe Borsei1y 16 ie Redubinr.
Z:id:hn-porki 9o2 4'P ared e  foretStol
R:BihneFoOvr-w
M:6/8
K:G
f | f2d>e | f2 g>f | f4 | e2 Adz2 | B2A>B c2c | ede d^c | dag gff | dde ddA |
AFG AcB | 3BB/d/f e3f| fgd edd|c2e Aff|A(ABAG|F
fecd/2B/2 GB|db,2g, e,2cG|,2,D,2 ,G,2G,=EF,4|[D2]4 "AFG||
G3 F/D/ED|:C2 G,G|D2,2 :|
,2 (A/A/B2 | c2 cc | c2 dB | G2 A2 ||
cc/A/ BA | AA c>B | Fc)G | dcBA | BA Bd | AGAB | c2dFAB | c2dc d2d2 | 3 d2e2 |
a2geF | fzg>e | e^c B4 | cd AA | G2 c>d fecd | ggf efd | c3 B d24 | Acdcd | dedB A3G | GAAA BAcB ||
BA)g edBe | (3fdg gg e2 | dG A4) | F2g2 | f4 (edfe|fg2 f22|dcB d4c|dBc dcd|fed c3(3=ec4|
d3 f c | dc cA |
G/B/d/c/ AF/E/ EE|D2 A | GB=B/ =B2 | A2B2 | cBdB | cBcB | c2g>f | c2f2 | fddB | B/c/e/d/ ge | fa ff | f2e2 | efeA | eeAc | B4
c>d ee-| gF/g/ d/d/A Ac | B2 B2 B2 | BA FF eD |
e2de d4 e2g2 | dBBA FcAB | BABA GE(E2)G | (c3 A c2A | BAF d2d | fad G2D :|2 GAA cAFA| FAF Acc | e3 c2e | "Eg fGedf | 2cf
```

Figure 22: This is the generated music text for dropout p = 0.2.

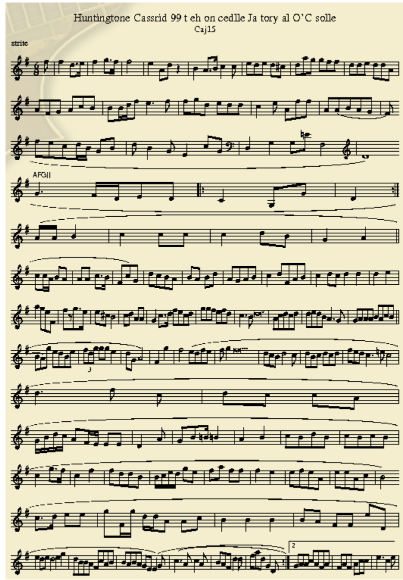

Figure 23: This is a picture of the music score for dropout p = 0.2.

```
X:15
T:Solaic at ne nanlieu terlo)
Z:Transcrit ea/o  et anson
C:C
s:FaraTrnainesc1b9l1i2)1Ornmaleee Brgealo
R:Franteb tvancon obrealdye
C:F
r:1
T:Trad.
R:pileadel, pap2ed3
M:2/4
L:1/8
K:A
c2c2 GFGA|G2B2 B2c2|d4d2G2|c2G2GAB||G2B2c2B2d2c2c2|c/2B/3Bcd:|
d2cB2c2:|A>A4 | c2Az A2 :: G4 :|dc eA cd | (3g2 |
f2 e2 d2 e2 d/2g/2 cd/B/ c/B/B|BB BcBdB|A2 |]BG Bc BB | cB B3|
AG EG | \B2 d2 e>d |e3 f (f/f/ |A/B/)dd | eedg | e2dc | A2B2 |
c2cB2D2 | D2EF A2B| G4 d2e|d2f ^dge||(3^ccB)f Fcc | A2c cdd |
cdd ABF | AFG AGF | |gb dBFA | c2d2 B2BA | A3B A3d2 | c4e2A2 |
G^GCG | 6G2| 2GG3 BE B: ABGE | E2EF A |G2A B3 | c2A :22z2 | :|
2 BBB2 | BdB2 | BBBB B: G2 C (GB A2d | fgf geA | c2d (:|
cdc/c/A/B/ BBAB | ABcF | B/B/G/A/ Fd | dd.B^c | d2c ddB |
B3|"Fd2e2  B3B/c/B/B/B/A/B/||
```

Figure 24: This is the generated music text for dropout p = 0.3.

Figure 25: This is a picture of the music score for dropout p = 0.3.

(e) These next two plots show the loss over epochs for different optimization techniques. Figure 26 is for Adagrad while Figure 27 is for RMSProp. When comparing these two, we can see that RMSProp does better for both training and validation loss than Adagrad. But the validation loss for RMSProp seems to plateau towards the end while the Adagrad validation loss is still falling.

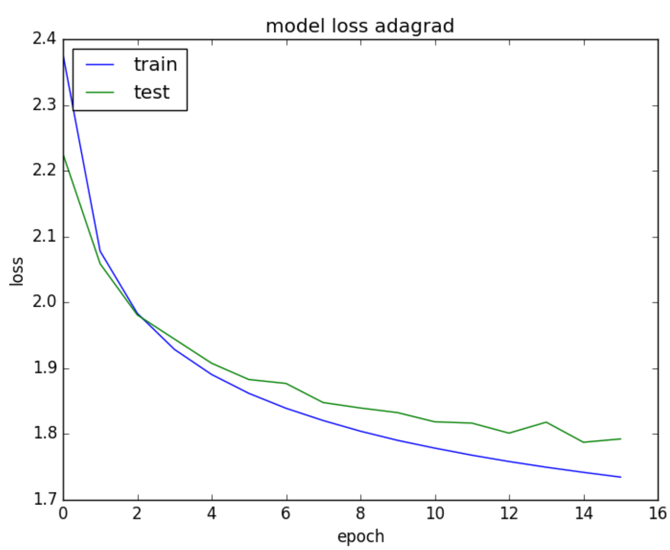

Figure 26: This plot shows training loss and validation loss vs number of epochs while using the Adagrad optimization technique.

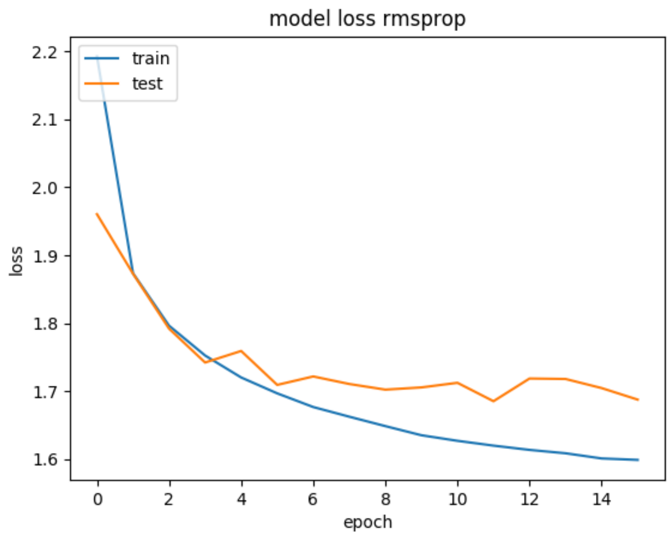

Figure 27: This plot shows training loss and validation loss vs number of epochs while using the RMSProp optimization technique.

(f) We ran our network and received the following text when creating the header:

```
ee also #17, #59
D:Bothy Band 19m4os
C:Trad.
R: fontedtillbirrcAmentelon"3dc -1sy
0:6/4P6
e: bdeis
R:carolan
Z:Transcrit et/ou corrig? par Michel BELLON - 2006-02-2rgoneet Bwuth Brance
C:Dunde 2o1radigi bon S9sKent of rvestolk1
C:folp A160cSeyDr017che louchernd Dintladd, The
R:
`
X:"paled pe.fto th e list andileso
Z:stanche Hamourt onpeerad
Z:id:hn-jig-i3
M:2/4
L:1/8
K:G
`
X:e8
T:EiscrRolen, #7
T:Leimlentare de bussone
8:Prip7to:alto #1i0t.
R:R-hr.t3
M:m/4/8n: Ales :1/8 16g6 1ouvalsoo
C:? of SStomen
T:Hitiledslo
S:Carney ouelss o5 "Res de gis
R:bolnce
L:1/86
K:Fr'ranal Glangel
Z:Pronchello th Mestedovetabr
R:aid
```

e : Choneour t

Figure 28 shows the heatmap resulting from the 52nd neuron firing. It is unclear which feature this neuron has learned. We notice that this neuron turns off during parts of the header which are long words. It also turns off whenever there is a colon. We guess that the feature this neuron has learned is that certain parts of the header must be short meaning that there should be a new line character coming up soon.

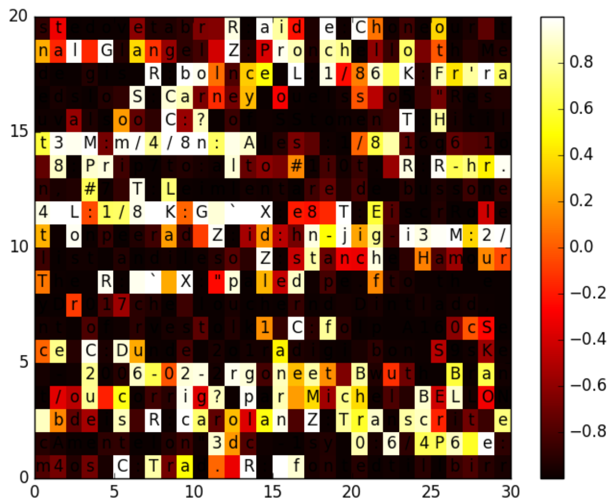

Figure 28: Heatmap showing how the 52nd neuron fires for the header.