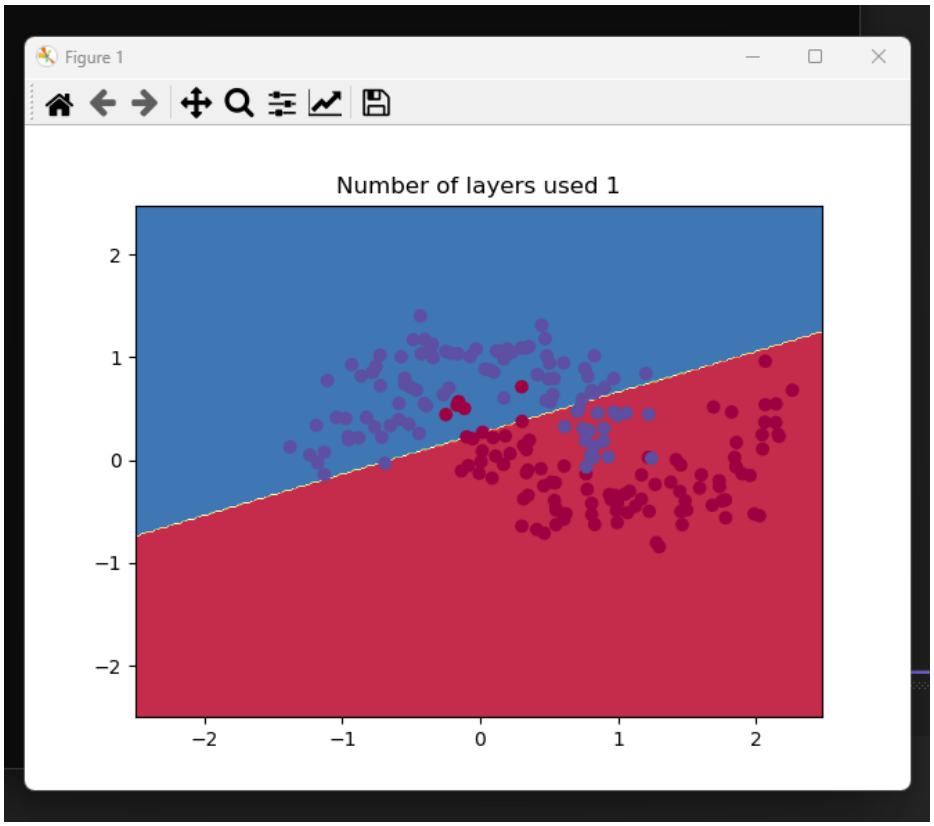
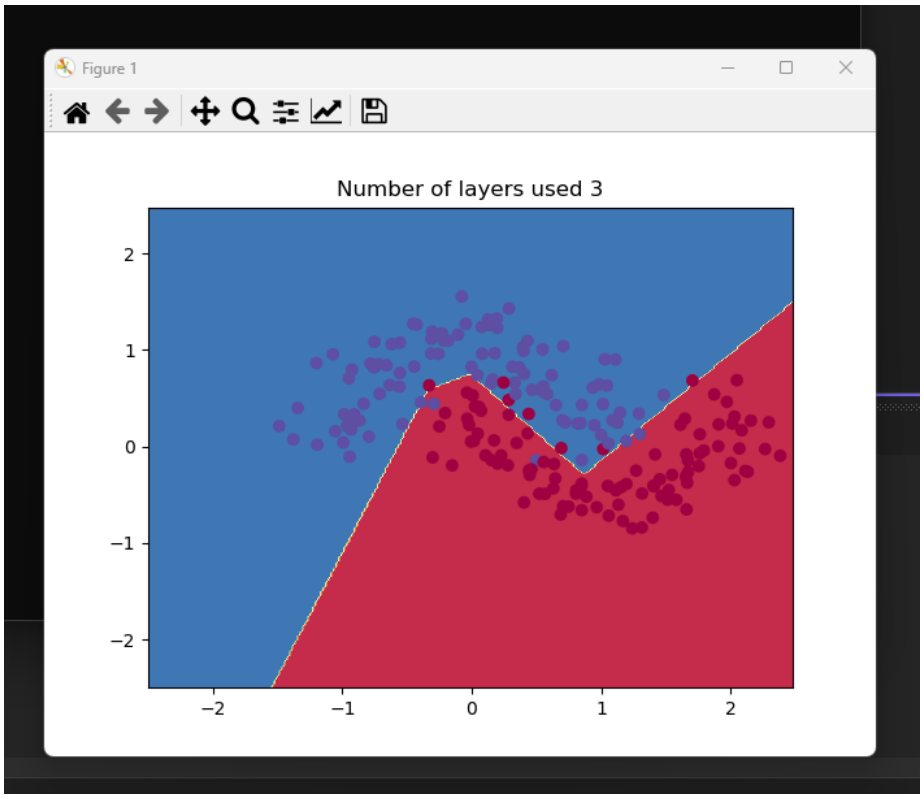
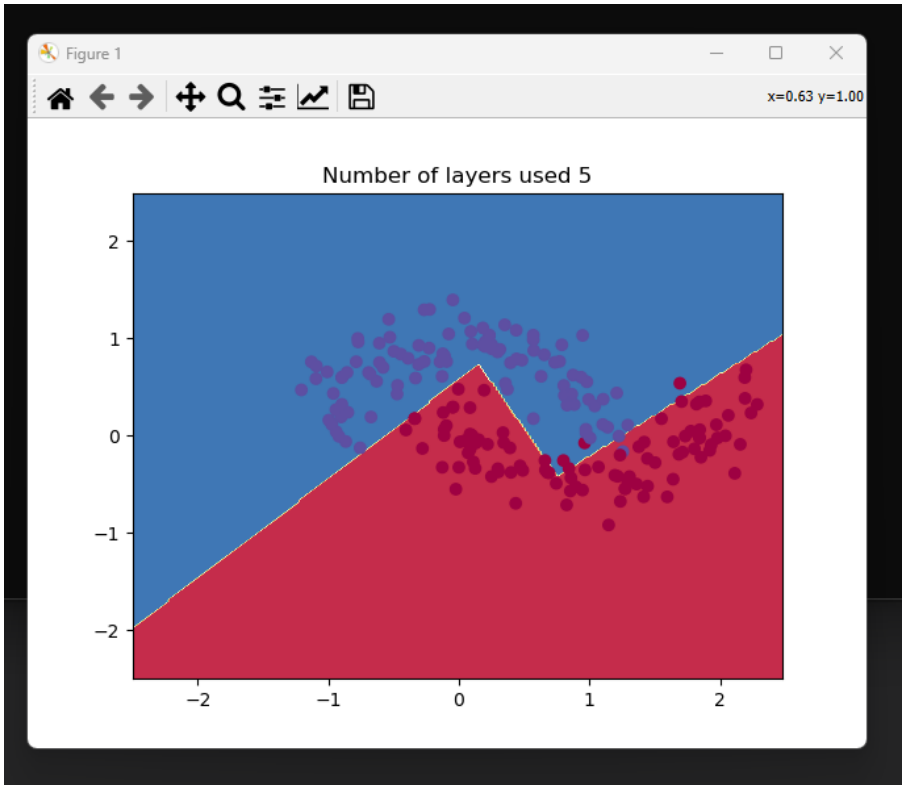
ClassificationNN OUTPUTS



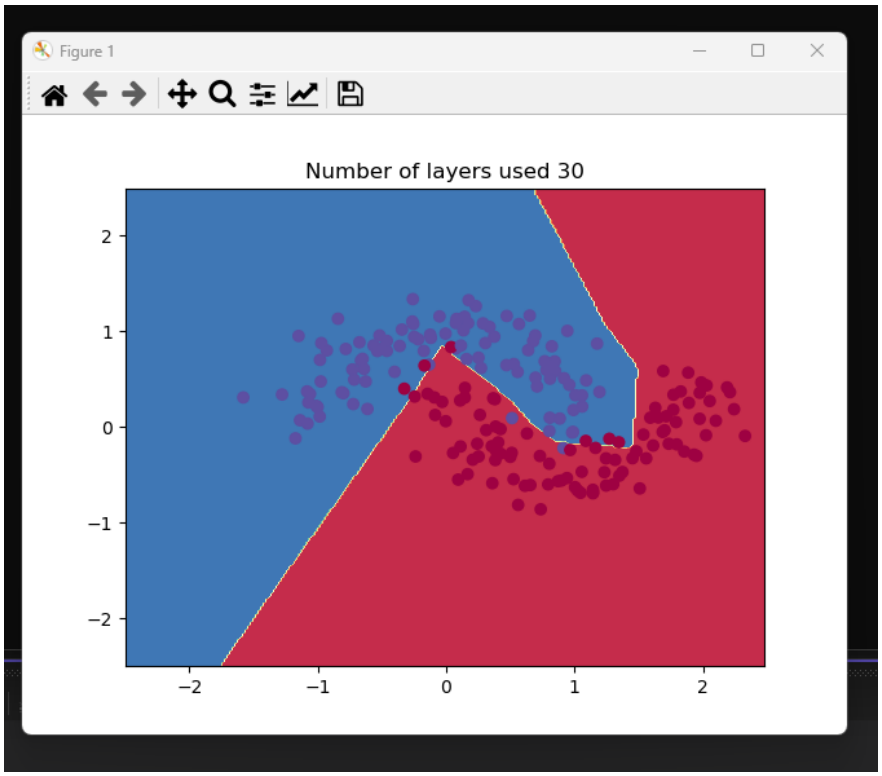

```
epoch: 184  loss= 19.255534220581467
epoch: 185  loss= 19.21428870776981
epoch: 186  loss= 19.781159439751868
epoch: 187  loss= 19.286547020485614
epoch: 188  loss= 19.41970144647621
epoch: 189  loss= 19.649395399495006
epoch: 190  loss= 19.87615908684029
epoch: 191  loss= 19.668115639909956
epoch: 192  loss= 19.56421587250952
epoch: 193  loss= 19.552918314628187
epoch: 194  loss= 19.182114570353406
epoch: 195  loss= 19.28263693550798
epoch: 196  loss= 19.141357535419957
epoch: 197  loss= 19.681785098013734
epoch: 198  loss= 19.325254540708556
epoch: 199  loss= 19.189404901893795
accuracy = 0.875
```




```
epoch: 184  loss= 4.307551568508765
epoch: 185  loss= 4.361649270511722
epoch: 186  loss= 4.404986330433233
epoch: 187  loss= 3.7264403588944432
epoch: 188  loss= 4.488303910500089
epoch: 189  loss= 4.390301330144133
epoch: 190  loss= 4.346254211926162
epoch: 191  loss= 4.359309571369711
epoch: 192  loss= 4.413778246179722
epoch: 193  loss= 4.014934050204298
epoch: 194  loss= 5.897567746422196
epoch: 195  loss= 4.310169752822851
epoch: 196  loss= 4.4252312510076175
epoch: 197  loss= 4.539648507510271
epoch: 198  loss= 4.454847648054311
epoch: 199  loss= 4.346177772443093
accuracy = 0.94
```

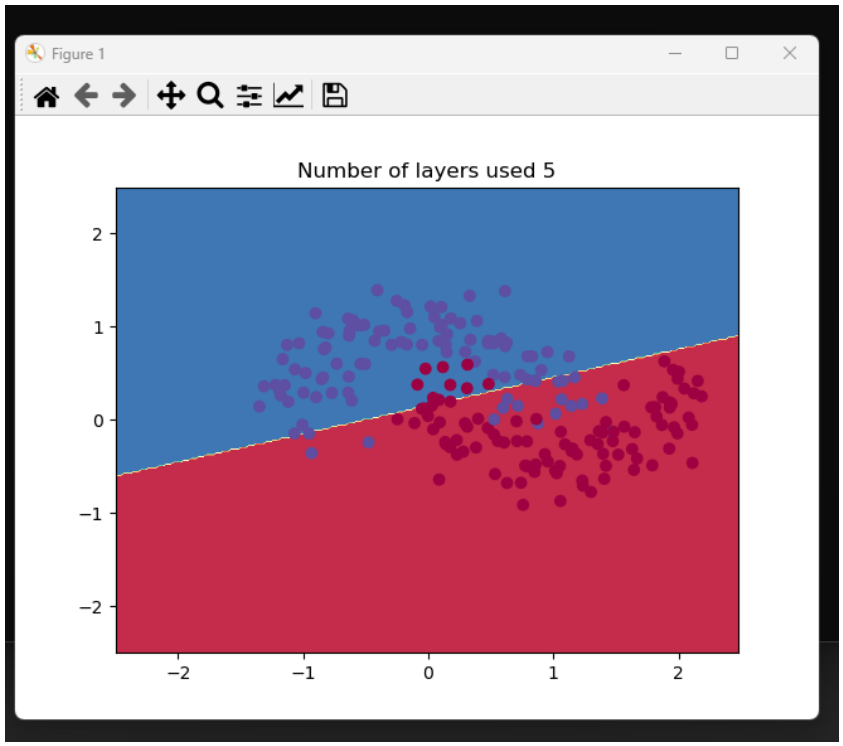



```
epoch: 185  loss= 6.370825967661267
epoch: 186  loss= 6.724350926730576
epoch: 187  loss= 6.515289226583268
epoch: 188  loss= 6.924966602270133
epoch: 189  loss= 6.922898005895218
epoch: 190  loss= 6.553491759777253
epoch: 191  loss= 7.038937921684248
epoch: 192  loss= 11.093066588888544
epoch: 193  loss= 7.453852656454654
epoch: 194  loss= 6.698616760018846
epoch: 195  loss= 7.975029661934762
epoch: 196  loss= 6.886094171093354
epoch: 197  loss= 6.496528733541655
epoch: 198  loss= 7.061511181452862
epoch: 199  loss= 6.535664374055769
accuracy = 0.98
```

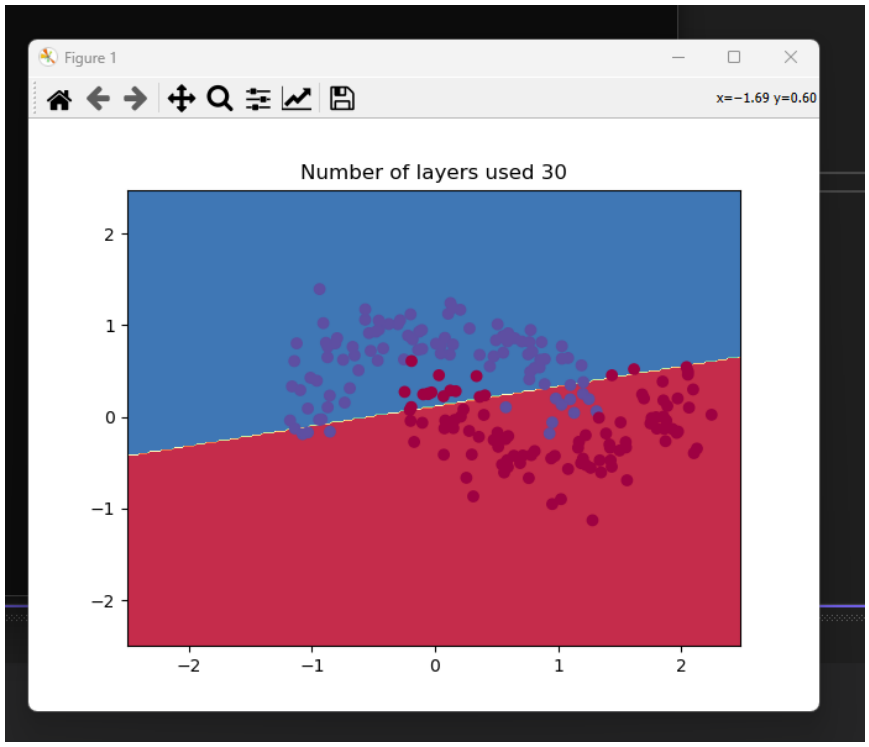



```
epoch: 486  loss= 4.189487827578941
epoch: 487  loss= 4.035142102191243
epoch: 488  loss= 4.264063790768653
epoch: 489  loss= 5.046808731723869
epoch: 490  loss= 4.038741072041621
epoch: 491  loss= 4.412918576212583
epoch: 492  loss= 4.276154428704283
epoch: 493  loss= 3.805759033950625
epoch: 494  loss= 4.618288385652111
epoch: 495  loss= 4.080526169103049
epoch: 496  loss= 4.1075894975761
epoch: 497  loss= 3.9370484086613122
epoch: 498  loss= 5.35709366589998
epoch: 499  loss= 4.789926358827164
accuracy = 0.96
```

Exercise – Disabling activation function, to observe the effect on the decision boundary




```
epoch: 188  loss= 20.37542255590961
epoch: 189  loss= 20.26987913013727
epoch: 190  loss= 20.58550043883224
epoch: 191  loss= 20.201802069605037
epoch: 192  loss= 20.740958086113096
epoch: 193  loss= 20.45410934717802
epoch: 194  loss= 20.428913734303933
epoch: 195  loss= 20.479848711237537
epoch: 196  loss= 20.557069529331784
epoch: 197  loss= 20.41259622323014
epoch: 198  loss= 20.317004611752054
epoch: 199  loss= 20.306503502750274
accuracy = 0.865
```

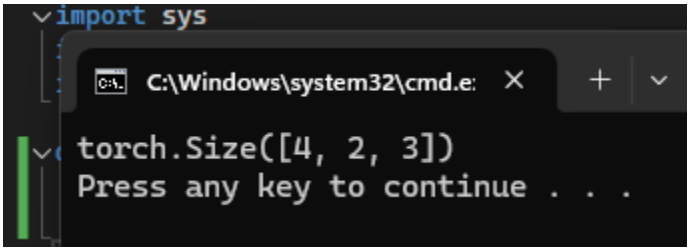



```
epoch: 190  loss= 18.523408752648663
epoch: 191  loss= 18.31954278473836
epoch: 192  loss= 18.415257867049604
epoch: 193  loss= 18.66230396336414
epoch: 194  loss= 18.334749117880392
epoch: 195  loss= 18.62585970772699
epoch: 196  loss= 18.661230879542927
epoch: 197  loss= 18.577451084686345
epoch: 198  loss= 18.40872977217623
epoch: 199  loss= 18.443722719026937
accuracy = 0.84
```

Conclusion

Decision boundary appears to be linear, due to missing Activation function which helps to introduce nonlinear behavior into the linear networks

PytorchFundamentals outputs



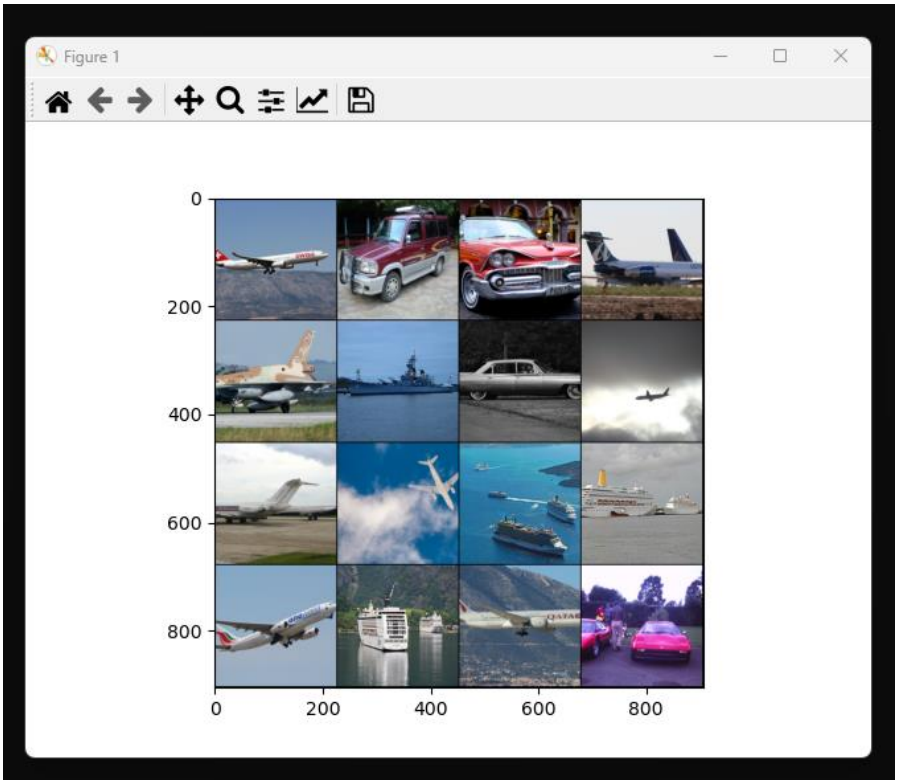
C:\Windows\system32\cmd.e

```
t1.shape: torch.Size([4, 2, 3])

t1.shape[0]: 4

t1.shape[1]: 2

t1.shape[2]: 3

t1.shape[-1]: 3

Adding extra dim to t1: torch.Size([1, 4, 2, 3])

Adding extra dim to t1 using view: torch.Size([1, 4, 2, 3])

Adding extra dim at the end: torch.Size([4, 2, 3, 1])

Adding extra dim using unsqueeze: torch.Size([1, 4, 2, 3])

Removing extra dim using squeeze: torch.Size([4, 2, 3])

Reshaping t1 to (4,6): torch.Size([4, 6])

Reshaping t1 to (4,-1): torch.Size([4, 6])

Unpacked list: [(2, 3, 5), (6, 7, 8)]

bb: (2, 6)

After None shape: torch.Size([1, 2, 2])

r: tensor([[2, 1],
       [3, 4]])

w: tensor([1, 2, 3, 4])
```

C:\Windows\system32\cmd.e

```
Mean along dim 1: tensor([[0.3333],
       [0.3333],
       [0.3333],
       [0.3333]])


Mask with True for 0 and False for non-zero: tensor([[ True,  True,  True],
       [ True,  True,  True],
       [ True,  True, False]])


Mask with 1 for True and 0 for False: tensor([[1, 1, 1],
       [1, 1, 1],
       [1, 1, 0]])


Mask with -1000 for non-zero: tensor([[    0,     0,     0],
       [    0,     0,     0],
       [    0,     0, -1000]])


Mask with float values (1.0 for > 0): tensor([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 1.]])


np.eye(3): [[1. 0. 0.]
[0. 1. 0.]
[0. 0. 1.]]


One-hot encoded board: [[1. 0. 0.]
[1. 0. 0.]
[0. 1. 0.]
[1. 0. 0.]
[0. 0. 1.]
[1. 0. 0.]
[0. 1. 0.]
[0. 0. 1.]
[1. 0. 0.]]

---------

Switching 2nd and 3rd column in one-hot encoded board: [[1. 0. 0.]
[1. 0. 0.]
[0. 0. 1.]
[1. 0. 0.]
[0. 1. 0.]
[1. 0. 0.]
[0. 0. 1.]
[0. 1. 0.]
```

SimpleClassification output



NetworkLinear() output

```
C:\Windows\system32\cmd.e

Image shape: torch.Size([3, 224, 224])
epoch: 0 iter: 100 loss: 0.8809245890378952

epoch: 1 iter: 100 loss: 1.4533831441402436

epoch: 2 iter: 100 loss: 1.3989401495456695

epoch: 3 iter: 100 loss: 1.3402063542604445

epoch: 4 iter: 100 loss: 1.3121134376525878

epoch: 5 iter: 100 loss: 1.2647291654348374

epoch: 6 iter: 100 loss: 1.2353571385145188

epoch: 7 iter: 100 loss: 1.227666712999344

epoch: 8 iter: 100 loss: 1.1950432556867598

epoch: 9 iter: 100 loss: 1.19761536359787

epoch: 10 iter: 100 loss: 1.1799530547857284

epoch: 11 iter: 100 loss: 1.1666854214668274

epoch: 12 iter: 100 loss: 1.1520605617761612

epoch: 13 iter: 100 loss: 1.153968396782875

epoch: 14 iter: 100 loss: 1.1419833010435105

epoch: 15 iter: 100 loss: 1.140053750872612

epoch: 16 iter: 100 loss: 1.1411062741279603

epoch: 17 iter: 100 loss: 1.131710443496704

epoch: 18 iter: 100 loss: 1.1268090116977691

epoch: 19 iter: 100 loss: 1.1378674423694461

epoch: 20 iter: 100 loss: 1.1181125473976135

epoch: 21 iter: 100 loss: 1.123964866399765

epoch: 22 iter: 100 loss: 1.123115332722664

epoch: 23 iter: 100 loss: 1.1219847577810287

epoch: 24 iter: 100 loss: 1.1202456748485565


 Accuracy: 0.7642857142857142

Press any key to continue . . .
```

NetworkCNN() output

C:\Windows\system32\cmd.e

```
Image shape: torch.Size([3, 224, 224])
epoch: 0 iter: 100 loss: 1.0543022668361663

epoch: 1 iter: 100 loss: 1.6313523352146149

epoch: 2 iter: 100 loss: 1.5491818803548814

epoch: 3 iter: 100 loss: 1.5159008204936981

epoch: 4 iter: 100 loss: 1.4561552697420 12

epoch: 5 iter: 100 loss: 1.4530759274959564

epoch: 6 iter: 100 loss: 1.4151126551628113

epoch: 7 iter: 100 loss: 1.3831702095270157

epoch: 8 iter: 100 loss: 1.3785346728563308

epoch: 9 iter: 100 loss: 1.334064707159996

epoch: 10 iter: 100 loss: 1.3042621791362763

epoch: 11 iter: 100 loss: 1.2735064309835433

epoch: 12 iter: 100 loss: 1.2938377910852432

epoch: 13 iter: 100 loss: 1.2424659031629561

epoch: 14 iter: 100 loss: 1.2362582594156266

epoch: 15 iter: 100 loss: 1.2328362649679183

epoch: 16 iter: 100 loss: 1.2135865885019301

epoch: 17 iter: 100 loss: 1.2213325881958008

epoch: 18 iter: 100 loss: 1.2032157891988755

epoch: 19 iter: 100 loss: 1.1975676053762436

epoch: 20 iter: 100 loss: 1.1806980699300766

epoch: 21 iter: 100 loss: 1.1557463139295578

epoch: 22 iter: 100 loss: 1.166347821354866

epoch: 23 iter: 100 loss: 1.156699515581131

epoch: 24 iter: 100 loss: 1.1485591173171996


 Accuracy: 0.825

Press any key to continue . . .
```