Part 1:Japanese Character Recognition

1.1 NetLin

the confusion matrix:

```
[[766.   5.   9.  11.  31.  66.   2.  62.  30.  18.]
 [  6. 674. 105.  18.  28.  23.  58.  12.  25.  51.]
 [  8.  63. 687.  27.  26.  21.  48.  38.  43.  39.]
 [  3.  37.  60. 756.  15.  58.  14.  18.  26.  13.]
 [ 60.  53.  76.  21. 622.  22.  34.  36.  19.  57.]
 [  8.  28. 120.  17.  20. 728.  27.   8.  33.  11.]
 [  5.  23. 145.  10.  25.  24. 726.  20.   8.  14.]
 [ 17.  28.  26.  12.  82.  17.  56. 623.  90.  49.]
 [ 10.  37.  95.  42.   7.  33.  45.   6. 704.  21.]
 [  8.  51.  85.   3.  56.  31.  19.  31.  38. 678.]]
```

Test set: Average loss: 1.0091, Accuracy: 6964/10000 (70%)

1.2 NetFull

the confusion matrix:

```
[[851.   3.   2.   6.  35.  31.   3.  35.  30.   4.]
 [  4. 806.  35.   5.  19.   7.  64.   5.  20.  35.]
 [  6.  12. 841.  36.  14.  15.  26.  10.  21.  19.]
 [  2.   9.  33. 912.   2.  18.   5.   4.   7.   8.]
 [ 40.  26.  15.   8. 826.   5.  30.  17.  19.  14.]
 [  7.  11.  80.  15.  11. 833.  23.   2.  13.   5.]
 [  3.  11.  50.   8.  12.   7. 894.   7.   1.   7.]
 [ 20.  10.  20.   6.  16.  10.  40. 829.  21.  28.]
 [ 12.  27.  39.  53.   3.   7.  34.   4. 813.   8.]
 [  3.  16.  51.   3.  33.   8.  23.  12.  11. 840.]]
```

Test set: Average loss: 0.5085, Accuracy: 8445/10000 (84%)

the total number of independent parameters:

$784 \times 200 + 200 = 157000$

$200 \times 10 + 10 = 2010$

$157000 + 2010 = 159010$

1.3NetConv

the confusion matrix:

```
[[970.   4.   2.   1.  13.   1.   0.   8.   0.   1.]
 [  1. 925.   7.   0.   5.   1.  33.   3.   4.  21.]
 [ 13.   6. 882.  56.   1.   4.  19.   6.   7.   6.]
 [  3.   0.   9. 967.   2.   3.   9.   3.   2.   2.]
 [ 30.  13.   1.   5. 909.   2.  16.   4.   9.  11.]
 [  4.  15.  50.  10.   1. 876.  26.   3.   2.  13.]
```

```
[   3.    3.   15.    2.    3.    0. 971.    2.    0.    1.]
[   6.    6.   14.    1.    2.    3.    8. 931.    4.   25.]
[   6.   17.   12.    6.   15.    1.    8.    1. 915.   19.]
[   6.    8.    9.    1.    6.    0.   10.    5.    3. 952.]]
```

Test set: Average loss: 0.2558, Accuracy: 9298/10000 (93%)

the total number of independent parameters:
(1×3×3)×32 + 32 = 320
(32×3×3)×64 + 64 = 18496
1600 × 128 + 128 = 204928
128 × 10 + 10 = 1290
320 + 18496+204928+1290 =225034

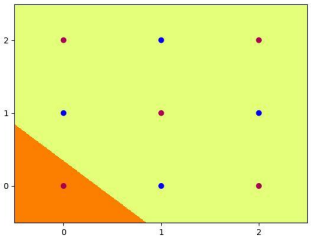
1.4
A.the relative accuracy of the three models:
NetConv > NetFull > NetLin
The accuracy increases with the complexity of the model structure. NetLin is linear and cannot learn spatial lines in images, resulting in the worst performance. NetFull has non-linear hidden layers and high accuracy. NetConv uses convolution to extract local image features and performs the best.

B.the number of independent parameters in each of the three models:
NetConv : 7850
NetFull : 159010
NetLin : 225034
With a large number of independent parameters, the model has better expression ability, requires more training data and resources, and has higher accuracy.

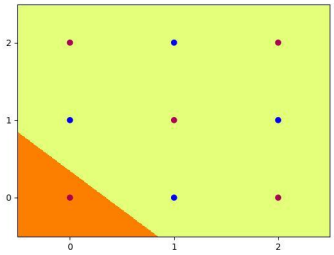
C.the confusion matrix for each model: which characters are most likely to be mistaken for which other characters, and why?:
Due to the lack of spatial structure, NetLin tends to confuse visually similar characters (such as re vs, su tsu).
NetFull has made improvements to this, but there are still issues when dealing with similar shapes (such as ha na).
NetConv solves most of the confusion, but there are still minor errors for inaccurate handwriting (such as wo re, na ha).

Part 2: Multi-Layer Perceptron

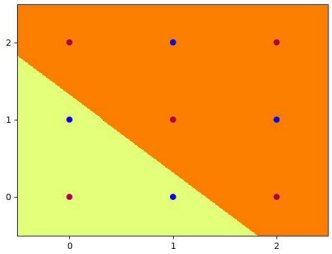
2.1



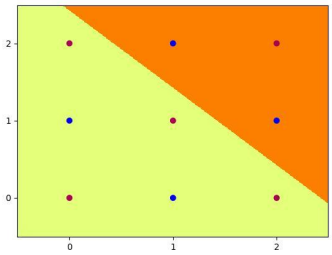
hid_5_0



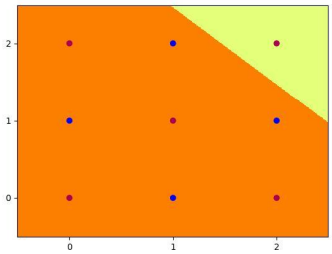
hid_5_1



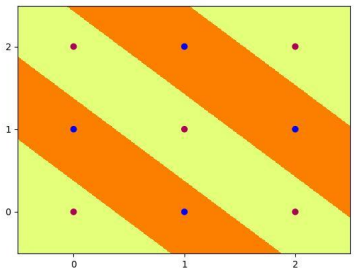
hid_5_2



hid_5_3



hid_5_4



Out_5

2.2

The four hidden units each implement a half-space defined by

$w_1x+w_2y+b=0$

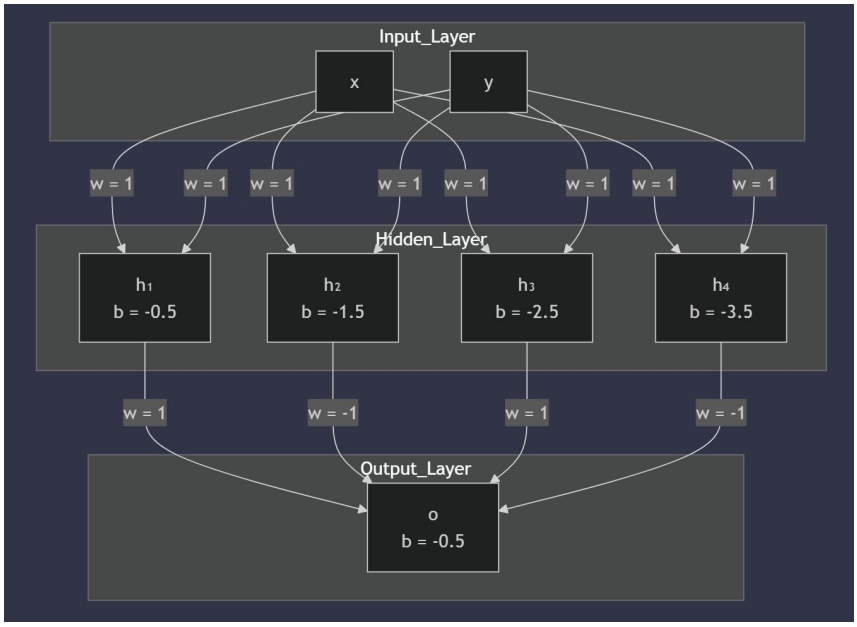
($w_1=w_2=1$ and $b \in \{-0.5,-1.5,-2.5,-3.5\}$)

Hidden unit 1: $x+y-0.5=0$ $y=-x+0.5$

Hidden unit 2: $x+y-1.5=0$ $y=-x+1.5$

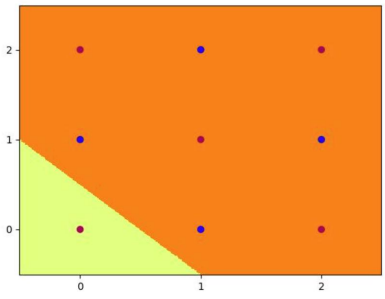
Hidden unit 3: $x+y-2.5=0$ $y=-x+2.5$

Hidden unit 4: $x+y-3.5=0$ $y=-x+3.5$

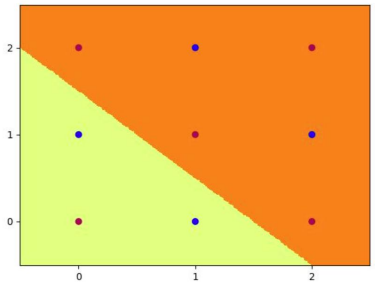


x	y	T	x+y≥0.5	x+y≥1.5	x+y≥2.5	x+y≥3.5	o = H(h1−h2+h3−h4-0.5)
0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	1
0	2	0	1	1	0	0	0
1	0	1	1	0	0	0	1
1	1	0	1	1	0	0	0
1	2	1	1	1	1	0	1
2	0	0	1	1	0	0	1
2	1	1	1	1	1	0	1
2	2	0	1	1	1	1	0

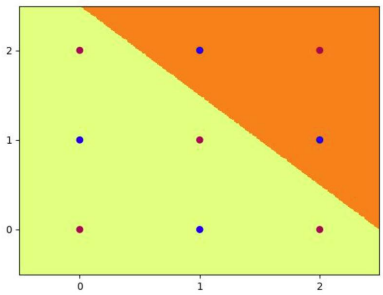
2.3



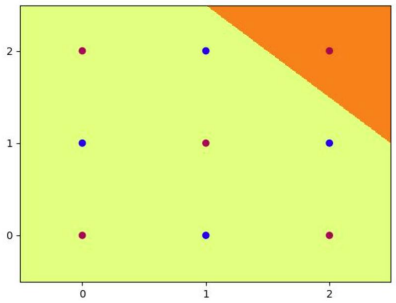
hid_4_0.jpg



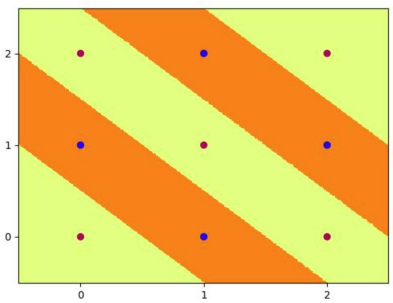
hid_4_1.jpg



hid_4_2.jpg

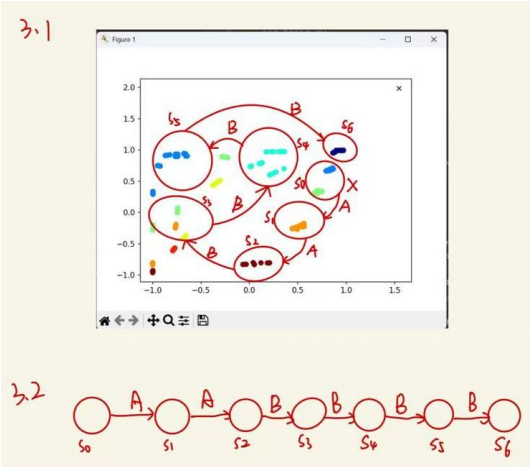


hid_4_3.jpg



out_4.jpg

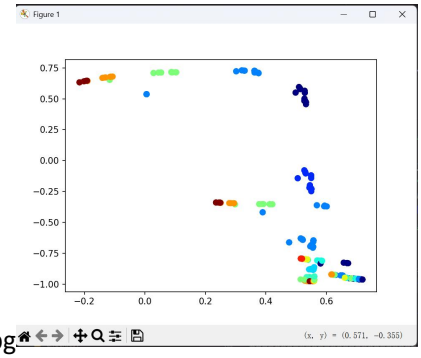
Part 3: Hidden Unit Dynamics for Recurrent Networks



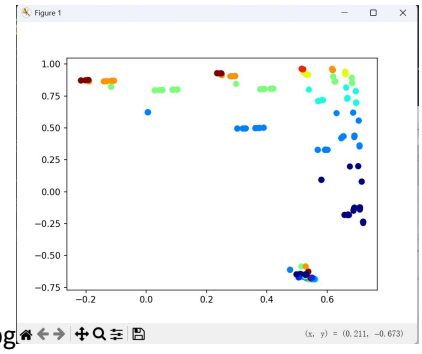
3.3

SRN forms several "stations" in a two-dimensional hidden space, corresponding to the stages of "unread, read A, middle read A, first B, middle read B, and last B", and then jumps the hidden state between these stations based on the input characters: when reading A, it stays in the "output A" area, when seeing the first B jump to "output B", the middle B remains "output B", when seeing the last B jump to a special area to start "output A", and when encountering A again, it returns to the "output A" area. This cycle can predict A, B, and then move on to the next A.

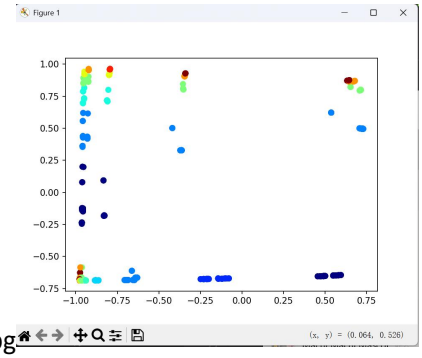
3.4



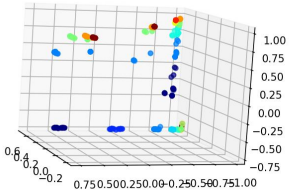
anb2nc3n_lstm3_01.jpg



anb2nc3n_lstm3_02.jpg



anb2nc3n_lstm3_03.jpg



interactive 3D

3.5

LSTM uses its three cell state dimensions, just like the independent "counts" of each fragment: one counts A to n, the second counts B to 2n, and the third counts C to 3n. These gates ensure that each count only increases during its stage, otherwise it remains stable, so the hidden state is divided into three clear clusters. Once the C counter reaches 3n, the gate will move to reset the count and guide the model back to cluster A, so it can start predicting A again.