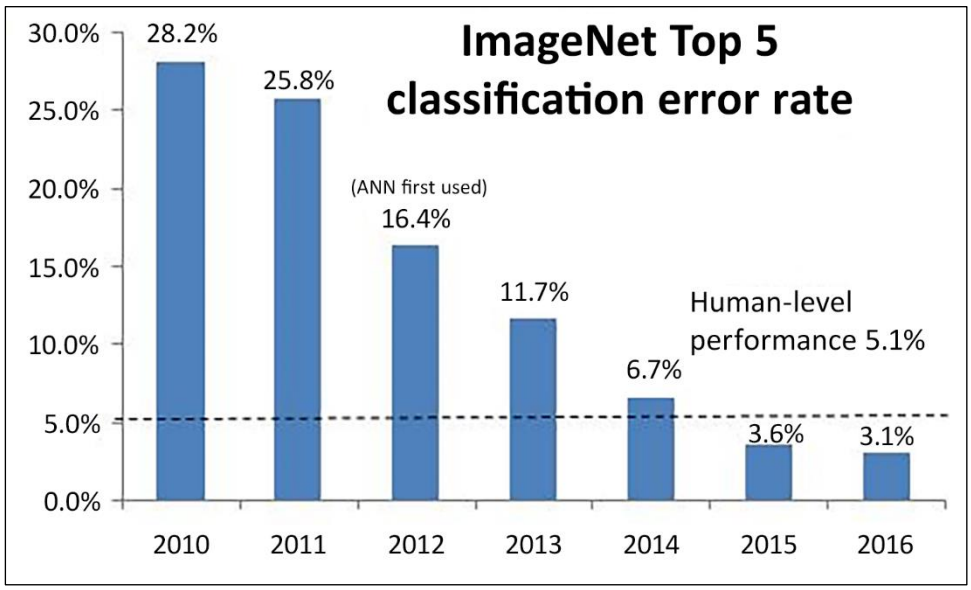
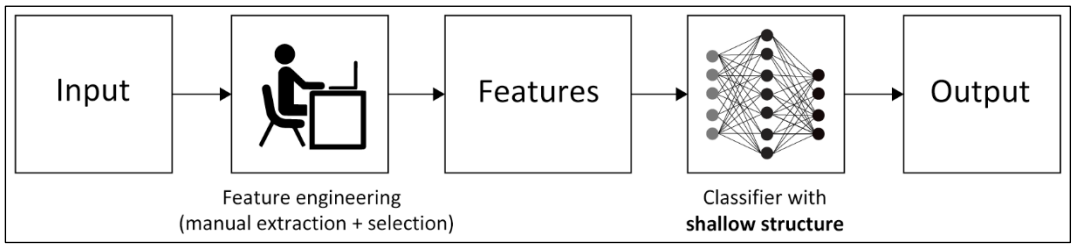
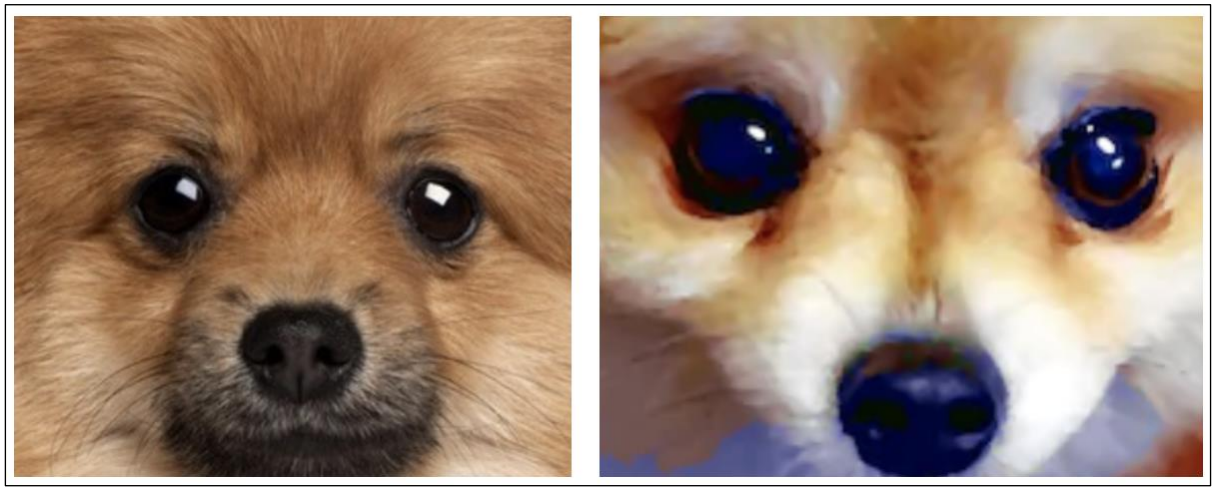
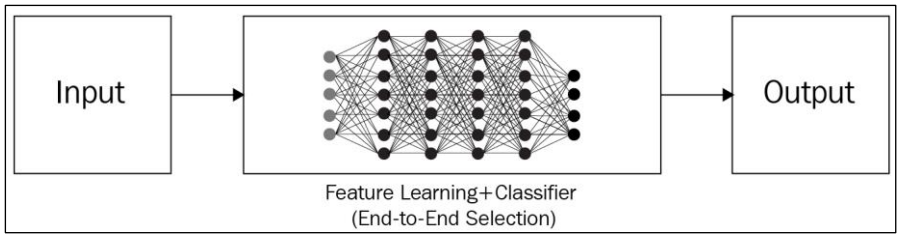
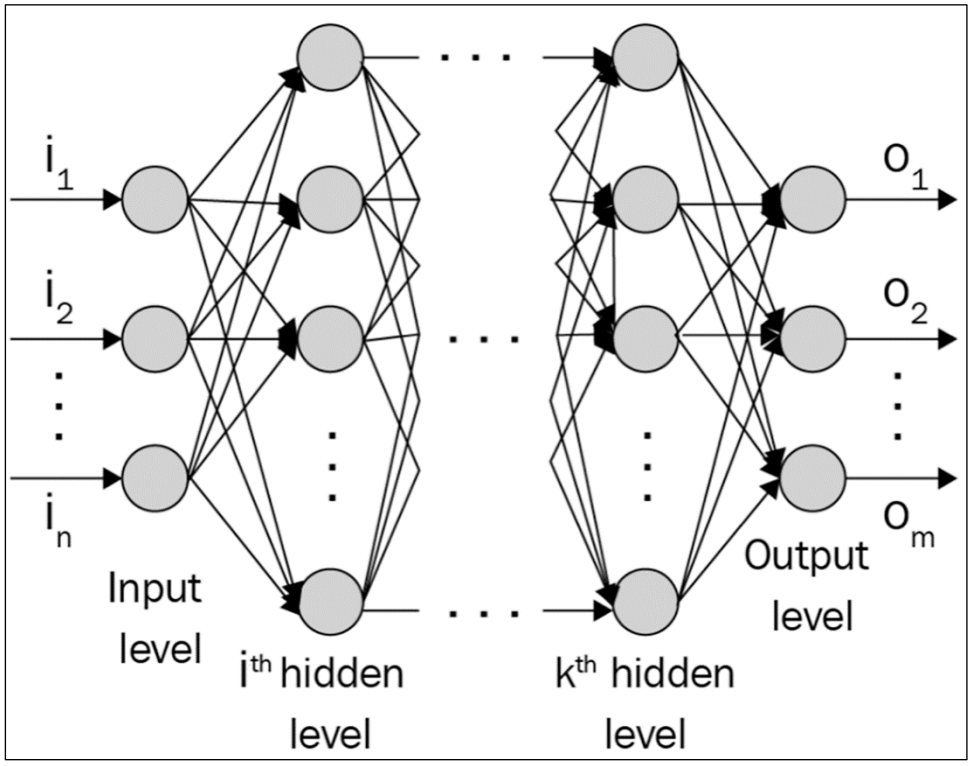
Chapter 1:

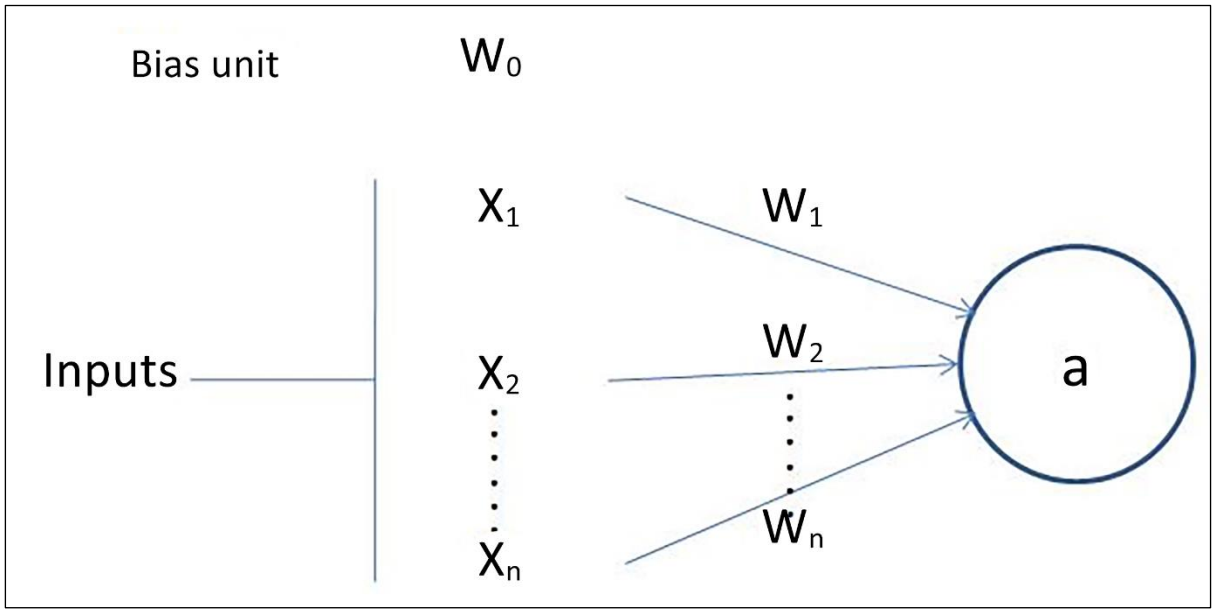

ImageNet Top 5
classification error rate
30.0%
25.0%
20.0%
15.0%
10.0%
5.0%
0.0%
28.2%
25.8%
(ANN first used)
16.4%
11.7%
6.7%
3.6%
3.1%
Human-level
performance 5.1%
2010
2011
2012
2013
2014
2015
2016

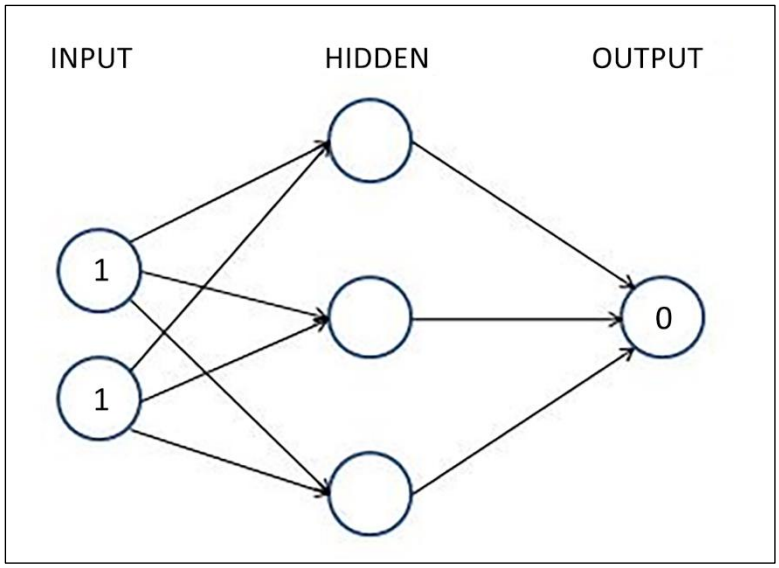

Input
Features
Output
Feature engineering
(manual extraction + selection)
Classifier with
shallow structure

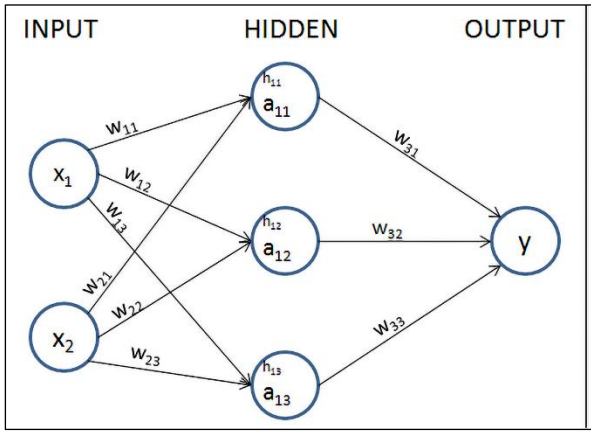


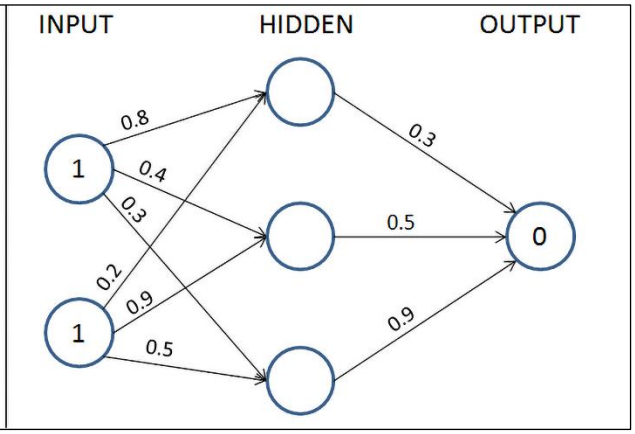

Input
Output
Feature Learning+Classifier
(End-to-End Selection)

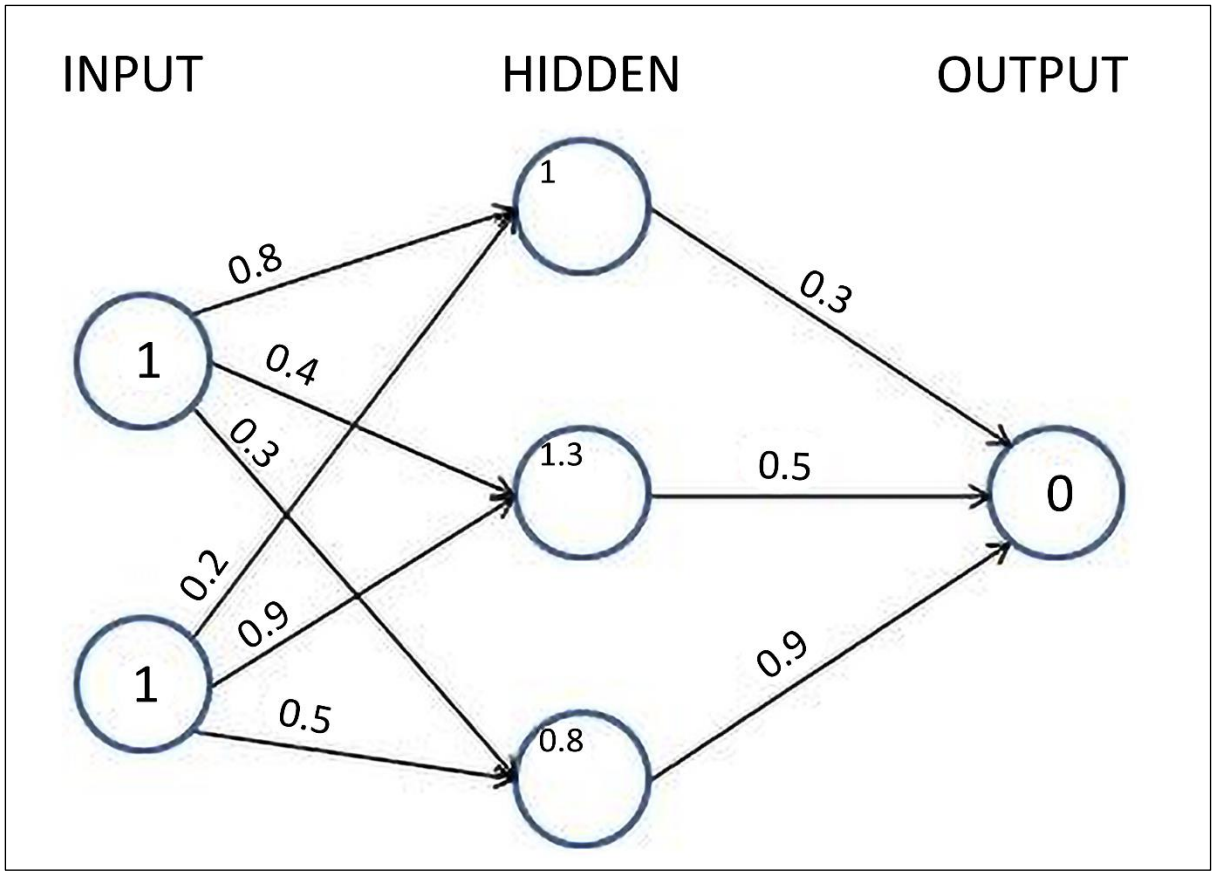

i1
i2
in
o1
o2
om
Input level
ith hidden level
kth hidden level
Output level

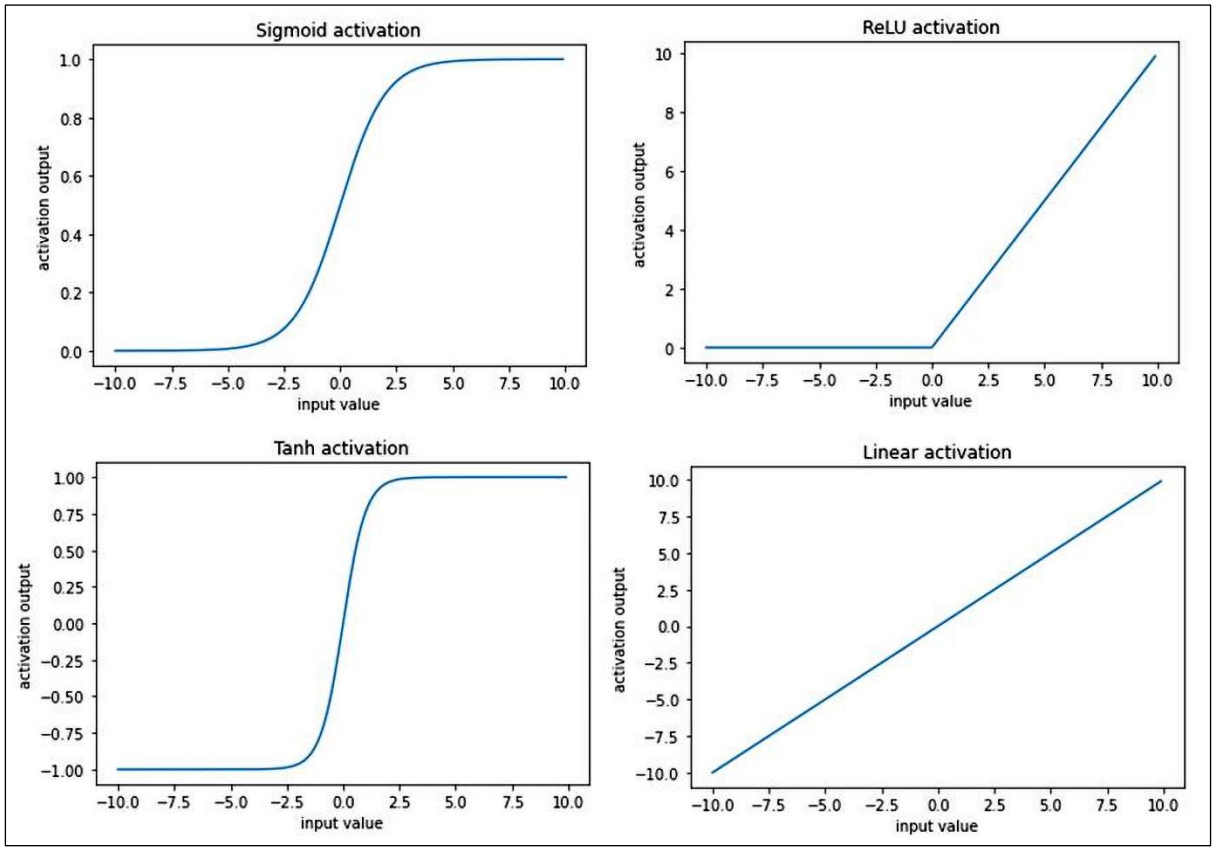

Bias unit
W0
Inputs
X1
X2
Xn
W1
W2
Wn
a

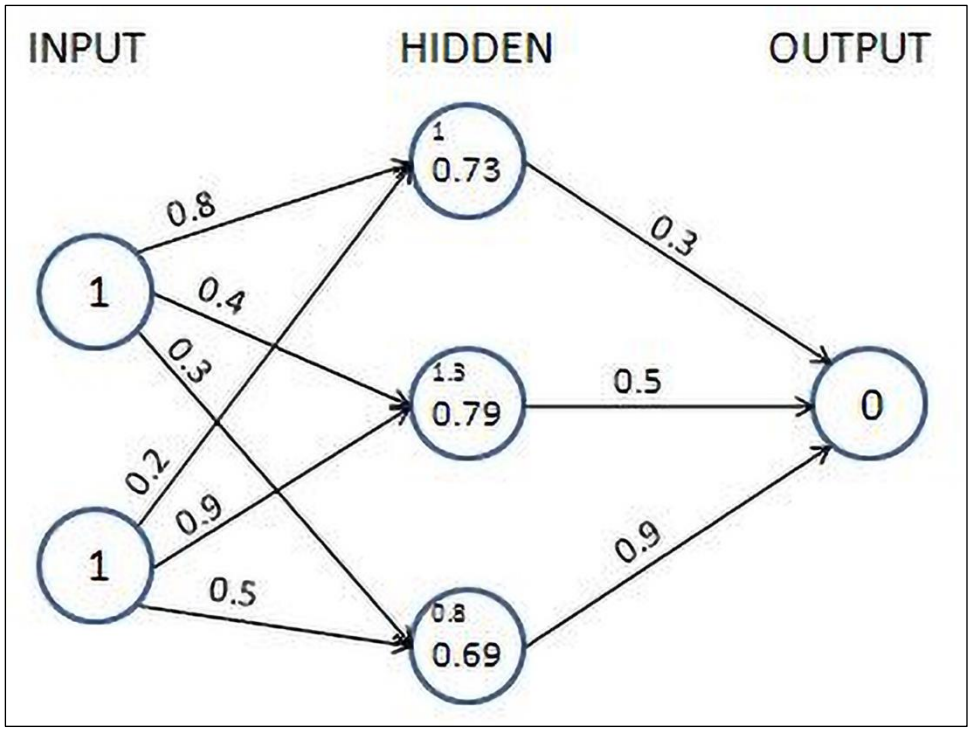

INPUT
HIDDEN
OUTPUT
1
1
0


INPUT
HIDDEN
OUTPUT
x1
x2
w11
w12
w13
w21
w22
w23
h11
a11
h12
a12
h13
a13
w31
w32
w33
y


INPUT
HIDDEN
OUTPUT
1
1
0.8
0.4
0.3
0.2
0.9
0.5
0.3
0.5
0.9
0


INPUT
HIDDEN
OUTPUT
1
1
1
1.3
0.8
0
0.8
0.4
0.3
0.3
0.5
0.2
0.9
0.9
0.5


Sigmoid activation
ReLU activation
Tanh activation
Linear activation
activation output
input value



Target (correct class)	Prediction probabilities					Cross-entropy loss	
	Dog	Cat	Cow	Hen	Rat		
Dog	0.88	0.02	0.04	0.04	0.02	-log(0.88)	= 0.128
Cat	0.26	0.21	0.17	0.18	0.18	-log(0.21)	= 1.56
Cow	0.01	0.01	0.96	0.01	0.01	-log(0.96)	= 0.04
Hen	0.14	0.09	0.01	0.57	0.19	-log(0.57)	= 0.56
Rat	0.21	0.02	0.05	0.17	0.55	-log(0.55)	= 0.597

Modern-Computer-Vision-with-PyTorch-2E / Chapter01 / **Feed_forward_propagation.ipynb**

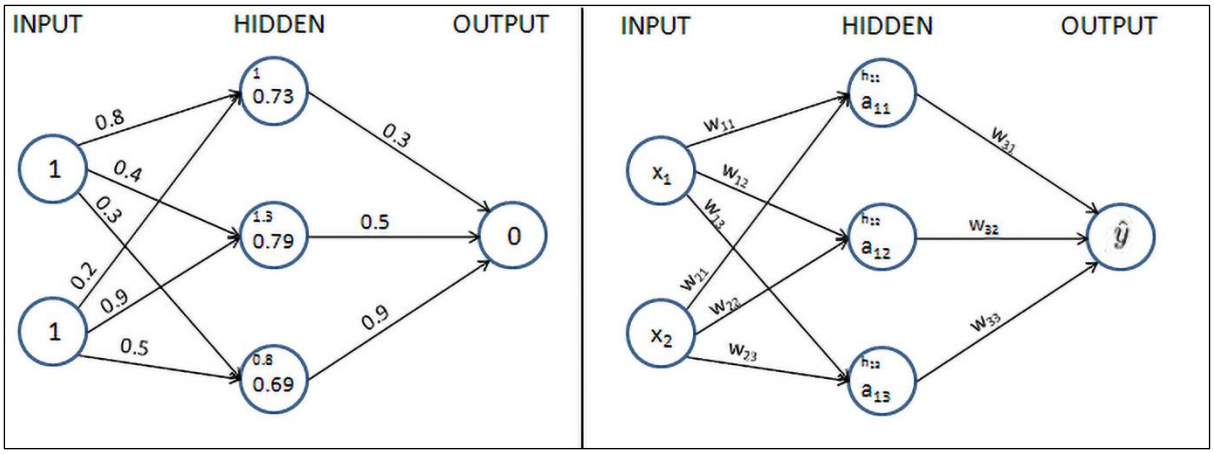
Preview | Code | Blame | 74 lines (74 loc) · 2.02 KB

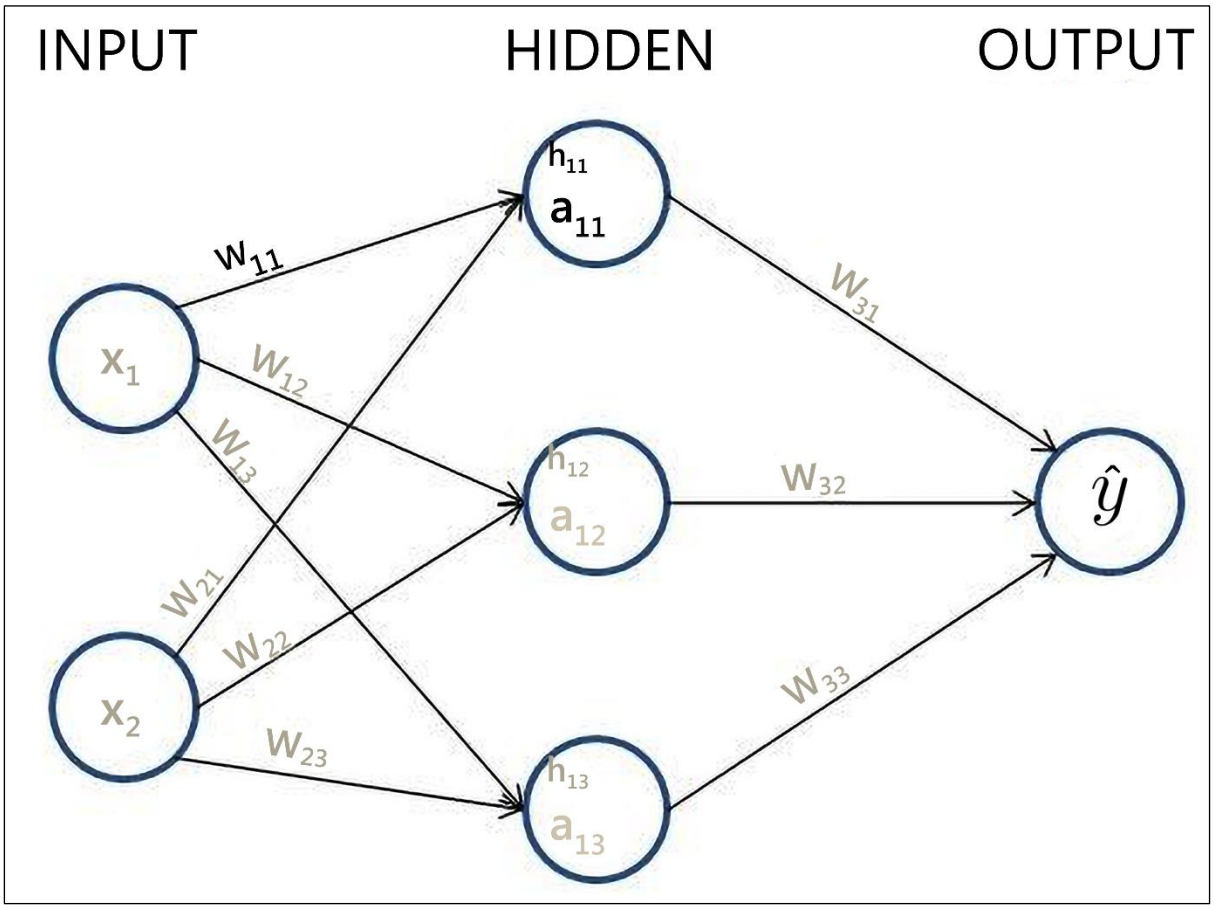
Open in Colab

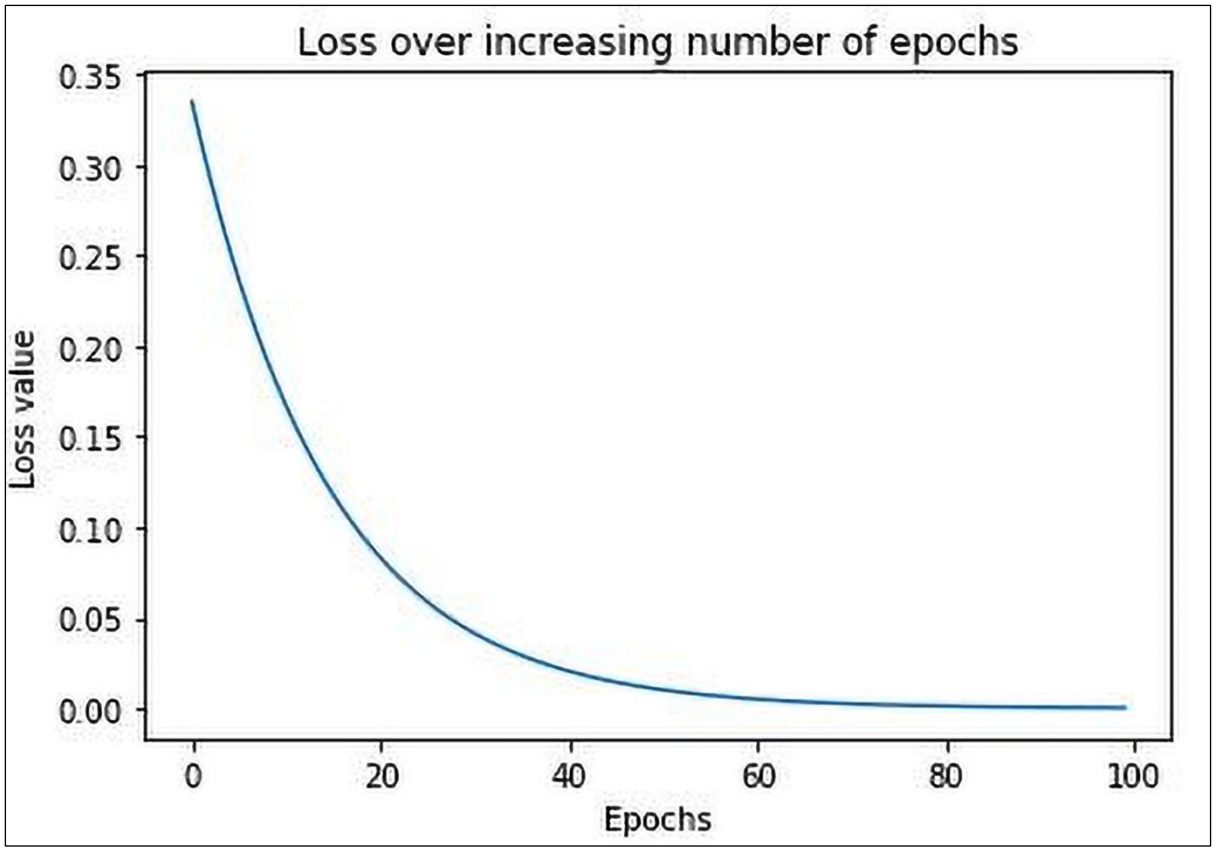
Forward Propagation

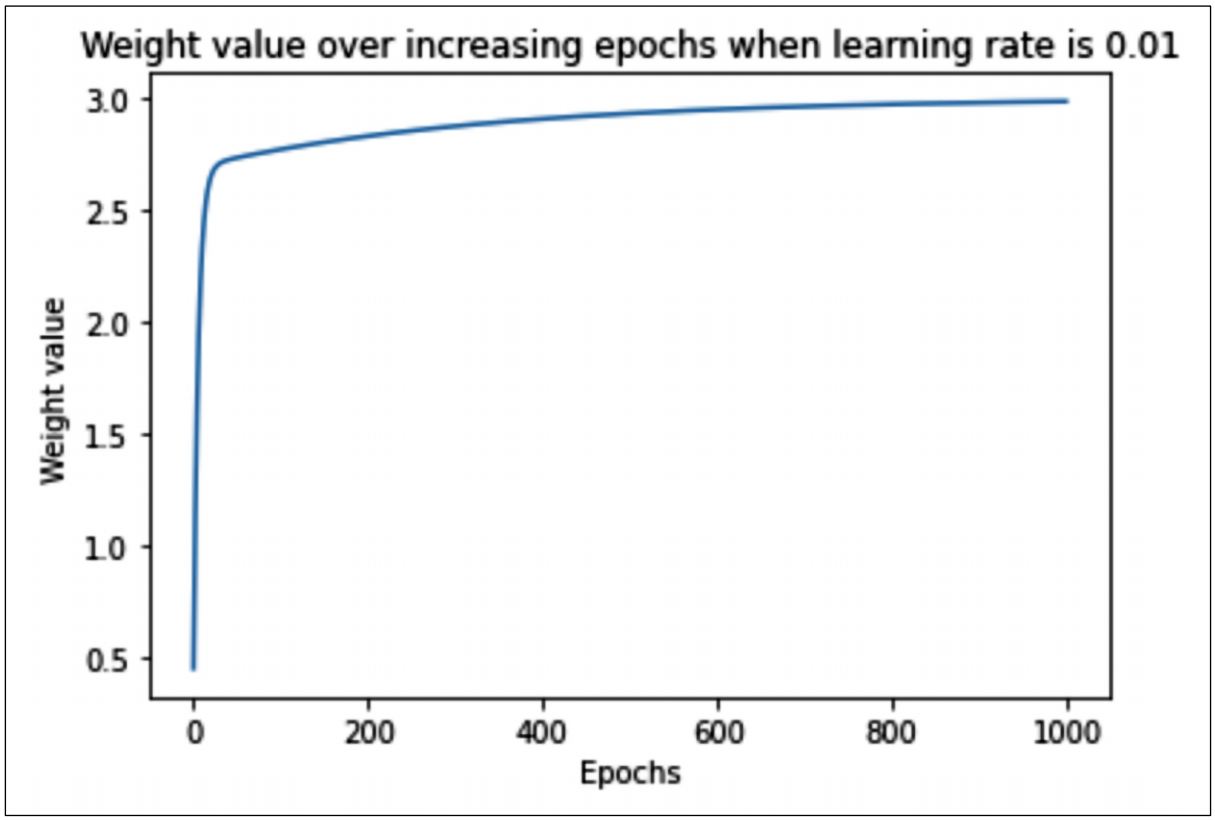
In [ ]:

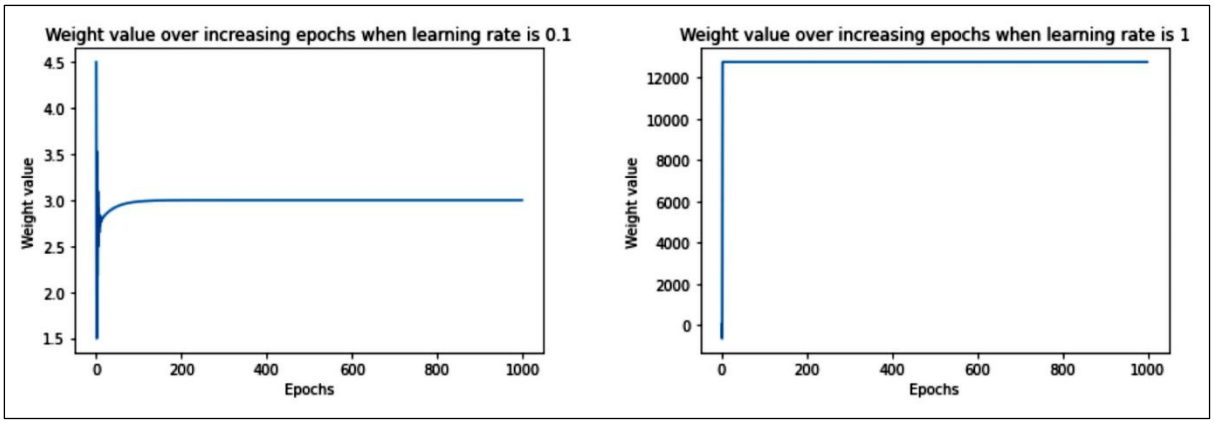
```
import numpy as np
def feed_forward(inputs, outputs, weights):
    pre_hidden = np.dot(inputs,weights[0])+ weights[1]
    hidden = 1/(1+np.exp(-pre_hidden))
    pred_out = np.dot(hidden, weights[2]) + weights[3]
    mean_squared_error = np.mean(np.square(pred_out - outputs))
    return mean_squared_error
```

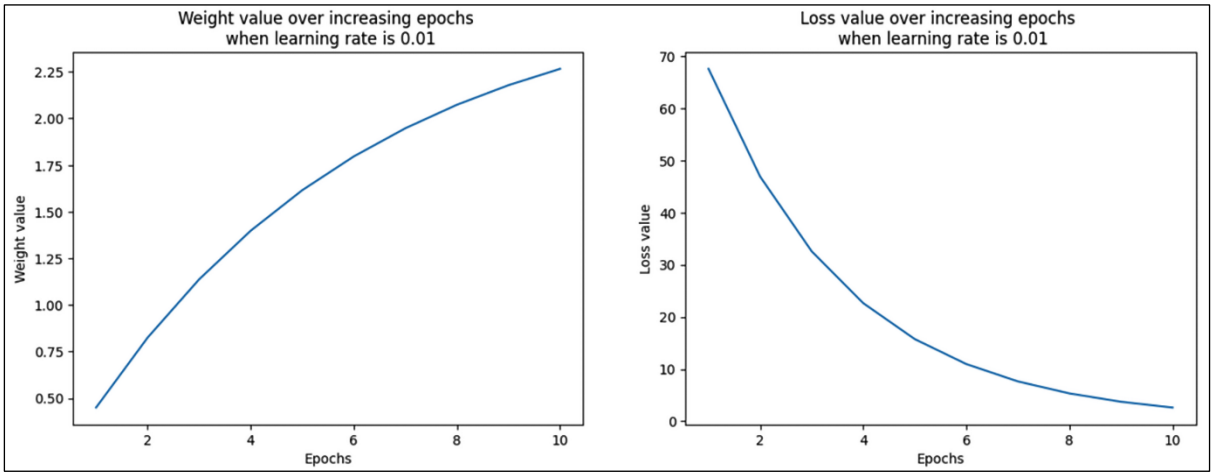

INPUT
HIDDEN
OUTPUT
1
1
1
0.73
1.3
0.79
0.8
0.69
0
0.8
0.4
0.3
0.2
0.9
0.5
0.3
0.5
0.9
INPUT
HIDDEN
OUTPUT
x_1
x_2
h_{11} a_{11}
h_{12} a_{12}
h_{13} a_{13}
w_{11}
w_{12}
w_{13}
w_{21}
w_{22}
w_{23}
w_{31}
w_{32}
w_{33}
$\hat{y}$

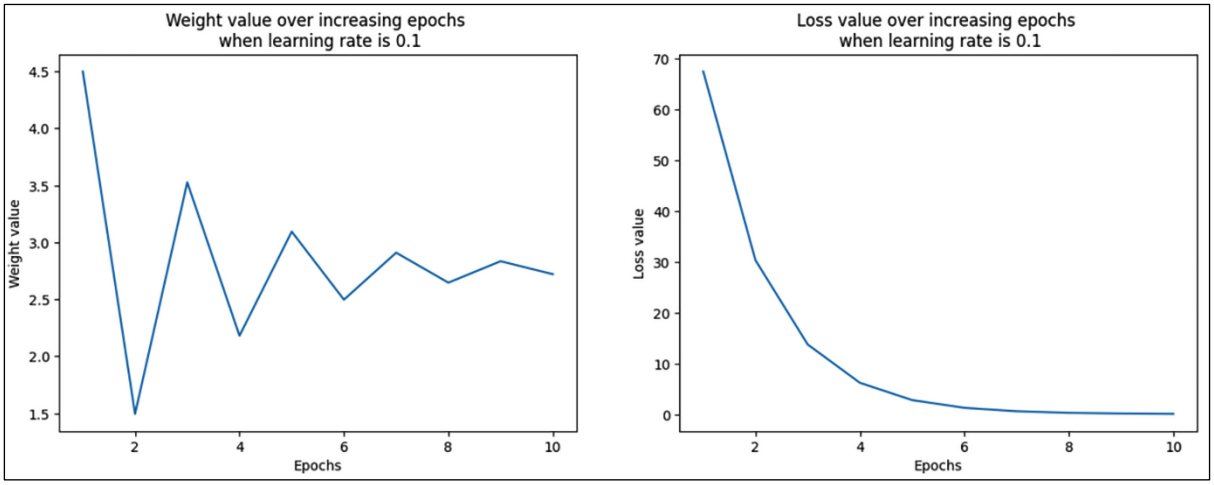

INPUT
HIDDEN
OUTPUT
x_1
x_2
h_{11} a_{11}
h_{12} a_{12}
h_{13} a_{13}
w_{11}
w_{12}
w_{13}
w_{21}
w_{22}
w_{23}
w_{31}
w_{32}
w_{33}
$\hat{y}$

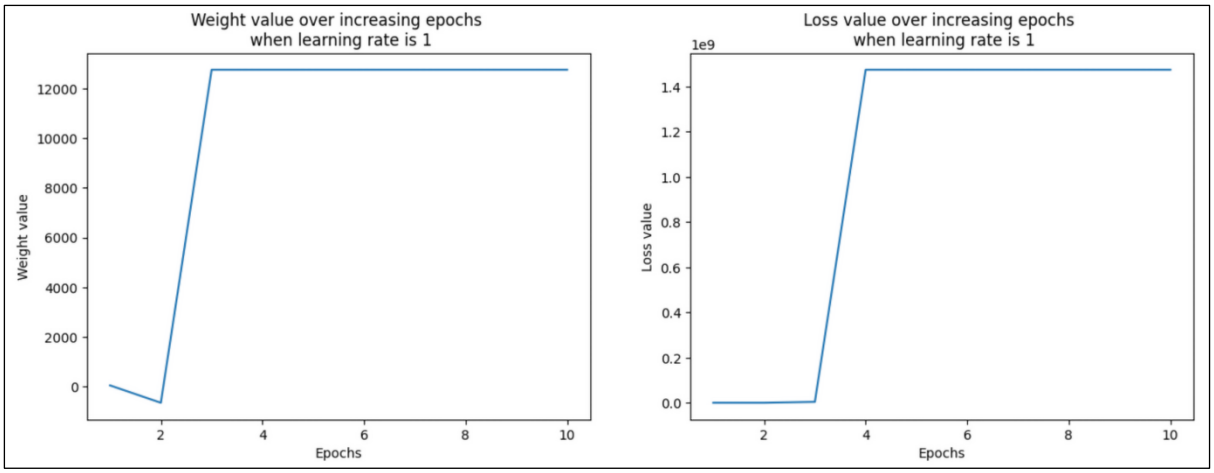

Loss over increasing number of epochs
Loss value
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
20
40
60
80
100
Epochs


Weight value over increasing epochs when learning rate is 0.01
Weight value
3.0
2.5
2.0
1.5
1.0
0.5
0
200
400
600
800
1000
Epochs


Weight value over increasing epochs when learning rate is 0.1
Weight value
Epochs
Weight value over increasing epochs when learning rate is 1
Weight value
Epochs

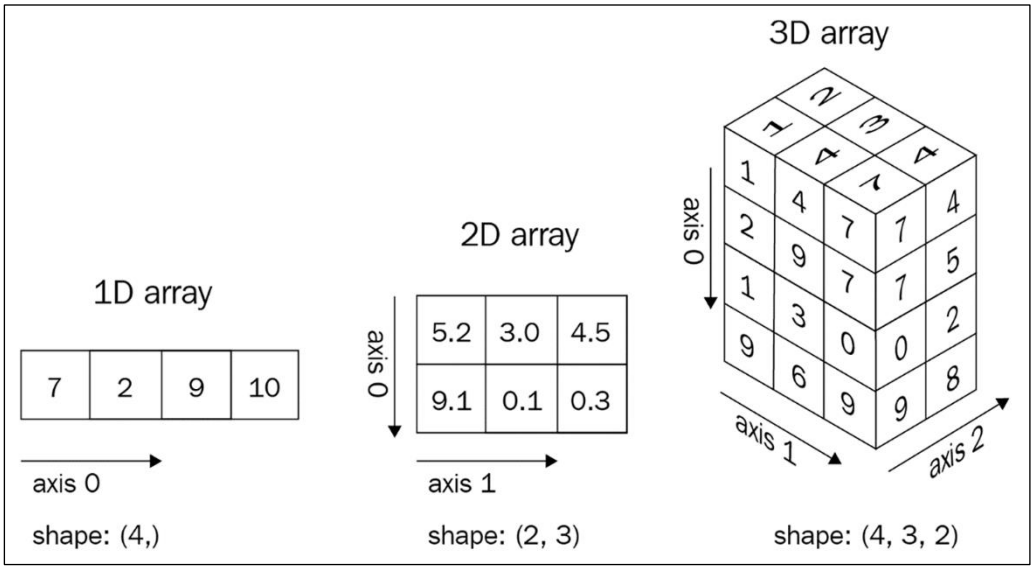

Weight value over increasing epochs
when learning rate is 0.01
Weight value
Epochs
Loss value over increasing epochs
when learning rate is 0.01
Loss value
Epochs


Weight value over increasing epochs
when learning rate is 0.1
Weight value
Epochs
Loss value over increasing epochs
when learning rate is 0.1
Loss value
Epochs


Weight value over increasing epochs
when learning rate is 1
Weight value
Epochs
12000
10000
8000
6000
4000
2000
0
2
4
6
8
10
Loss value over increasing epochs
when learning rate is 1
1e9
Loss value
Epochs
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0

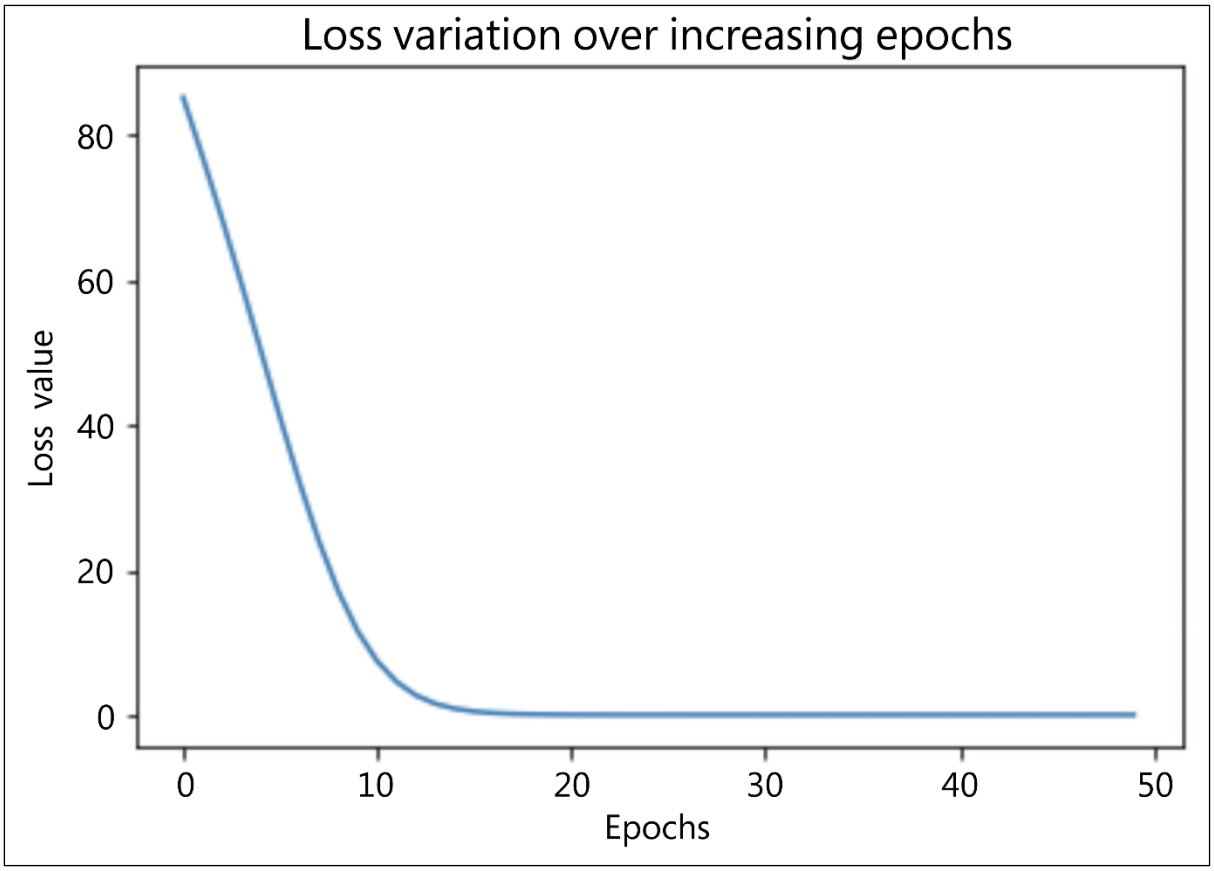
Chapter 2:

PyTorch Build	Stable (2.3.0)		Preview (Nightly)	
Your OS	Linux	Mac	Windows	
Package	Conda	Pip	LibTorch	Source
Language	Python		C++ / Java	
Compute Platform	CUDA 11.8	CUDA 12.1	ROCm 6.0	CPU
Run this Command:	`conda install pytorch torchvision torchaudio pytorch-cuda=12.1 -c pytorch -c nvidia`			




```
Parameter containing:
tensor([[ 0.5670,  0.2775],
        [-0.5525, -0.0506],
        [-0.1226, -0.0549],
        [-0.3667,  0.5775],
        [-0.2847, -0.7009],
        [-0.0449,  0.3303],
        [ 0.2479, -0.1501],
        [-0.4169, -0.0649]], requires_grad=True)
```

```
Parameter containing:
tensor([[ 0.5670,  0.2775],
        [-0.5525, -0.0506],
        [-0.1226, -0.0549],
        [-0.3667,  0.5775],
        [-0.2847, -0.7009],
        [-0.0449,  0.3303],
        [ 0.2479, -0.1501],
        [-0.4169, -0.0649]], requires_grad=True)
Parameter containing:
tensor([-0.7037,  0.4445, -0.4399,  0.6718,  0.2934, -0.6325,  0.2646, -0.5508],
       requires_grad=True)
Parameter containing:
tensor([[ 0.1219, -0.2936,  0.0820,  0.1212, -0.0885, -0.0113,  0.2657,  0.2921]],
       requires_grad=True)
Parameter containing:
tensor([0.0119], requires_grad=True)
```

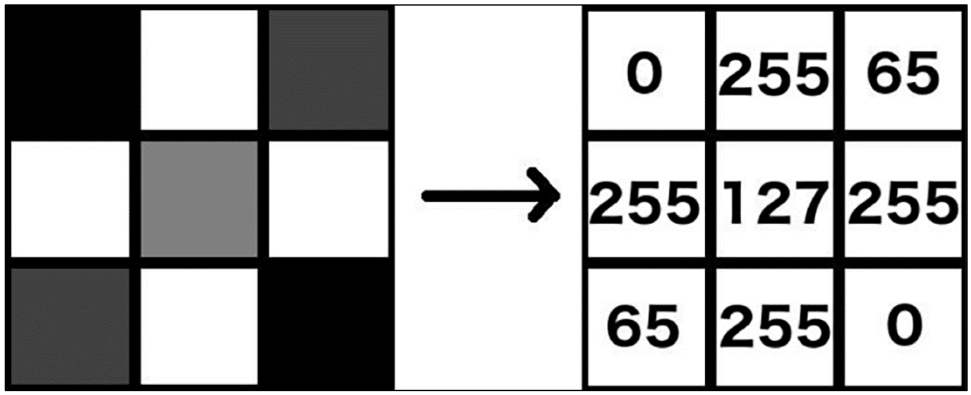


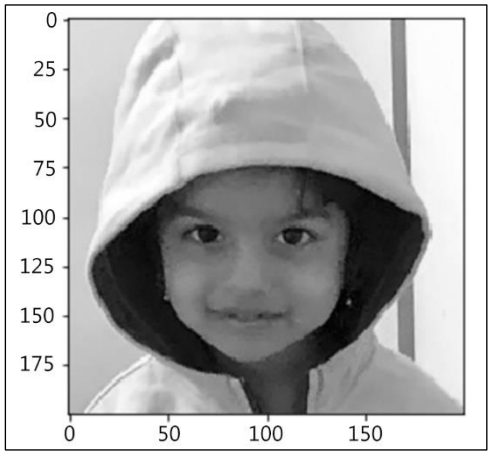

```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
├─Linear: 1-1                            [-1, 8]                   24
├─ReLU: 1-2                              [-1, 8]                   --
├─Linear: 1-3                            [-1, 1]                   9
==========================================================================================
Total params: 33
Trainable params: 33
Non-trainable params: 0
Total mult-adds (M): 0.00
==========================================================================================
Input size (MB): 0.00
Forward/backward pass size (MB): 0.00
Params size (MB): 0.00
Estimated Total Size (MB): 0.00
==========================================================================================
```

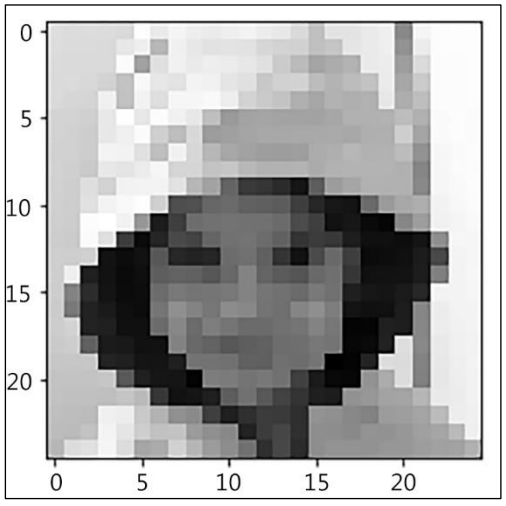
```
1  model.state_dict()
```

```
OrderedDict([('0.weight', tensor([[ 0.5090,  0.6708],
                      [-0.5887, -0.2970],
                      [ 0.3078, -0.4445],
                      [-0.3859,  0.0028],
                      [-0.1816,  0.9181],
                      [ 0.1532,  0.6011],
                      [ 0.2814, -0.4834],
                      [-0.6280, -0.5868]])),
             ('0.bias',
              tensor([ 0.7432, -0.5181, -0.1400,  0.3236, -0.1791, -0.4466, -0.1104, -0.1615])),
             ('2.weight',
              tensor([[ 0.9044, -0.2407, -0.1512, -0.2253,  0.5417,  0.4821,  0.1548,  0.0964]])),
             ('2.bias', tensor([0.0956]))])
```

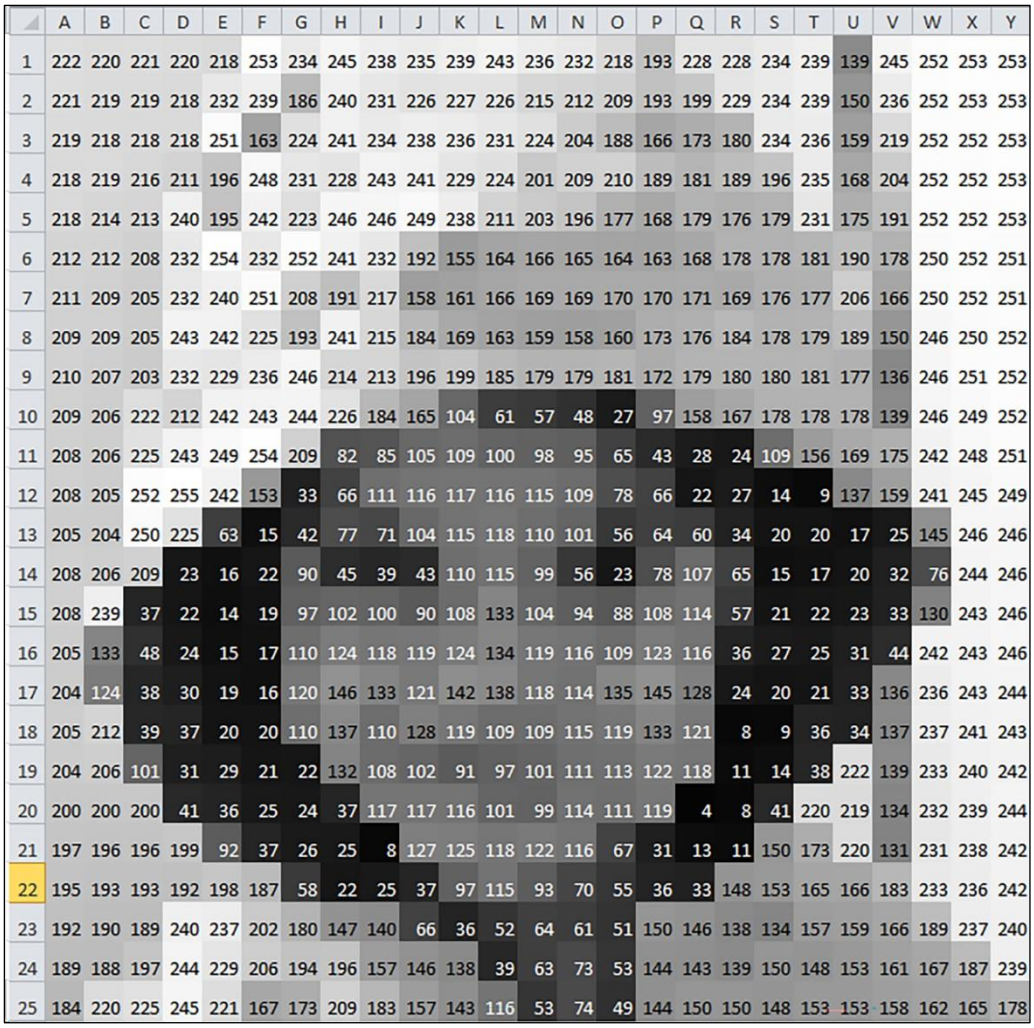
Chapter 3:

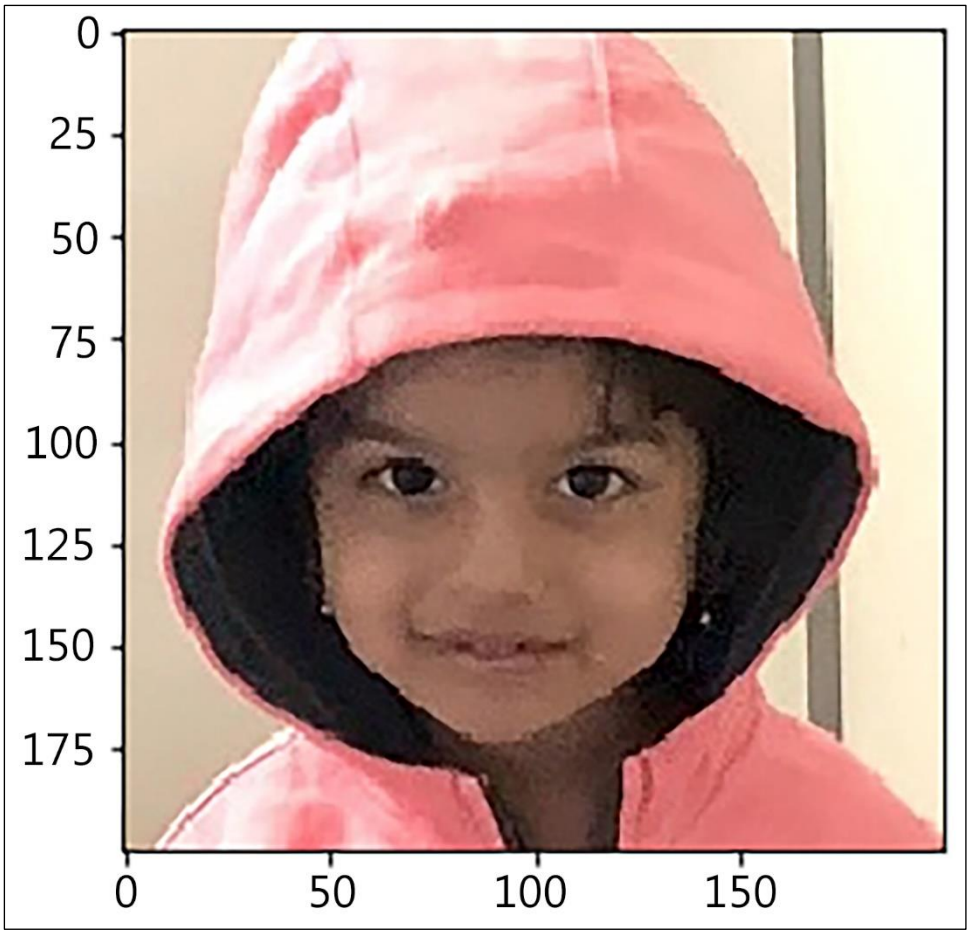


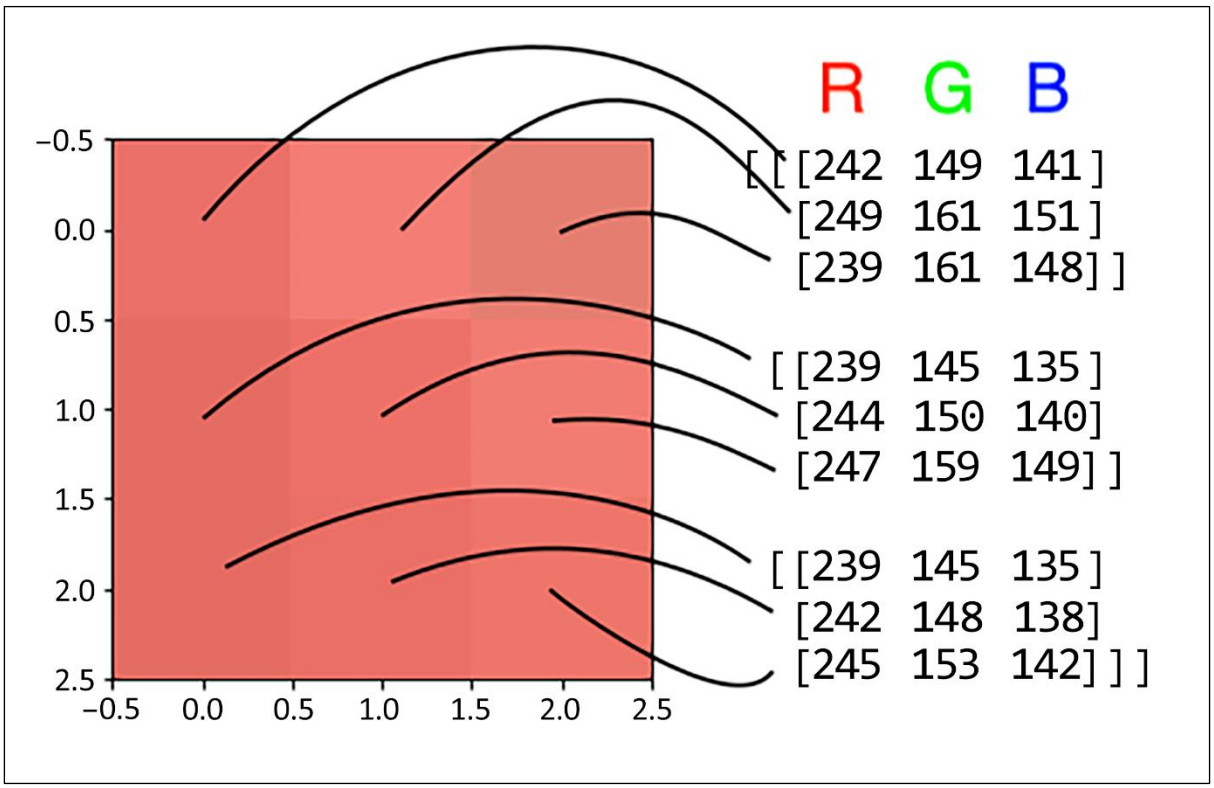


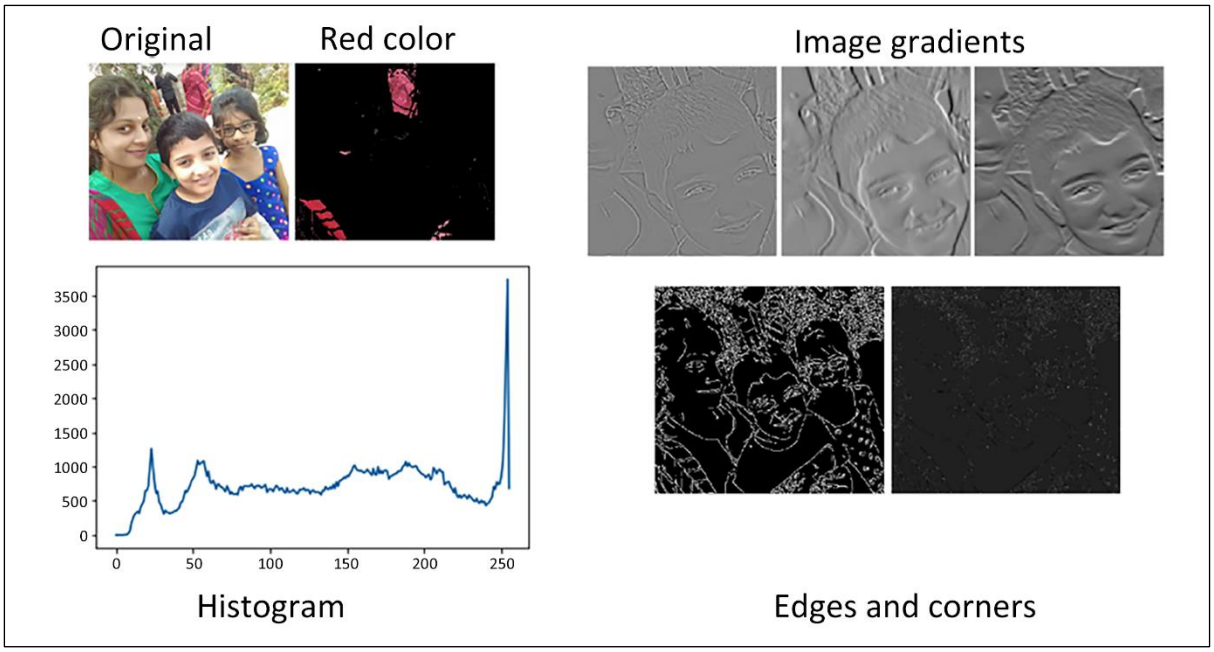



```
array([[222, 220, 221, 220, 218, 253, 234, 245, 238, 235, 239, 243, 236,
        232, 218, 193, 228, 228, 234, 239, 139, 245, 252, 253, 253],
       [221, 219, 219, 218, 232, 239, 186, 240, 231, 226, 227, 226, 215,
        212, 209, 193, 199, 229, 234, 239, 150, 236, 252, 253, 253],
       [219, 218, 218, 218, 251, 163, 224, 241, 234, 238, 236, 231, 224,
        204, 188, 166, 173, 180, 234, 236, 159, 219, 252, 252, 253],
       [218, 219, 216, 211, 196, 248, 231, 228, 243, 241, 229, 224, 201,
        209, 210, 189, 181, 189, 196, 235, 168, 204, 252, 252, 253],
```

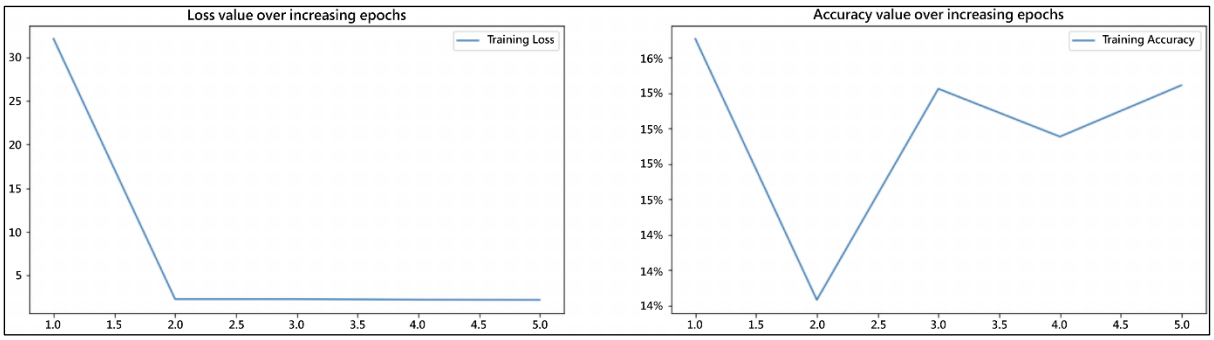


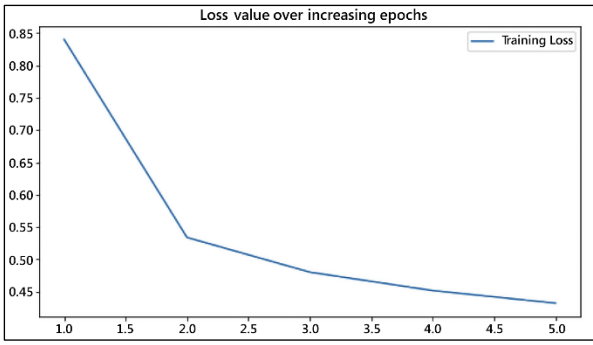

0
25
50
75
100
125
150
175
0
50
100
150

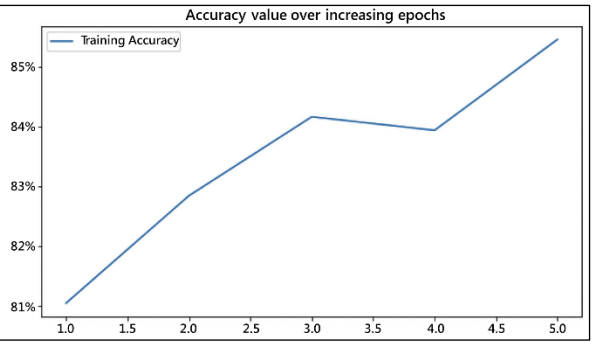

R G B
[[[242 149 141]
[249 161 151]
[239 161 148]]
[[239 145 135]
[244 150 140]
[247 159 149]]
[[239 145 135]
[242 148 138]
[245 153 142]]]
−0.5
0.0
0.5
1.0
1.5
2.0
2.5
−0.5
0.0
0.5
1.0
1.5
2.0
2.5




```
tr_images & tr_targets:
        X - torch.Size([60000, 28, 28])
        Y - torch.Size([60000])
        Y - Unique Values : tensor([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
TASK:
        10 class Classification
UNIQUE CLASSES:
        ['T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat', 'Sandal', 'Shirt', 'Sneaker', 'Bag', 'Ankle boot']
```

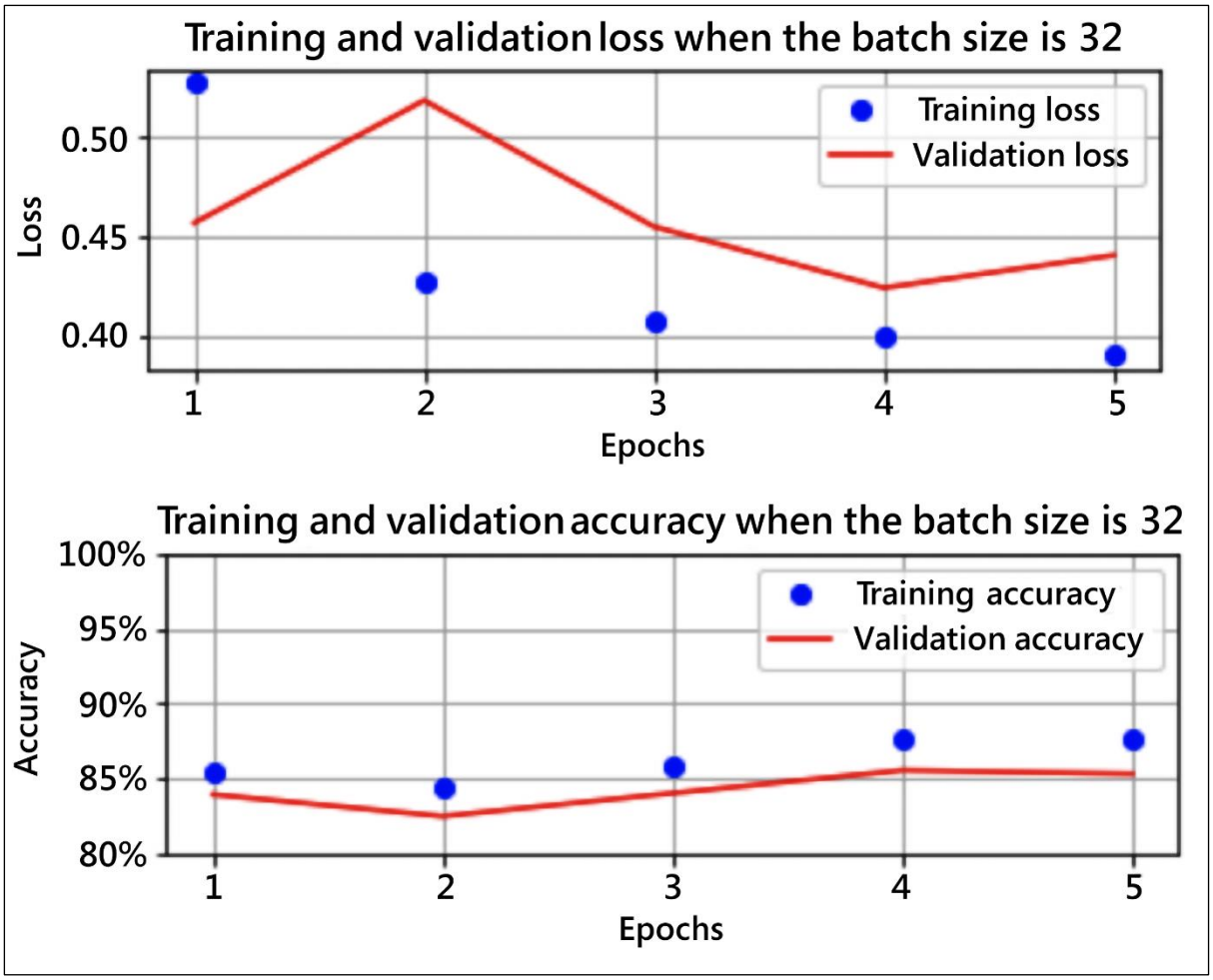


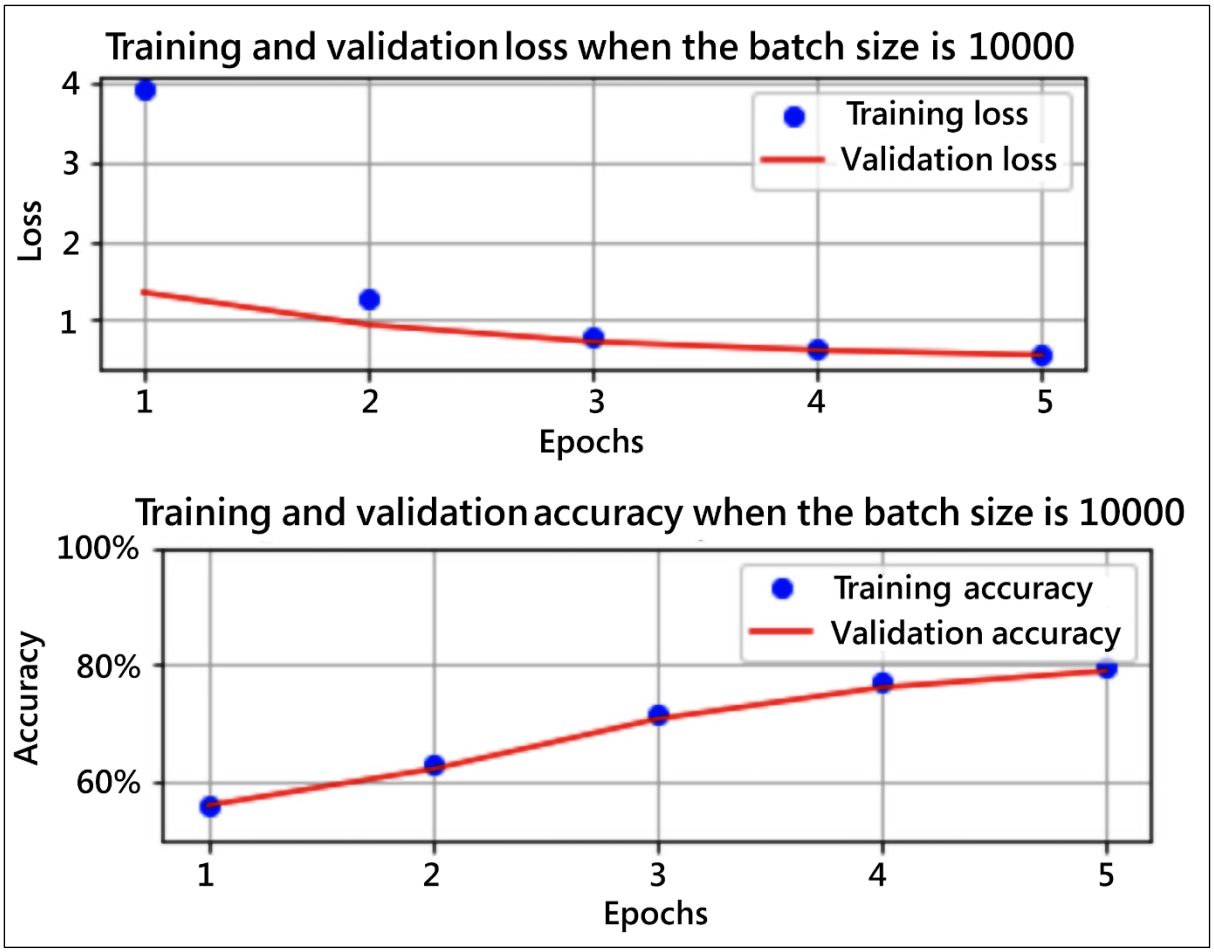


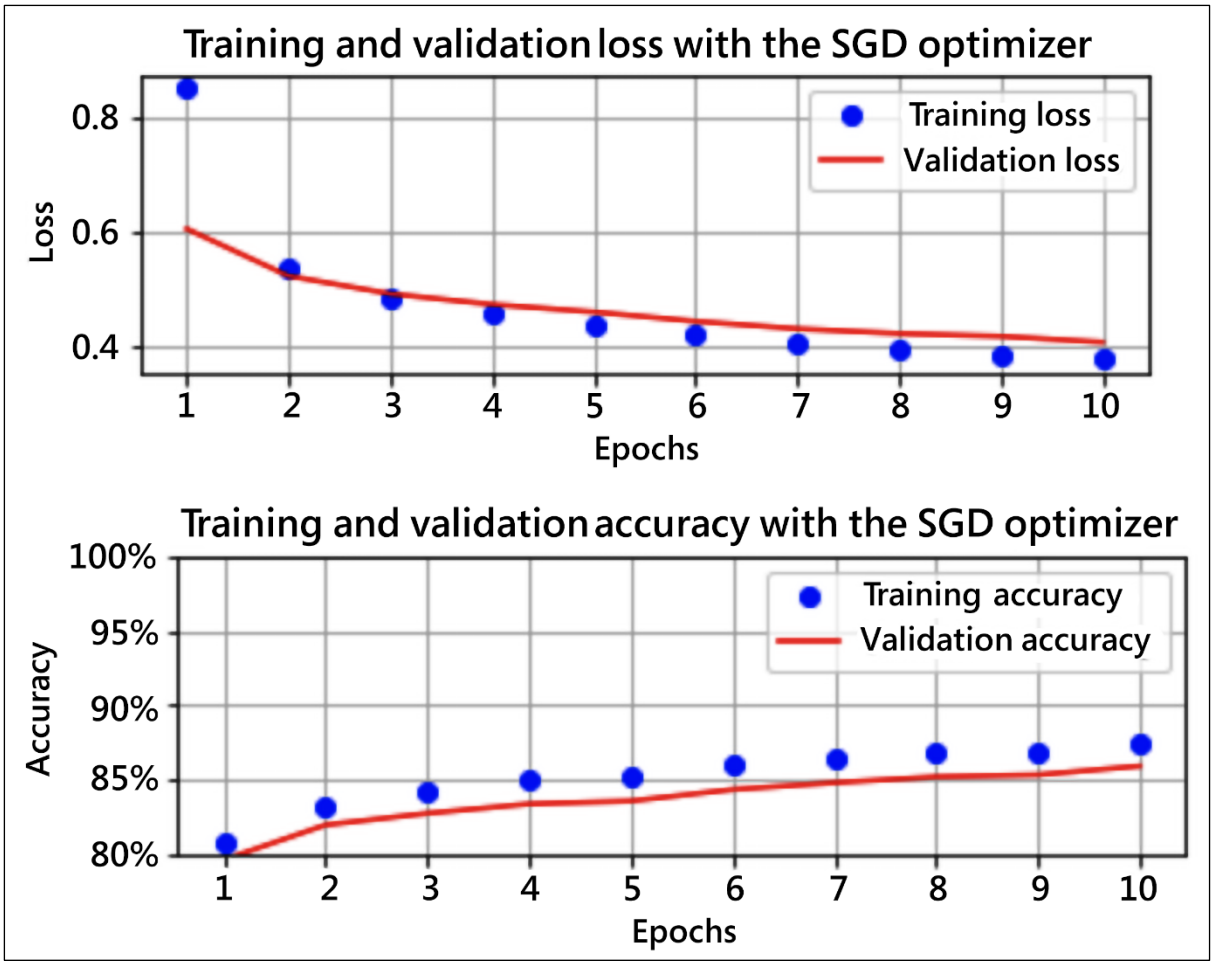


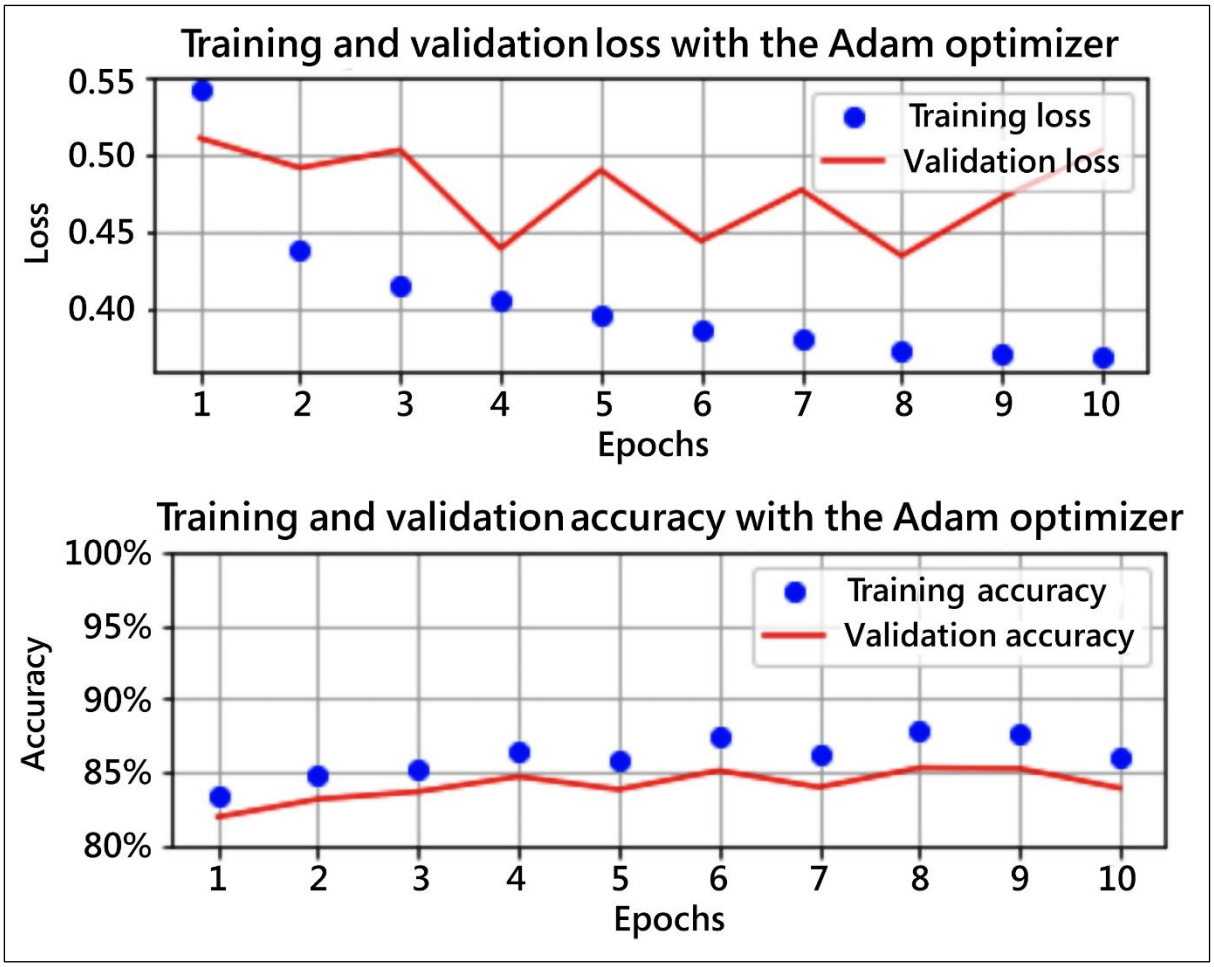
Input	Weight	Sigmoid
255	0.00001	0.501
255	0.0001	0.506
255	0.001	0.563
255	0.01	0.928
255	0.1	1.000
255	0.2	1.000
255	0.3	1.000
255	0.4	1.000
255	0.5	1.000
255	0.6	1.000
255	0.7	1.000
255	0.8	1.000
255	0.9	1.000
255	1	1.000

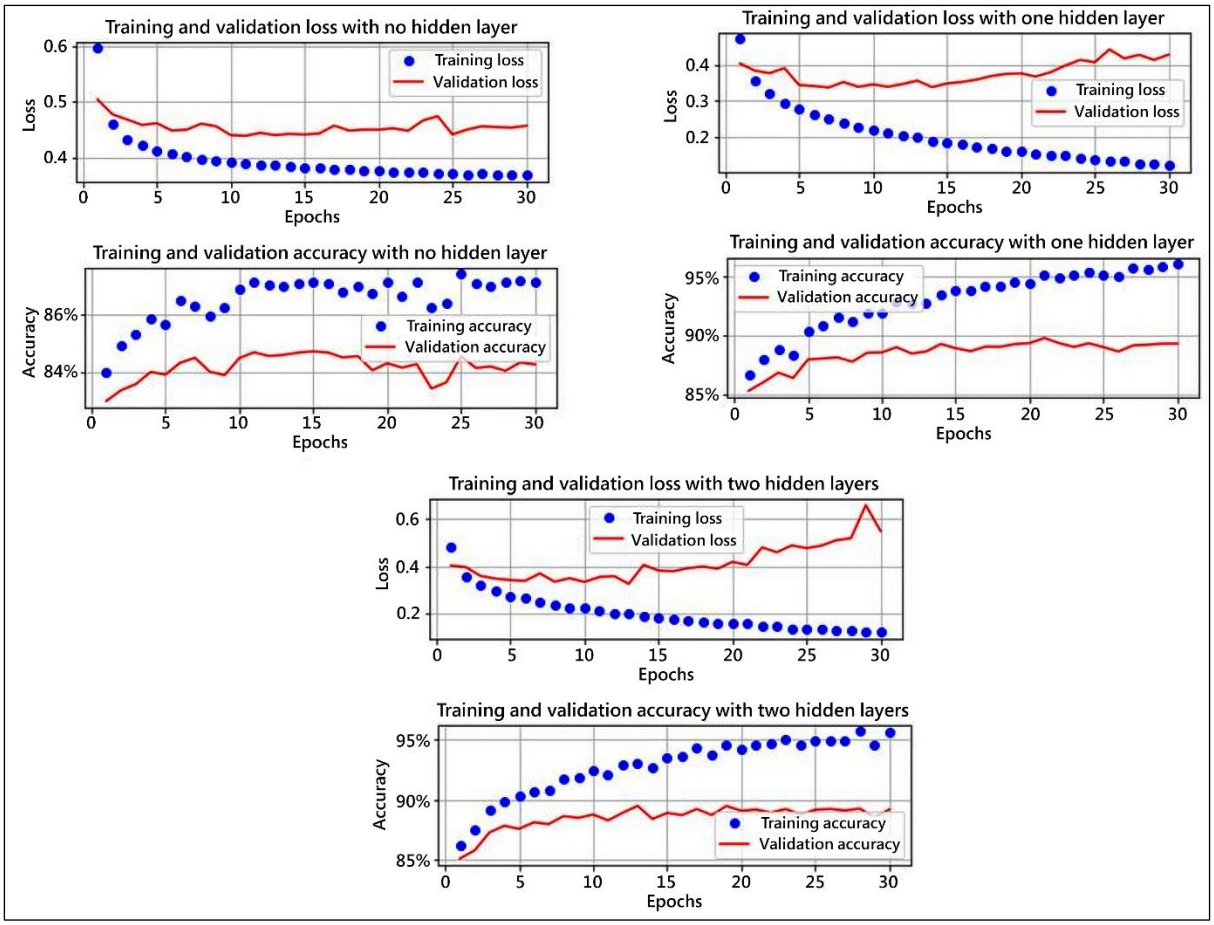
Input	Weight	Sigmoid
1	0.00001	0.500
1	0.0001	0.500
1	0.001	0.500
1	0.01	0.502
1	0.1	0.525
1	0.2	0.550
1	0.3	0.574
1	0.4	0.599
1	0.5	0.622
1	0.6	0.646
1	0.7	0.668
1	0.8	0.690
1	0.9	0.711
1	1	0.731


Training and validation loss when the batch size is 32
Training loss
Validation loss
Loss
0.50
0.45
0.40
1
2
3
4
5
Epochs
Training and validation accuracy when the batch size is 32
Training accuracy
Validation accuracy
Accuracy
100%
95%
90%
85%
80%
1
2
3
4
5
Epochs

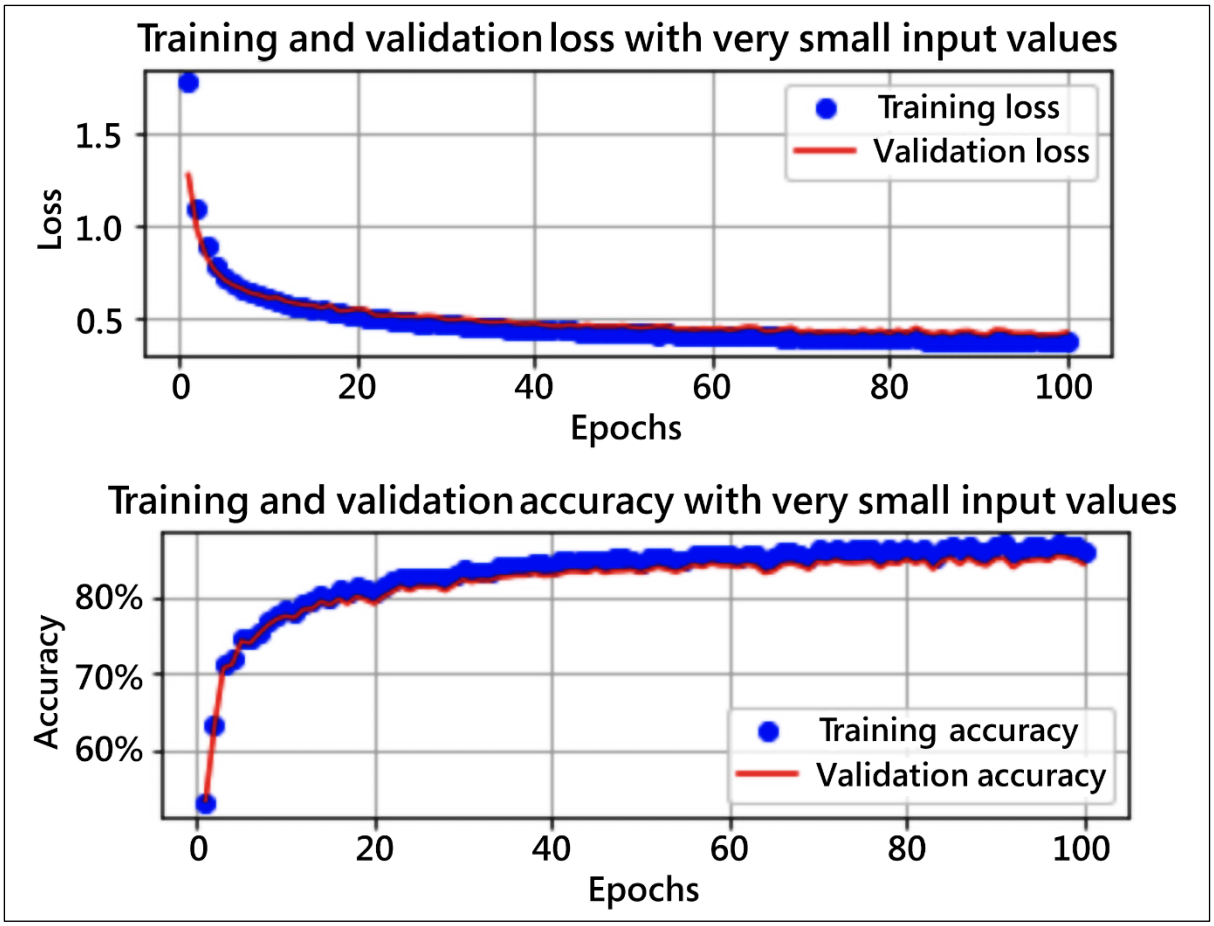

Training and validation loss when the batch size is 10000
Loss
Epochs
Training loss
Validation loss
Training and validation accuracy when the batch size is 10000
Accuracy
Epochs
Training accuracy
Validation accuracy

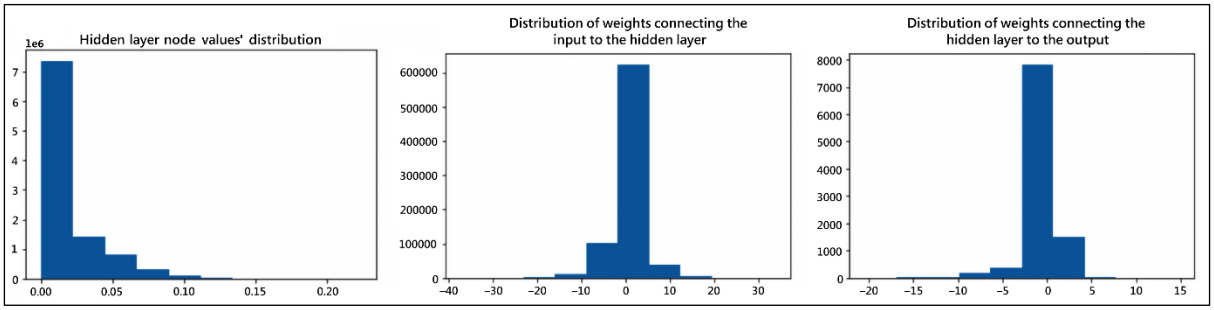

Training and validation loss with the SGD optimizer
Training loss
Validation loss
Loss
0.8
0.6
0.4
1
2
3
4
5
6
7
8
9
10
Epochs
Training and validation accuracy with the SGD optimizer
Training accuracy
Validation accuracy
Accuracy
100%
95%
90%
85%
80%
1
2
3
4
5
6
7
8
9
10
Epochs

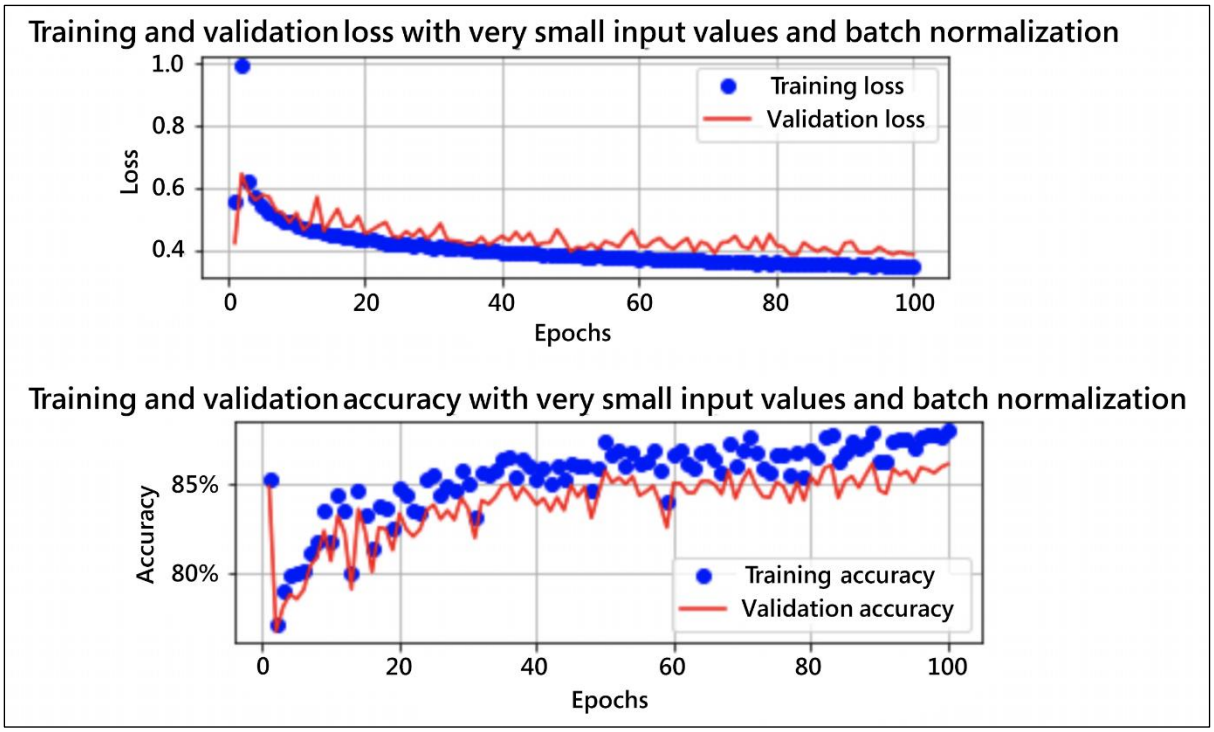

Training and validation loss with the Adam optimizer
Loss
0.55
0.50
0.45
0.40
Training loss
Validation loss
1
2
3
4
5
6
7
8
9
10
Epochs
Training and validation accuracy with the Adam optimizer
Accuracy
100%
95%
90%
85%
80%
Training accuracy
Validation accuracy
1
2
3
4
5
6
7
8
9
10
Epochs

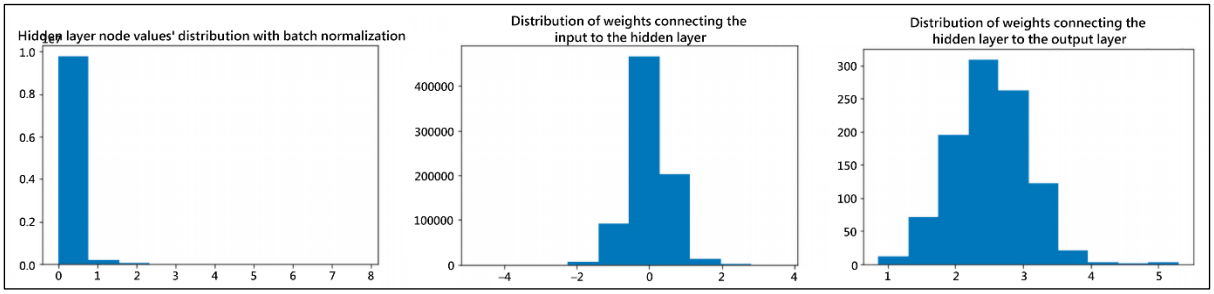


Input	Weight	Sigmoid
0.01	0.00001	0.500
0.01	0.0001	0.500
0.01	0.001	0.500
0.01	0.01	0.500
0.01	0.1	0.500
0.01	0.2	0.500
0.01	0.3	0.501
0.01	0.4	0.501
0.01	0.5	0.501
0.01	0.6	0.501
0.01	0.7	0.502
0.01	0.8	0.502
0.01	0.9	0.502
0.01	1	0.502


Training and validation loss with very small input values
Training loss
Validation loss
Loss
1.5
1.0
0.5
0
20
40
60
80
100
Epochs
Training and validation accuracy with very small input values
Training accuracy
Validation accuracy
Accuracy
80%
70%
60%
0
20
40
60
80
100
Epochs


Hidden layer node values' distribution
1e6
Distribution of weights connecting the input to the hidden layer
Distribution of weights connecting the hidden layer to the output



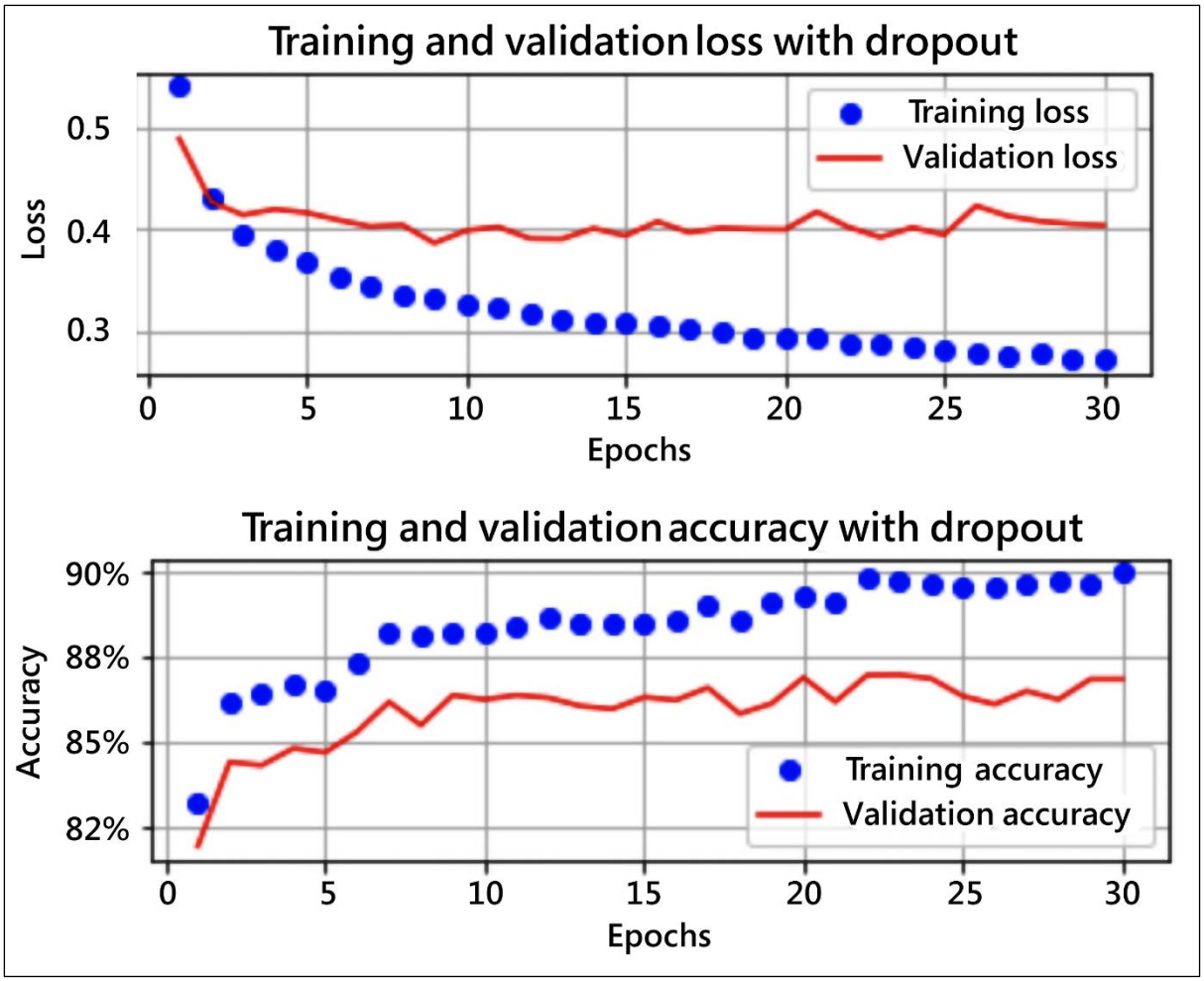


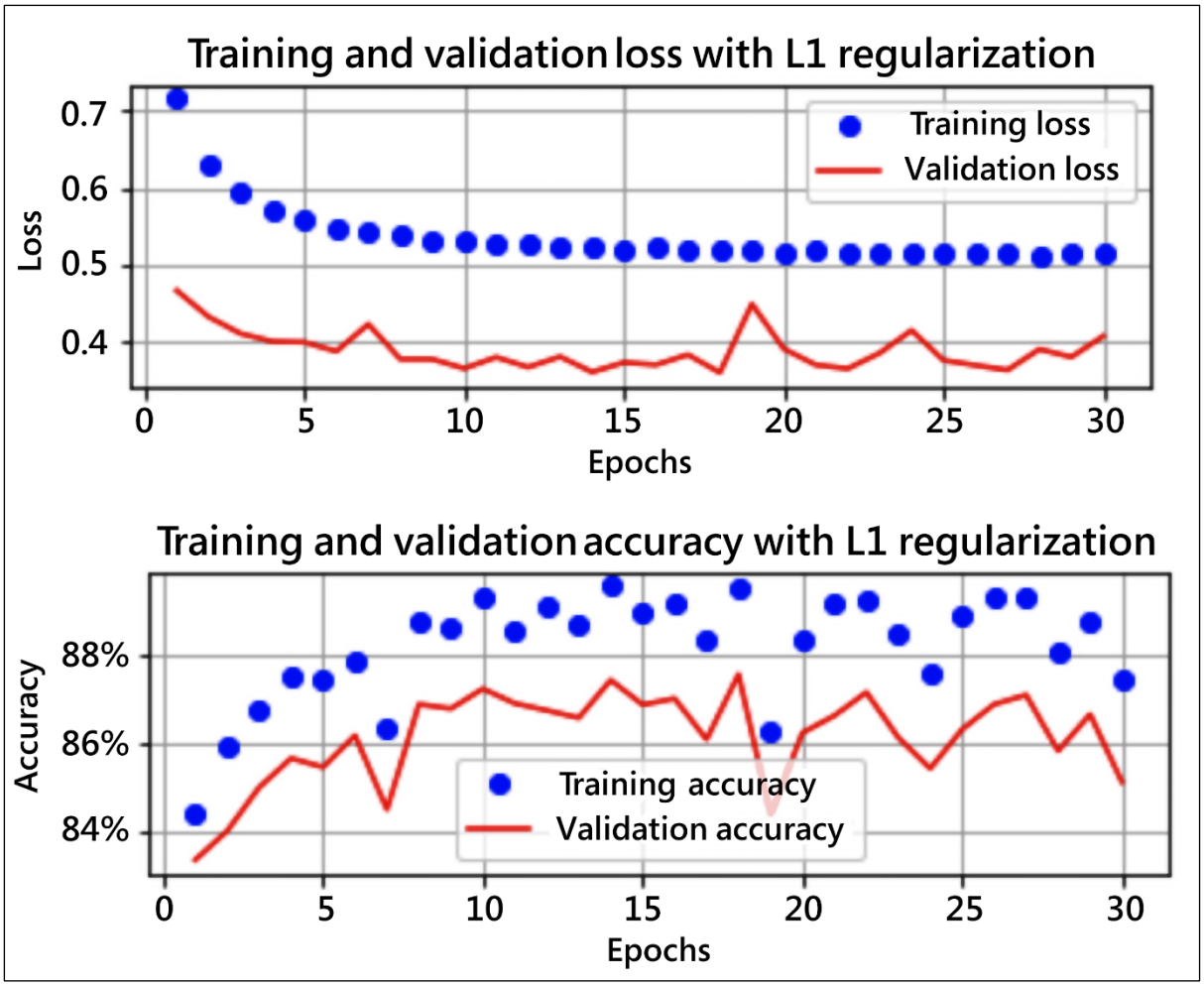

```
[20]  1  model.train()
      2  batch = next(iter(trn_dl))
      3  for i in range(5):
      4      output = model(batch[0])
      5      print(output.mean().item())
```

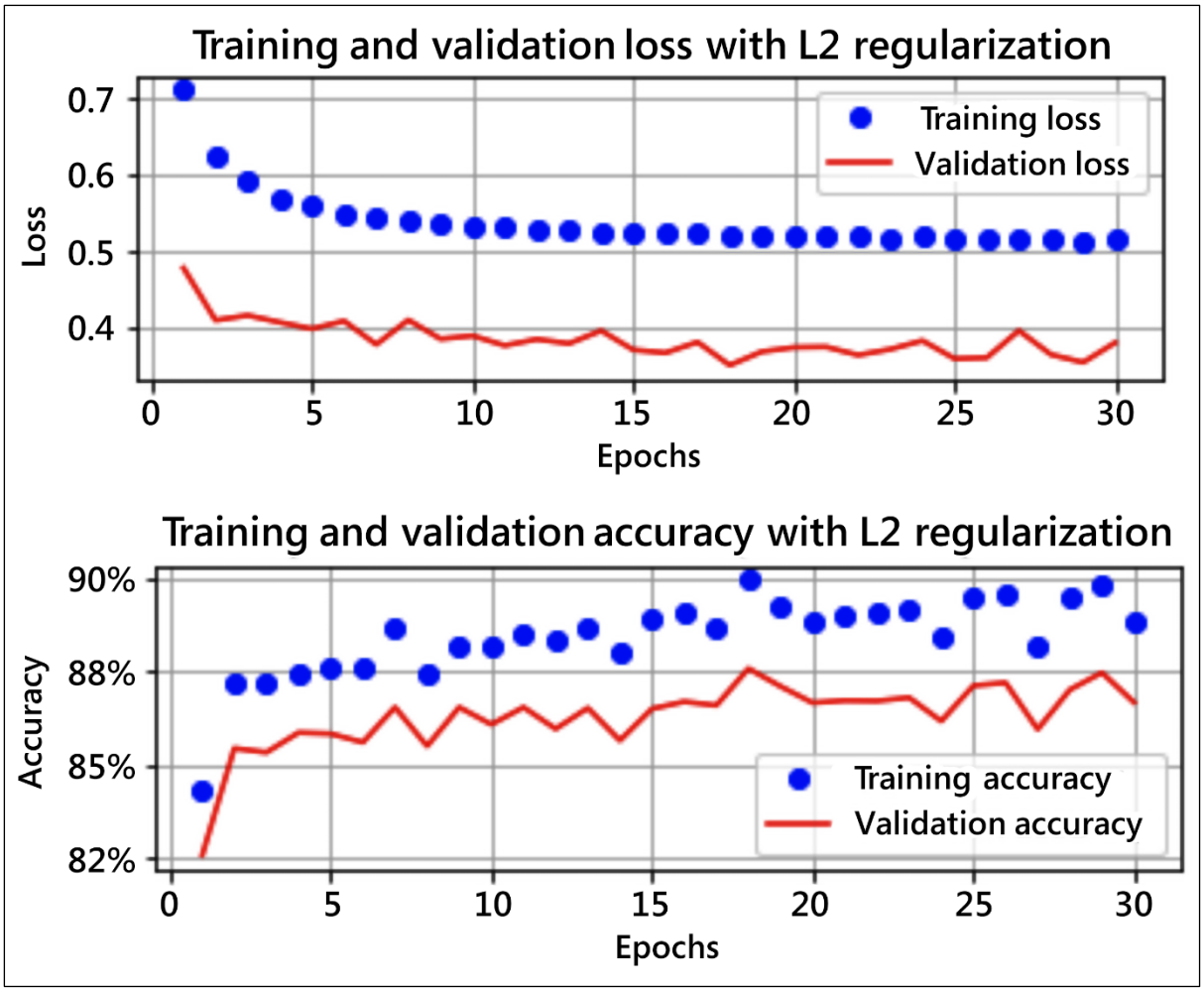
```
-13.275323867797852
-12.834677696228027
-11.895054817199707
-12.713885307312012
-13.302783012390137
```

```
      1  model.eval()
      2  batch = next(iter(trn_dl))
      3  for i in range(5):
      4      output = model(batch[0])
      5      print(output.mean().item())
```

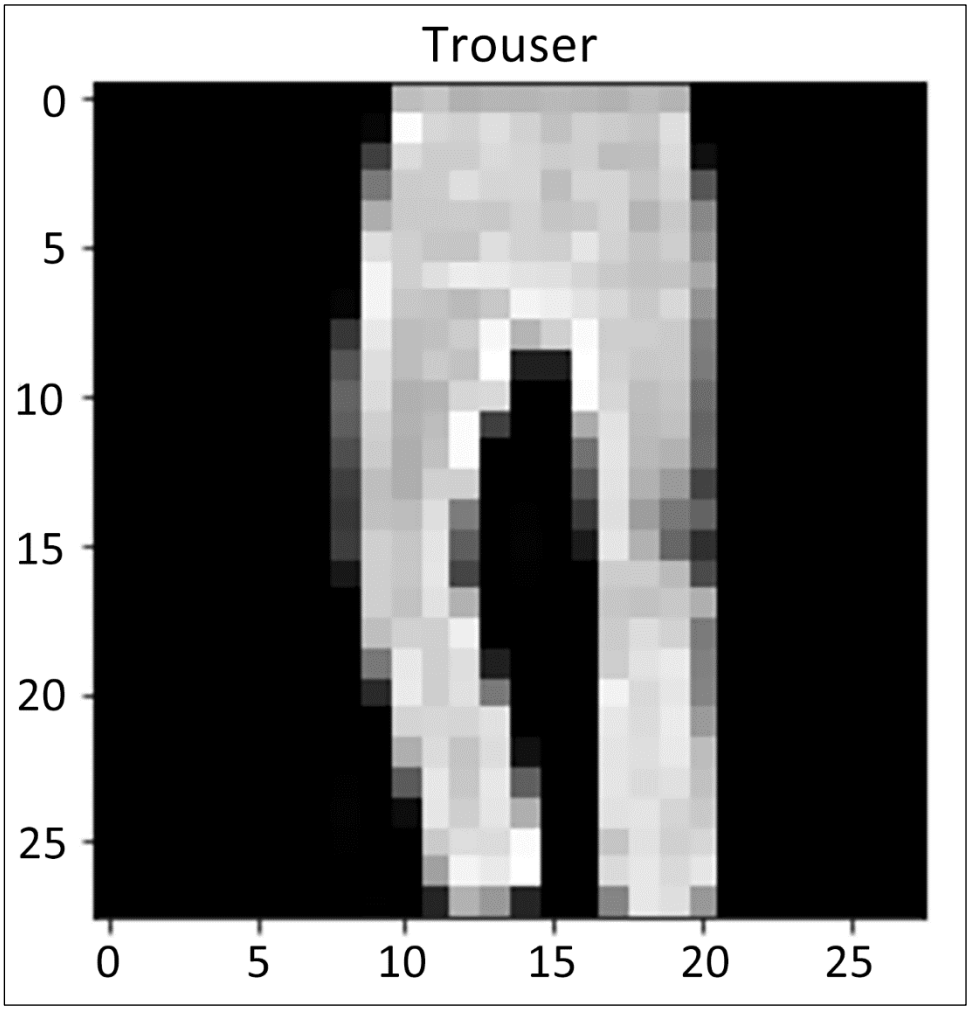
```
-12.40117359161377
-12.40117359161377
-12.40117359161377
-12.40117359161377
-12.40117359161377
```


Training and validation loss with dropout
Training loss
Validation loss
Loss
0.5
0.4
0.3
0
5
10
15
20
25
30
Epochs
Training and validation accuracy with dropout
Training accuracy
Validation accuracy
Accuracy
90%
88%
85%
82%
0
5
10
15
20
25
30
Epochs

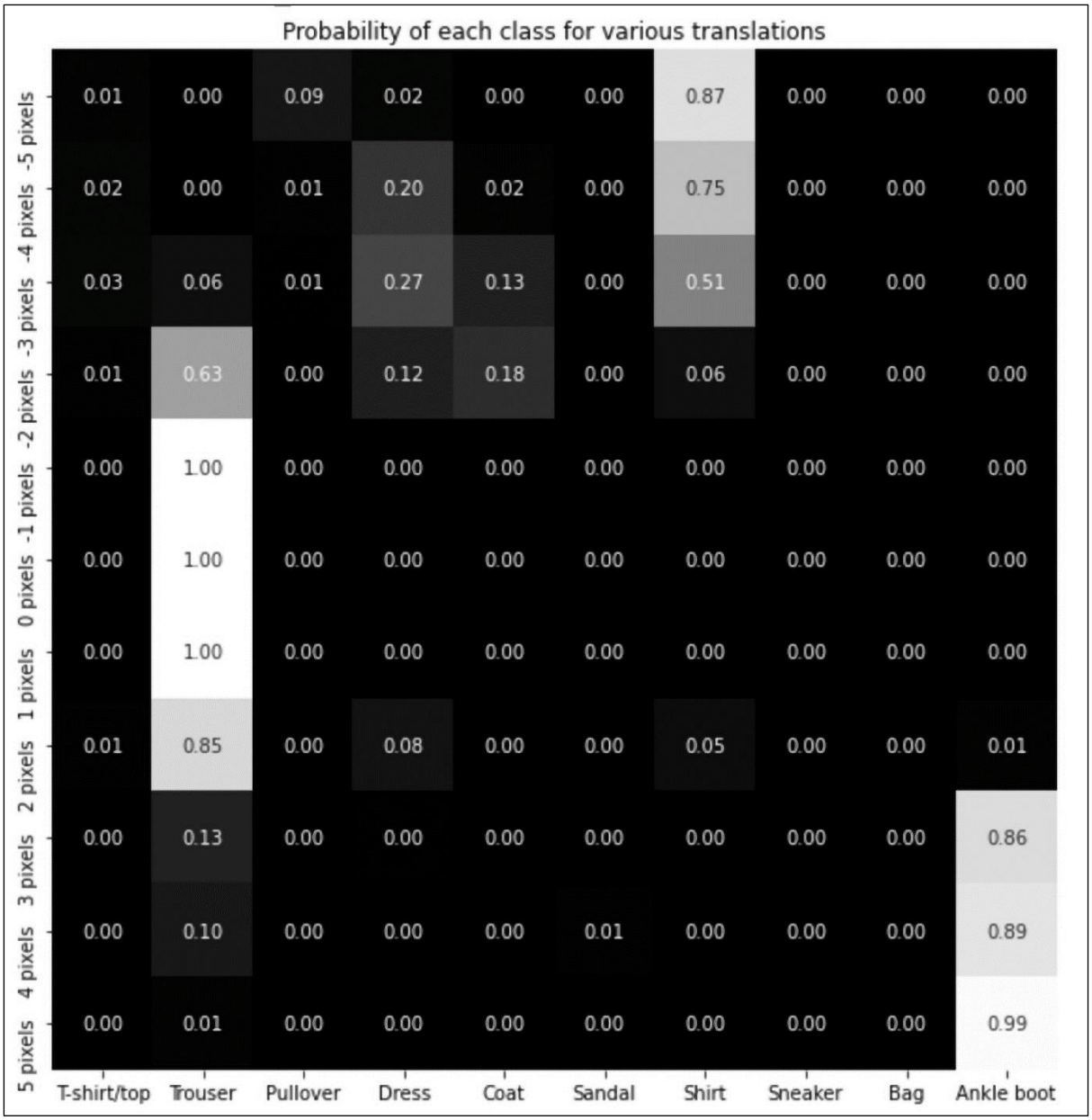

Training and validation loss with L1 regularization
Loss
0.7
0.6
0.5
0.4
0
5
10
15
20
25
30
Epochs
Training loss
Validation loss
Training and validation accuracy with L1 regularization
Accuracy
88%
86%
84%
0
5
10
15
20
25
30
Epochs
Training accuracy
Validation accuracy


Training and validation loss with L2 regularization
Loss
0.7
0.6
0.5
0.4
0
5
10
15
20
25
30
Epochs
Training loss
Validation loss
Training and validation accuracy with L2 regularization
Accuracy
90%
88%
85%
82%
0
5
10
15
20
25
30
Epochs
Training accuracy
Validation accuracy

Chapter 4:



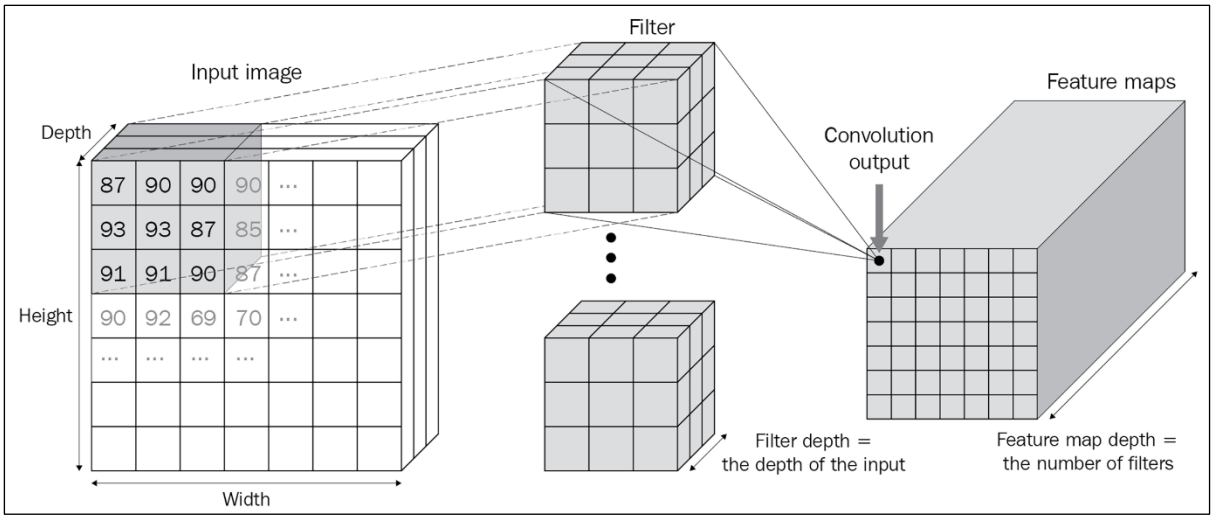

```
array([1.7608897e-08, 1.0000000e+00, 2.6042574e-13, 1.1353759e-10,
       3.1050048e-12, 7.2957764e-16, 8.0109371e-11, 3.8039204e-22,
       1.2800090e-15, 2.8759430e-18], dtype=float32)
```



1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

1	2
3	4

44	54	64
84	94	104
124	134	144



44	64
124	144

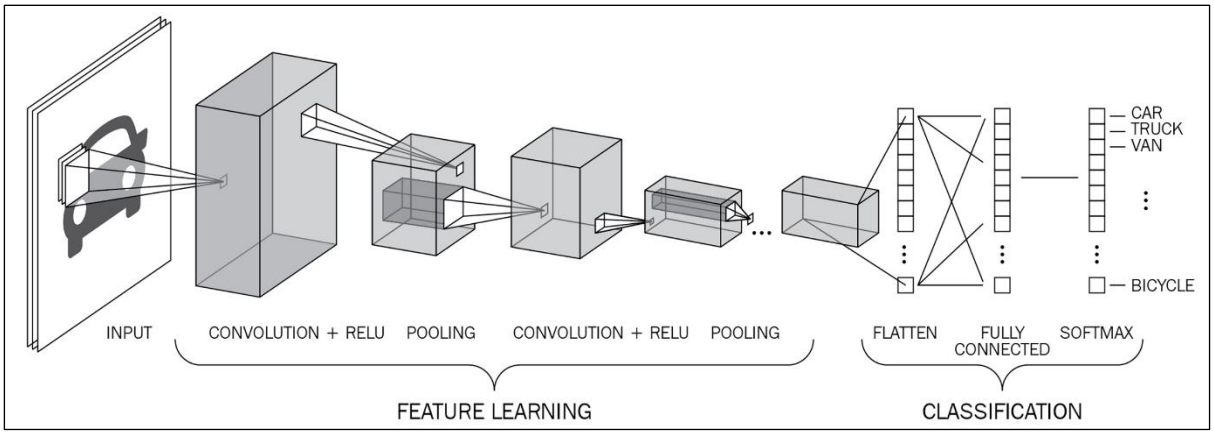
0	0	0	0	0	0
0	1	2	3	4	0
0	5	6	7	8	0
0	9	10	11	12	0
0	13	14	15	16	0
0	0	0	0	0	0

1	2
3	4

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

6	8
14	16

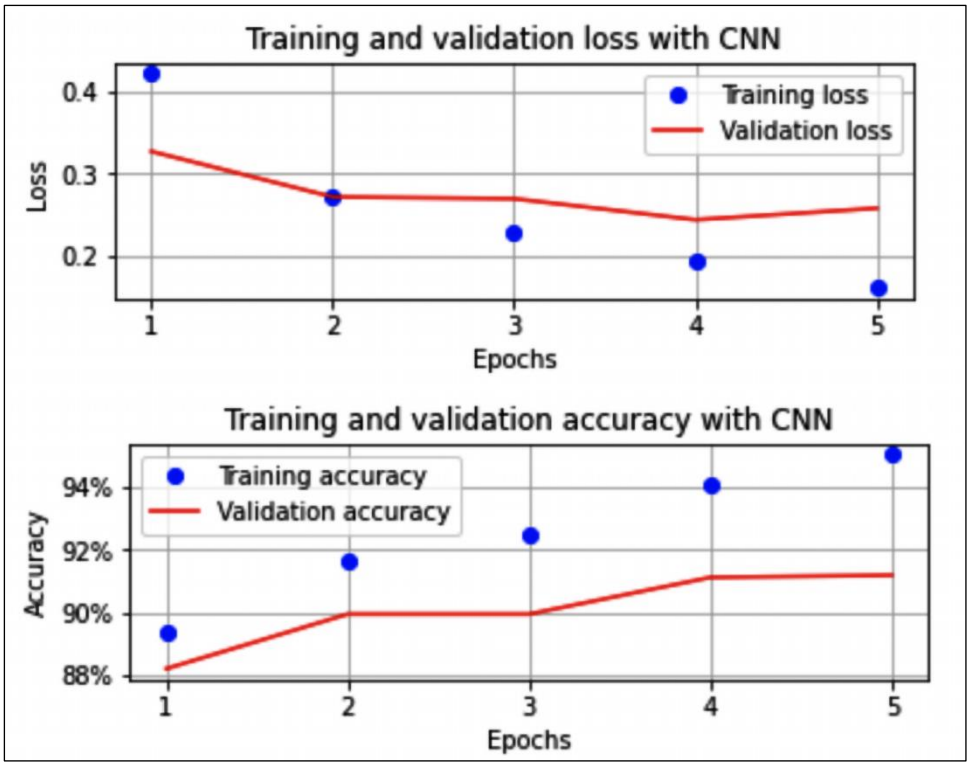


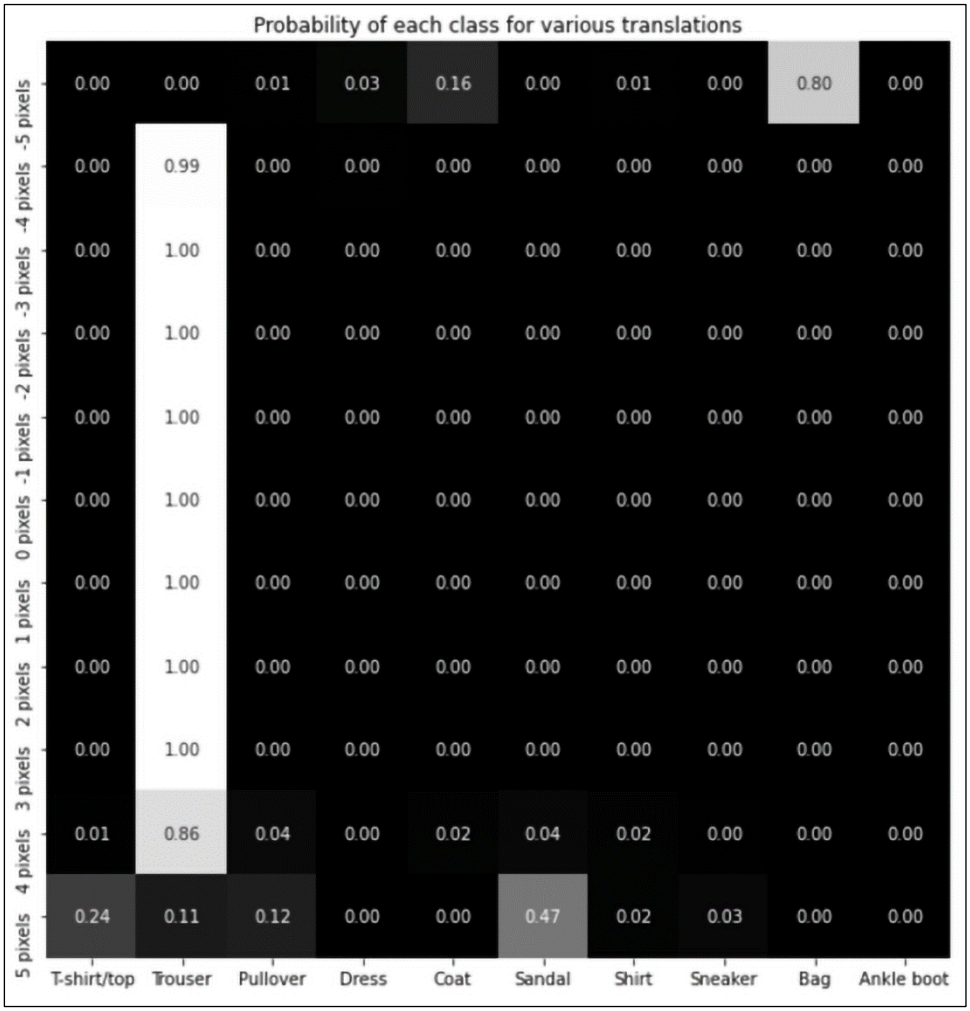

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1            [-1, 1, 2, 2]              10
         MaxPool2d-2            [-1, 1, 1, 1]               0
              ReLU-3            [-1, 1, 1, 1]               0
           Flatten-4                  [-1, 1]               0
            Linear-5                  [-1, 1]               2
           Sigmoid-6                  [-1, 1]               0
================================================================
Total params: 12
Trainable params: 12
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.00
Forward/backward pass size (MB): 0.00
Params size (MB): 0.00
Estimated Total Size (MB): 0.00
----------------------------------------------------------------
```

```
help(nn.Conv2d)
```

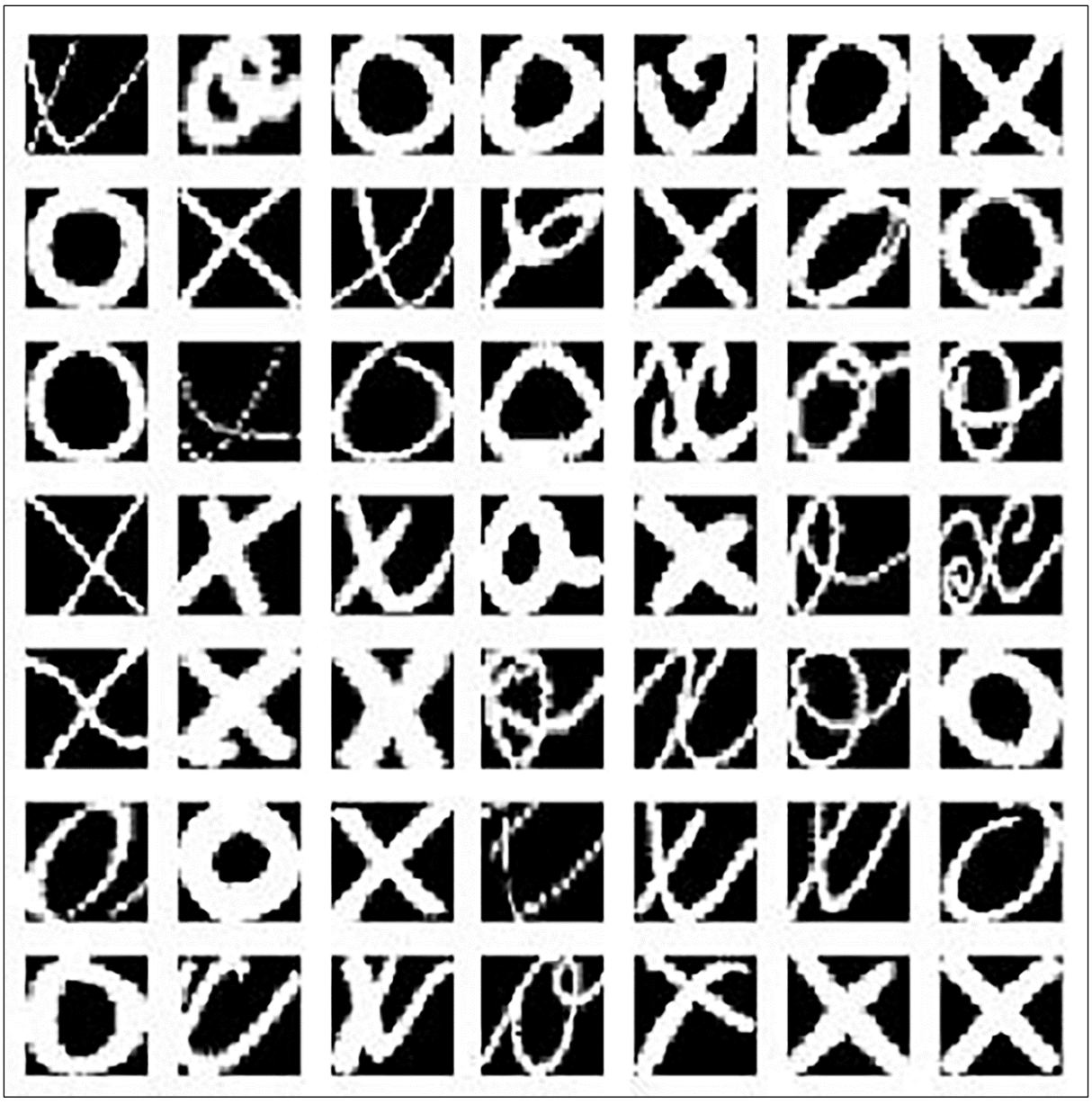
```
  Args:
      in_channels (int): Number of channels in the input image
      out_channels (int): Number of channels produced by the convolution
      kernel_size (int or tuple): Size of the convolving kernel
      stride (int or tuple, optional): Stride of the convolution. Default: 1
      padding (int or tuple, optional): Zero-padding added to both sides of the input. Default: 0
      padding_mode (string, optional). Accepted values `zeros` and `circular` Default: `zeros`
      dilation (int or tuple, optional): Spacing between kernel elements. Default: 1
      groups (int, optional): Number of blocked connections from input channels to output channels. Default: 1
      bias (bool, optional): If ``True``, adds a learnable bias to the output. Default: ``True``
```

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1          [-1, 64, 26, 26]             640
         MaxPool2d-2          [-1, 64, 13, 13]               0
              ReLU-3          [-1, 64, 13, 13]               0
            Conv2d-4         [-1, 128, 11, 11]          73,856
         MaxPool2d-5           [-1, 128, 5, 5]               0
              ReLU-6           [-1, 128, 5, 5]               0
           Flatten-7                [-1, 3200]               0
            Linear-8                 [-1, 256]         819,456
              ReLU-9                 [-1, 256]               0
           Linear-10                  [-1, 10]           2,570
================================================================
Total params: 896,522
Trainable params: 896,522
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.00
Forward/backward pass size (MB): 0.69
Params size (MB): 3.42
Estimated Total Size (MB): 4.11
----------------------------------------------------------------
```

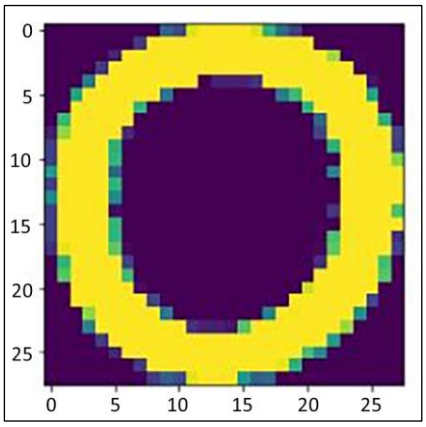

Training and validation loss with CNN
Training loss
Validation loss
Loss
0.4
0.3
0.2
1
2
3
4
5
Epochs
Training and validation accuracy with CNN
Training accuracy
Validation accuracy
Accuracy
94%
92%
90%
88%
1
2
3
4
5
Epochs

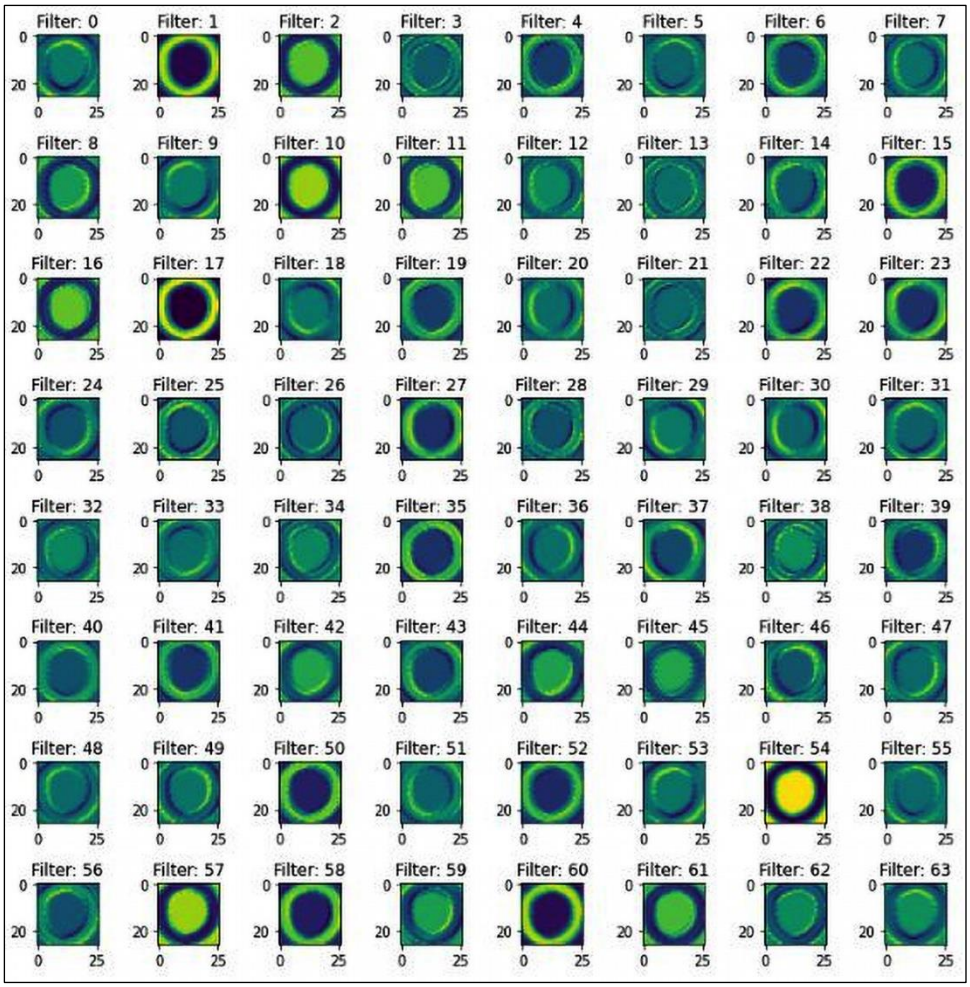

Probability of each class for various translations
-5 pixels
-4 pixels
-3 pixels
-2 pixels
-1 pixels
0 pixels
1 pixels
2 pixels
3 pixels
4 pixels
5 pixels
0.00 0.00 0.01 0.03 0.16 0.00 0.01 0.00 0.80 0.00
0.00 0.99 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 1.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 1.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 1.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 1.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 1.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 1.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 1.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.01 0.86 0.04 0.00 0.02 0.04 0.02 0.00 0.00 0.00
0.24 0.11 0.12 0.00 0.00 0.47 0.02 0.03 0.00 0.00
T-shirt/top
Trouser
Pullover
Dress
Coat
Sandal
Shirt
Sneaker
Bag
Ankle boot

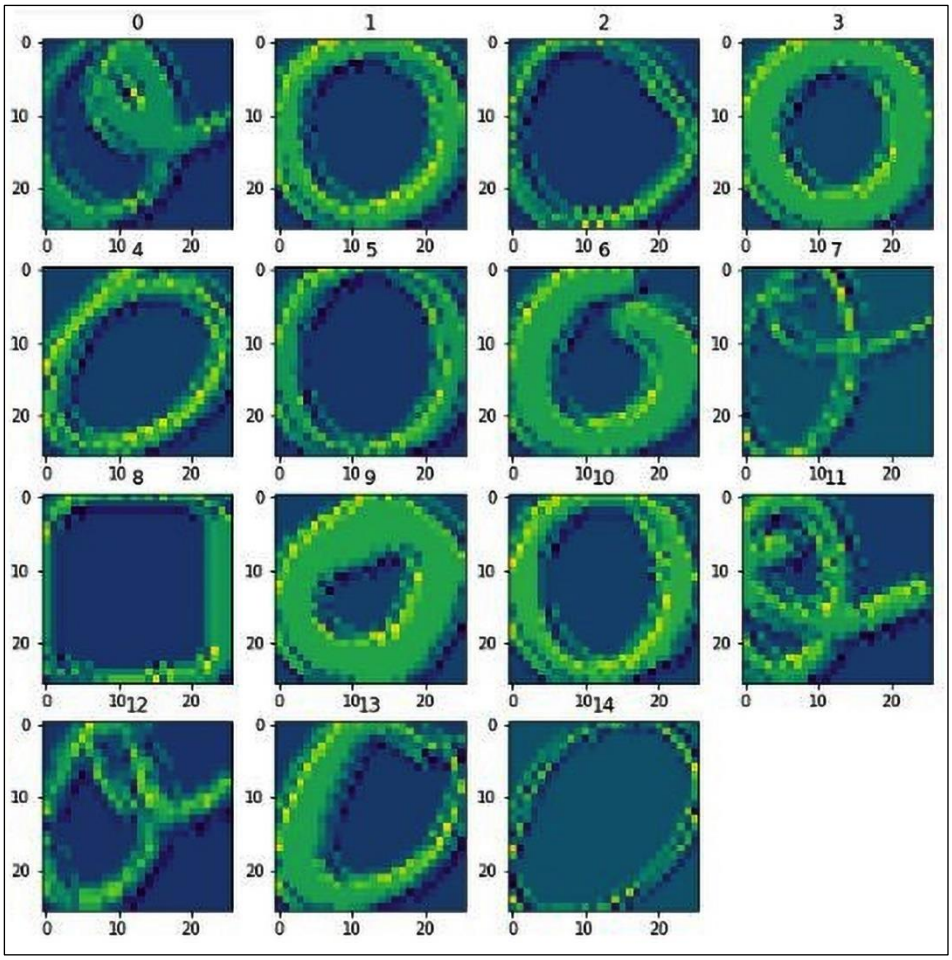
all/o@InterconnectedDemo-Bold@IttL47.png
all/o@Refresh-Regular@LX2MG4.png
all/x@CallistaOllander@7EWgpq.png
all/x@ChristmasSeason@xZ7mjB.png

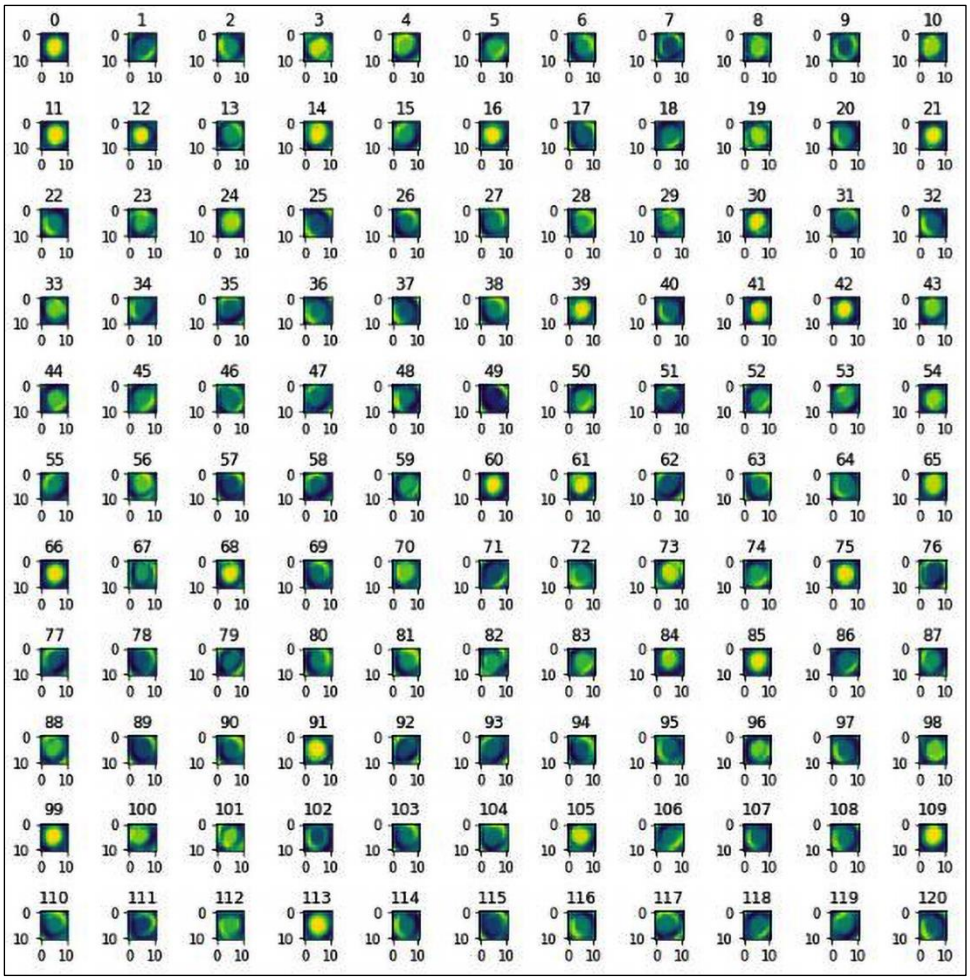


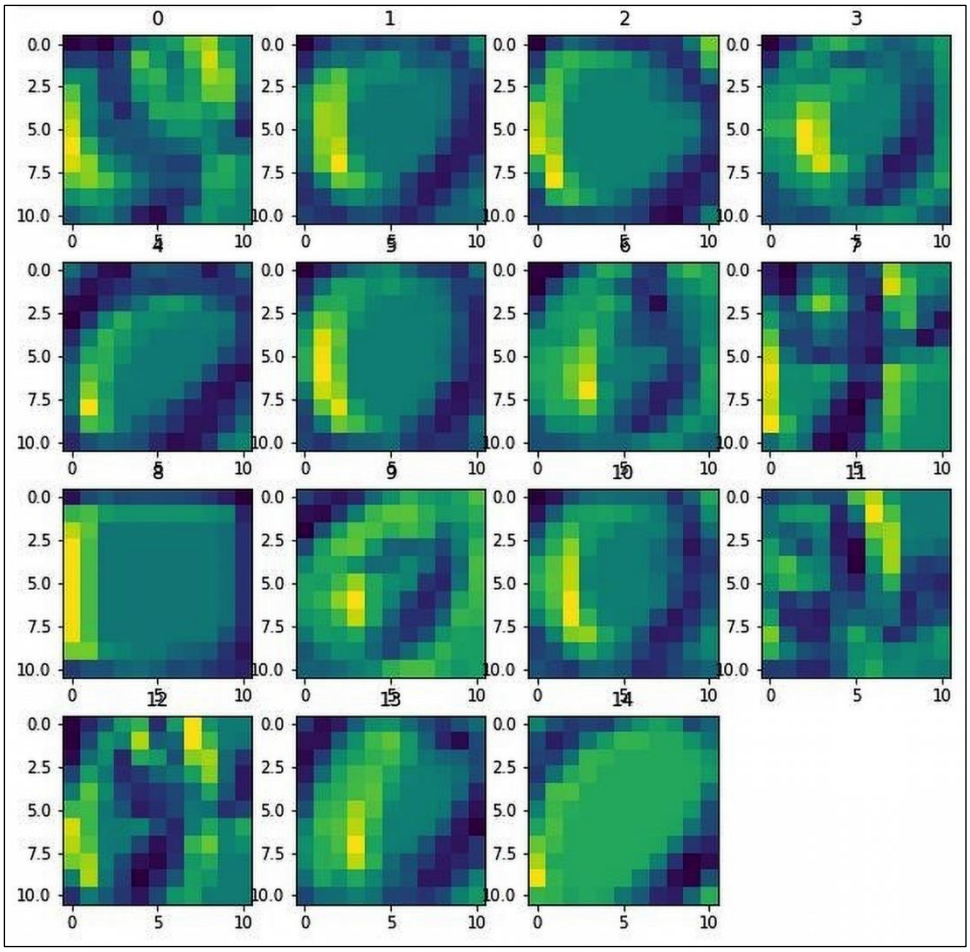
```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1          [-1, 64, 26, 26]             640
         MaxPool2d-2          [-1, 64, 13, 13]               0
              ReLU-3          [-1, 64, 13, 13]               0
            Conv2d-4         [-1, 128, 11, 11]          73,856
         MaxPool2d-5           [-1, 128, 5, 5]               0
              ReLU-6           [-1, 128, 5, 5]               0
           Flatten-7                [-1, 3200]               0
            Linear-8                 [-1, 256]         819,456
              ReLU-9                 [-1, 256]               0
           Linear-10                   [-1, 1]             257
          Sigmoid-11                   [-1, 1]               0
================================================================
Total params: 894,209
Trainable params: 894,209
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.00
Forward/backward pass size (MB): 0.69
Params size (MB): 3.41
Estimated Total Size (MB): 4.10
----------------------------------------------------------------
```

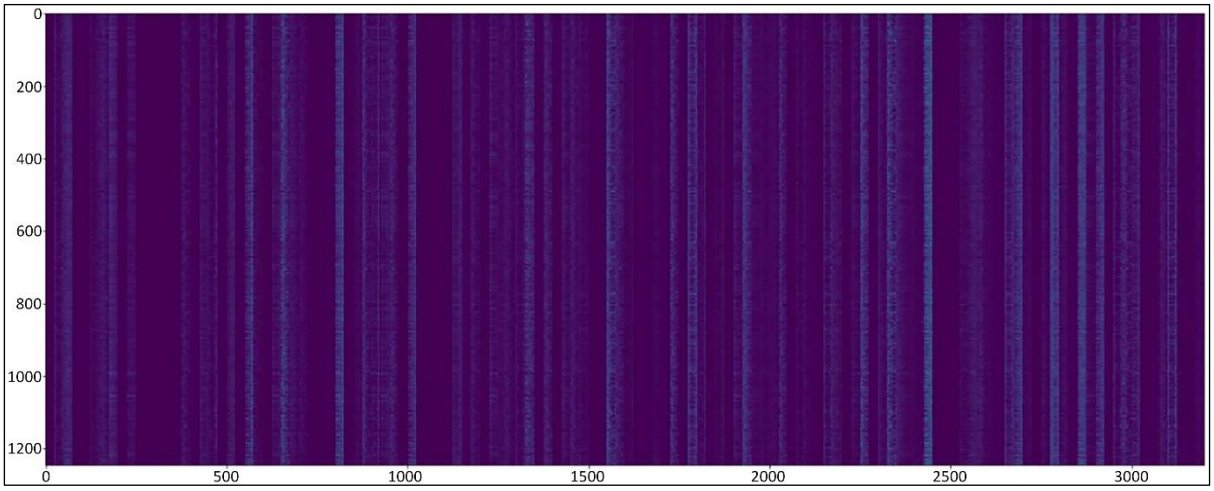



Filter: 0
Filter: 1
Filter: 2
Filter: 3
Filter: 4
Filter: 5
Filter: 6
Filter: 7
Filter: 8
Filter: 9
Filter: 10
Filter: 11
Filter: 12
Filter: 13
Filter: 14
Filter: 15
Filter: 16
Filter: 17
Filter: 18
Filter: 19
Filter: 20
Filter: 21
Filter: 22
Filter: 23
Filter: 24
Filter: 25
Filter: 26
Filter: 27
Filter: 28
Filter: 29
Filter: 30
Filter: 31
Filter: 32
Filter: 33
Filter: 34
Filter: 35
Filter: 36
Filter: 37
Filter: 38
Filter: 39
Filter: 40
Filter: 41
Filter: 42
Filter: 43
Filter: 44
Filter: 45
Filter: 46
Filter: 47
Filter: 48
Filter: 49
Filter: 50
Filter: 51
Filter: 52
Filter: 53
Filter: 54
Filter: 55
Filter: 56
Filter: 57
Filter: 58
Filter: 59
Filter: 60
Filter: 61
Filter: 62
Filter: 63
0
20
0
25

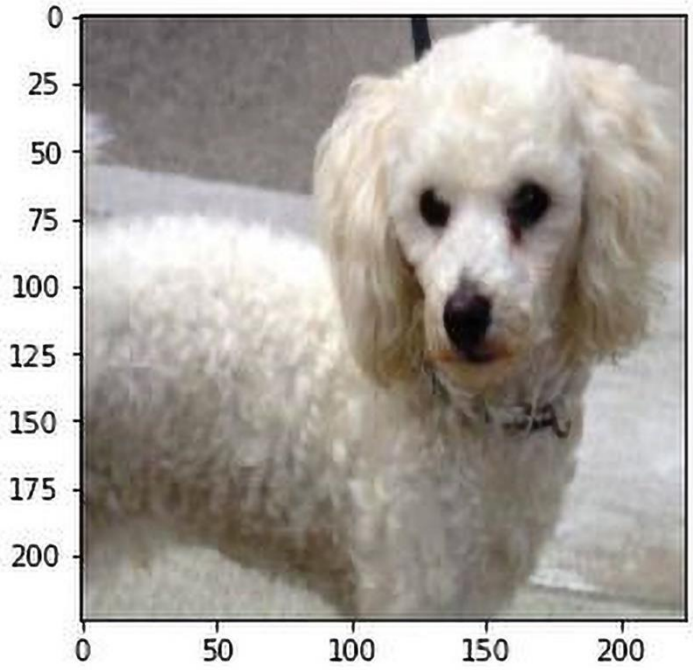

0
1
2
3
4
5
6
7
8
9
10
11
12
13
14


0 1 2 3 4 5 6 7 8 9 10
11 12 13 14 15 16 17 18 19 20 21
22 23 24 25 26 27 28 29 30 31 32
33 34 35 36 37 38 39 40 41 42 43
44 45 46 47 48 49 50 51 52 53 54
55 56 57 58 59 60 61 62 63 64 65
66 67 68 69 70 71 72 73 74 75 76
77 78 79 80 81 82 83 84 85 86 87
88 89 90 91 92 93 94 95 96 97 98
99 100 101 102 103 104 105 106 107 108 109
110 111 112 113 114 115 116 117 118 119 120

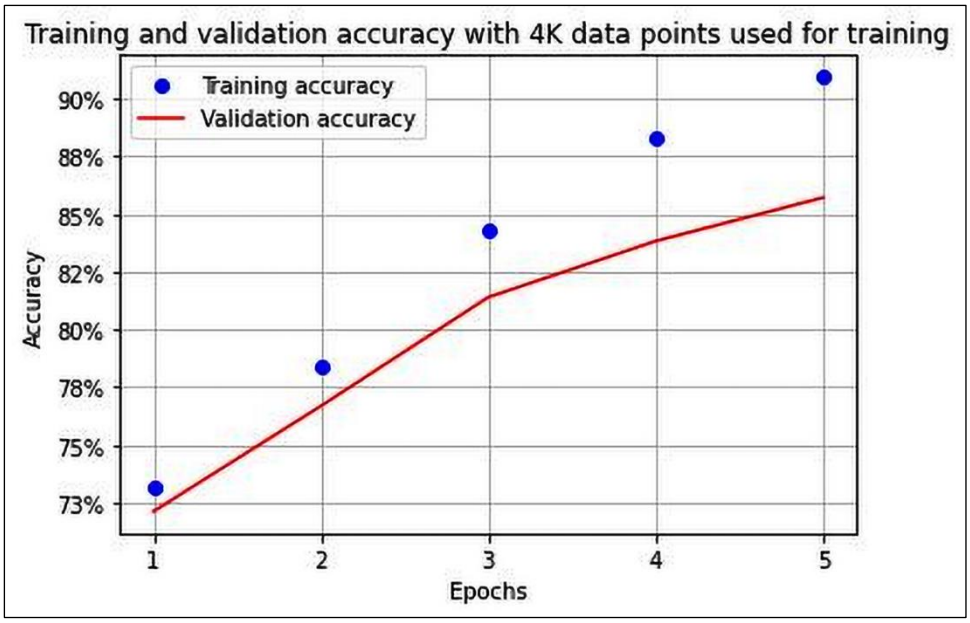

0
1
2
3
4
5
6
7
8
9
10
11
12
13
14

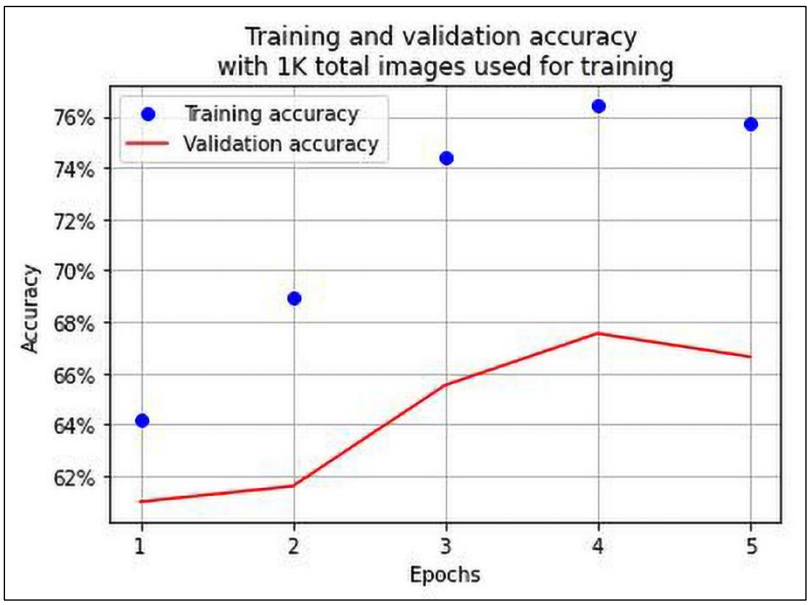

0
200
400
600
800
1000
1200
0
500
1000
1500
2000
2500
3000

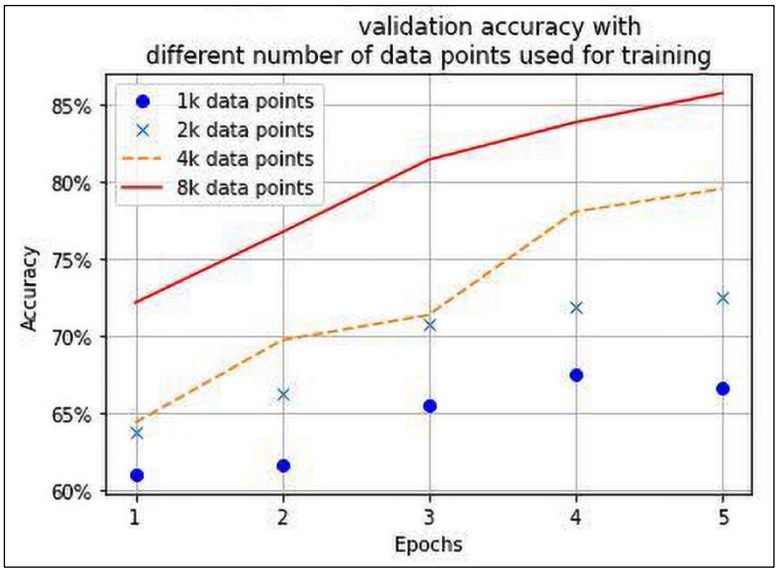
```
tensor([1.], device='cuda:0')
```



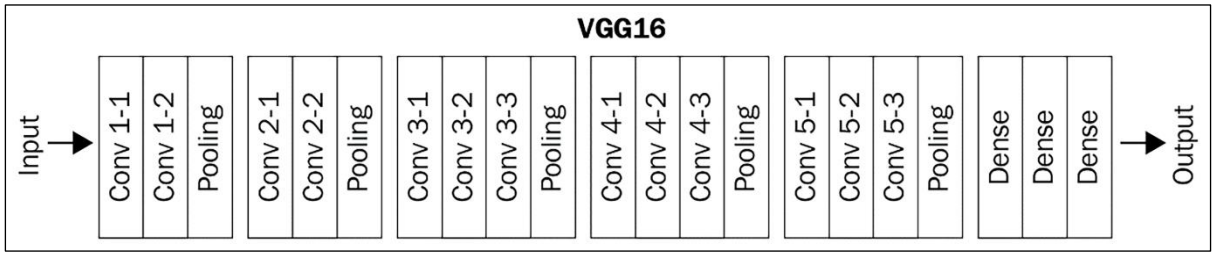
```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1         [-1, 64, 222, 222]           1,792
              ReLU-2         [-1, 64, 222, 222]               0
       BatchNorm2d-3         [-1, 64, 222, 222]             128
         MaxPool2d-4         [-1, 64, 111, 111]               0
            Conv2d-5        [-1, 512, 109, 109]         295,424
              ReLU-6        [-1, 512, 109, 109]               0
       BatchNorm2d-7        [-1, 512, 109, 109]           1,024
         MaxPool2d-8          [-1, 512, 54, 54]               0
            Conv2d-9          [-1, 512, 52, 52]       2,359,808
             ReLU-10          [-1, 512, 52, 52]               0
      BatchNorm2d-11          [-1, 512, 52, 52]           1,024
        MaxPool2d-12          [-1, 512, 26, 26]               0
           Conv2d-13          [-1, 512, 24, 24]       2,359,808
             ReLU-14          [-1, 512, 24, 24]               0
      BatchNorm2d-15          [-1, 512, 24, 24]           1,024
        MaxPool2d-16          [-1, 512, 12, 12]               0
           Conv2d-17          [-1, 512, 10, 10]       2,359,808
             ReLU-18          [-1, 512, 10, 10]               0
      BatchNorm2d-19          [-1, 512, 10, 10]           1,024
        MaxPool2d-20            [-1, 512, 5, 5]               0
           Conv2d-21            [-1, 512, 3, 3]       2,359,808
             ReLU-22            [-1, 512, 3, 3]               0
      BatchNorm2d-23            [-1, 512, 3, 3]           1,024
        MaxPool2d-24            [-1, 512, 1, 1]               0
          Flatten-25                  [-1, 512]               0
           Linear-26                    [-1, 1]             513
          Sigmoid-27                    [-1, 1]               0
================================================================
Total params: 9,742,209
Trainable params: 9,742,209
Non-trainable params: 0
```

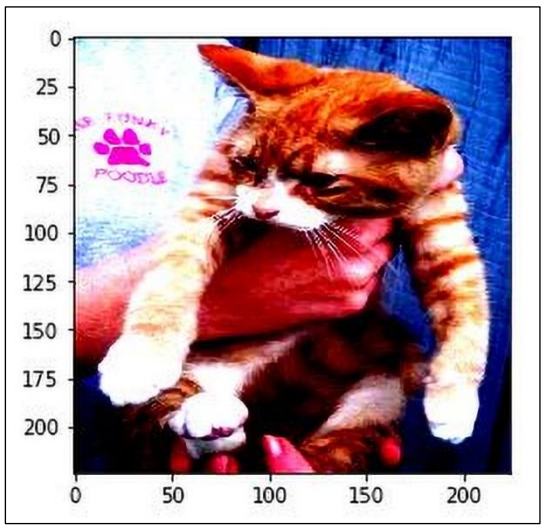

Training and validation accuracy with 4K data points used for training
Training accuracy
Validation accuracy
90%
88%
85%
82%
80%
78%
75%
73%
Accuracy
1
2
3
4
5
Epochs

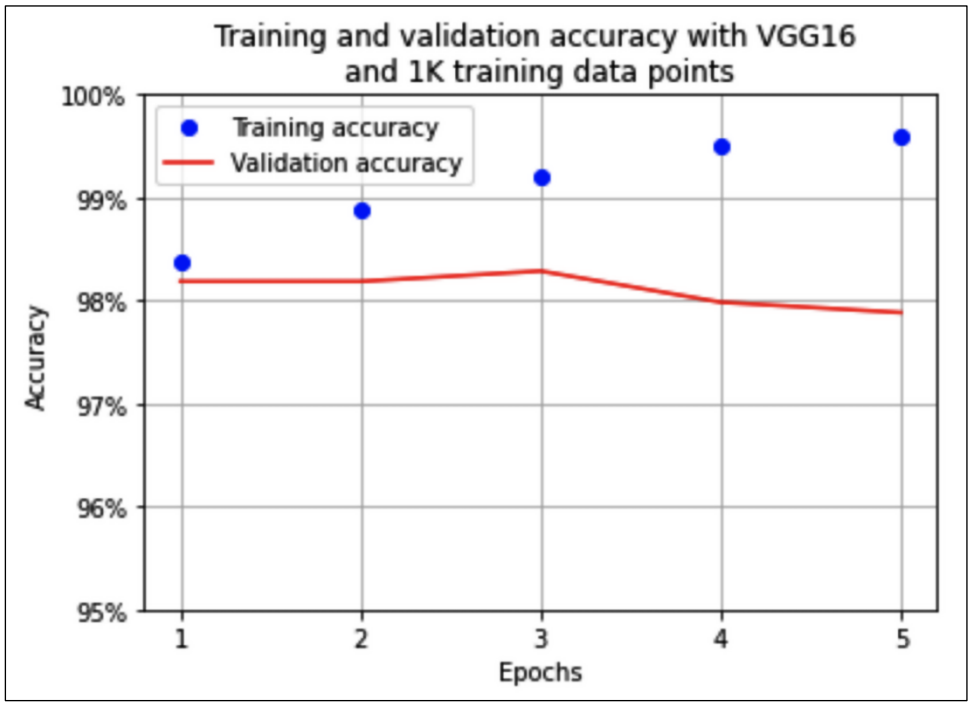

Training and validation accuracy
with 1K total images used for training
Training accuracy
Validation accuracy
76%
74%
72%
70%
68%
66%
64%
62%
Accuracy
1
2
3
4
5
Epochs

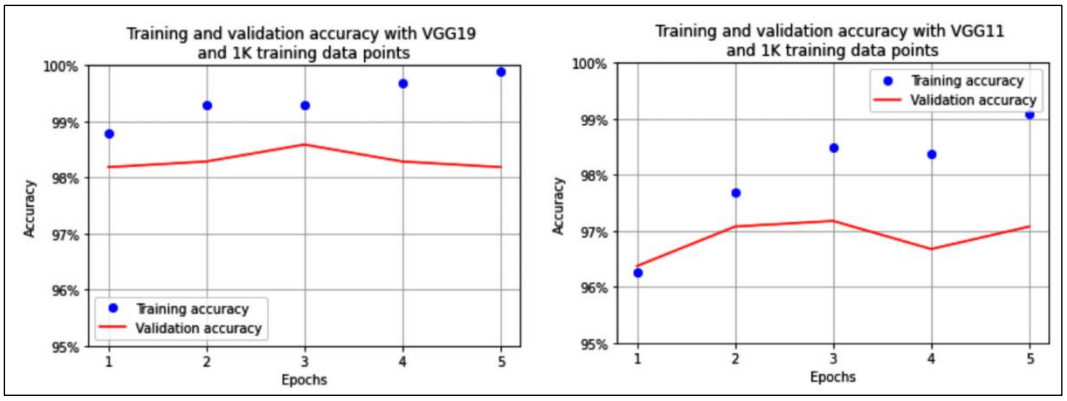

validation accuracy with
different number of data points used for training
1k data points
2k data points
4k data points
8k data points
85%
80%
75%
70%
65%
60%
Accuracy
1
2
3
4
5
Epochs

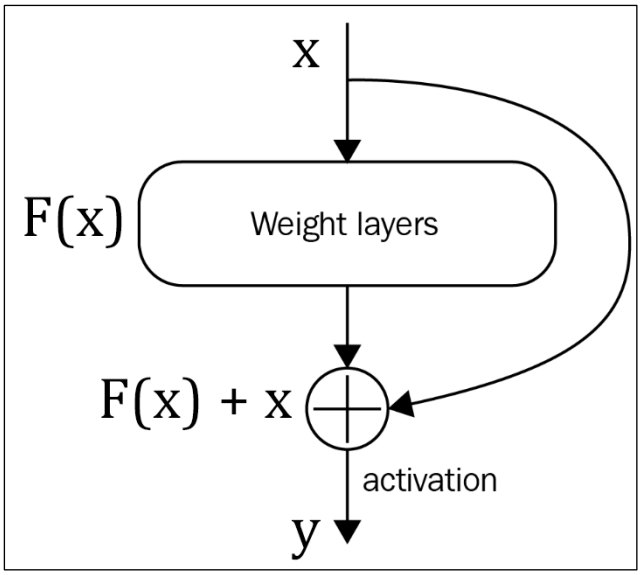
Chapter 5:


VGG16
Input
Conv 1-1
Conv 1-2
Pooling
Conv 2-1
Conv 2-2
Pooling
Conv 3-1
Conv 3-2
Conv 3-3
Pooling
Conv 4-1
Conv 4-2
Conv 4-3
Pooling
Conv 5-1
Conv 5-2
Conv 5-3
Pooling
Dense
Dense
Dense
Output


0
25
50
75
100
125
150
175
200
0
50
100
150
200


Training and validation accuracy with VGG16
and 1K training data points
Training accuracy
Validation accuracy
100%
99%
98%
97%
96%
95%
Accuracy
1
2
3
4
5
Epochs


Training and validation accuracy with VGG19
and 1K training data points
Accuracy
Epochs
Training accuracy
Validation accuracy
Training and validation accuracy with VGG11
and 1K training data points
Accuracy
Epochs
Training accuracy
Validation accuracy


x
F(x)
Weight layers
F(x) + x
activation
y

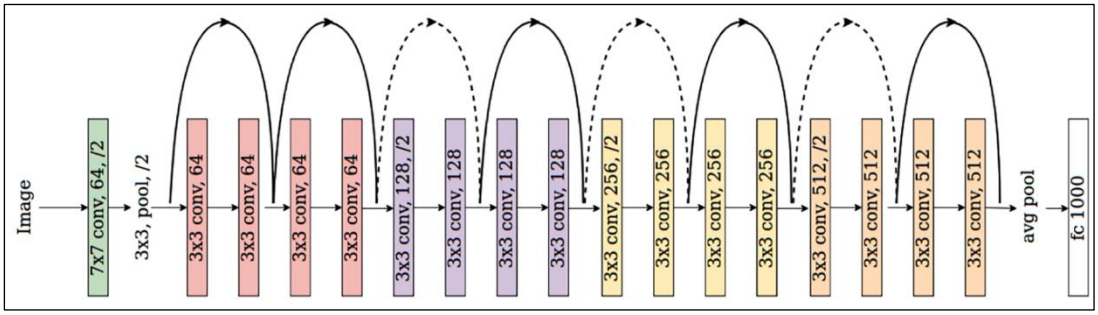
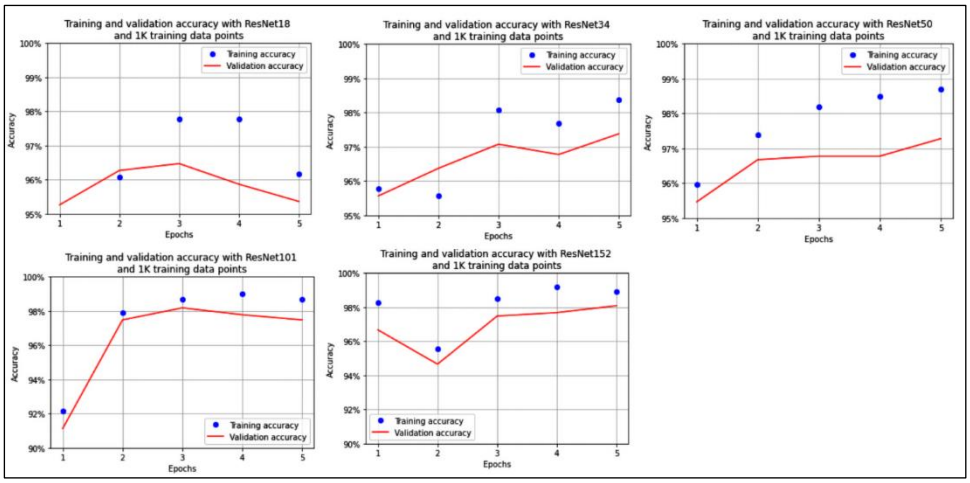
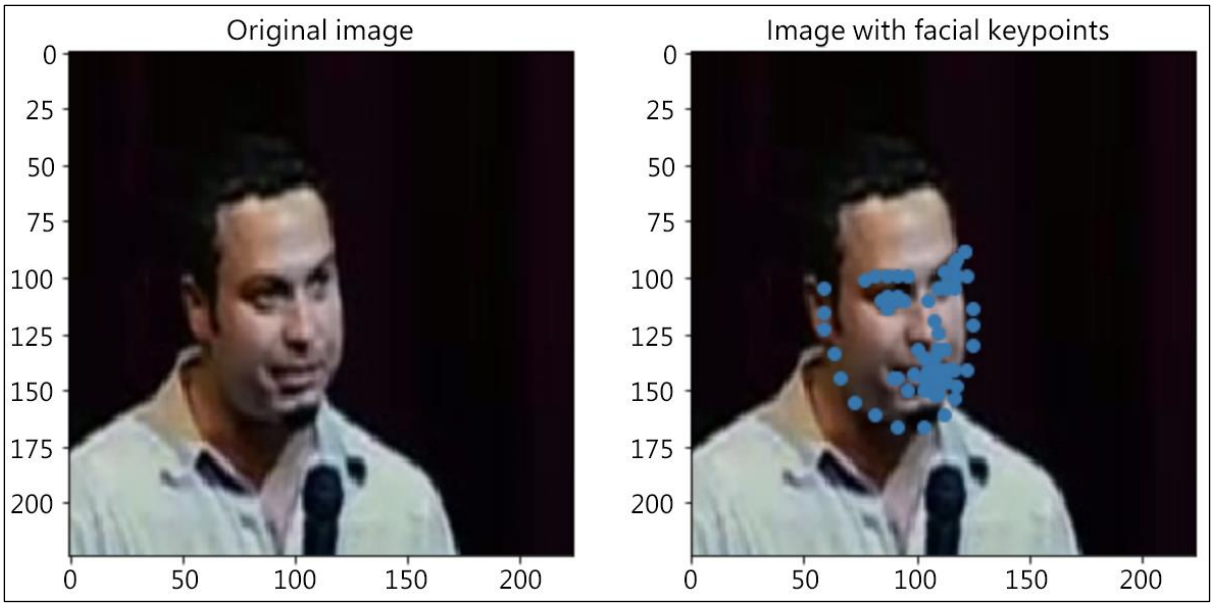


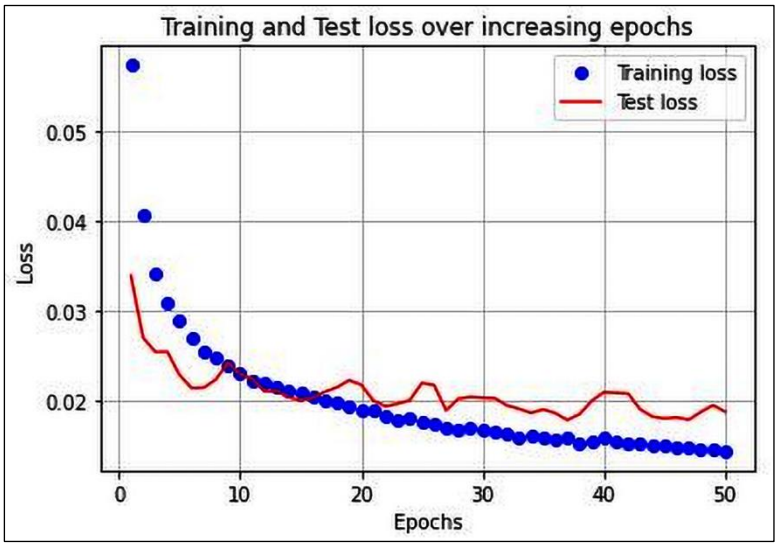
Image
7x7 conv, 64, /2
3x3, pool, /2
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 128, /2
3x3 conv, 128
3x3 conv, 128
3x3 conv, 128
3x3 conv, 256, /2
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
3x3 conv, 512, /2
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
avg pool
fc 1000

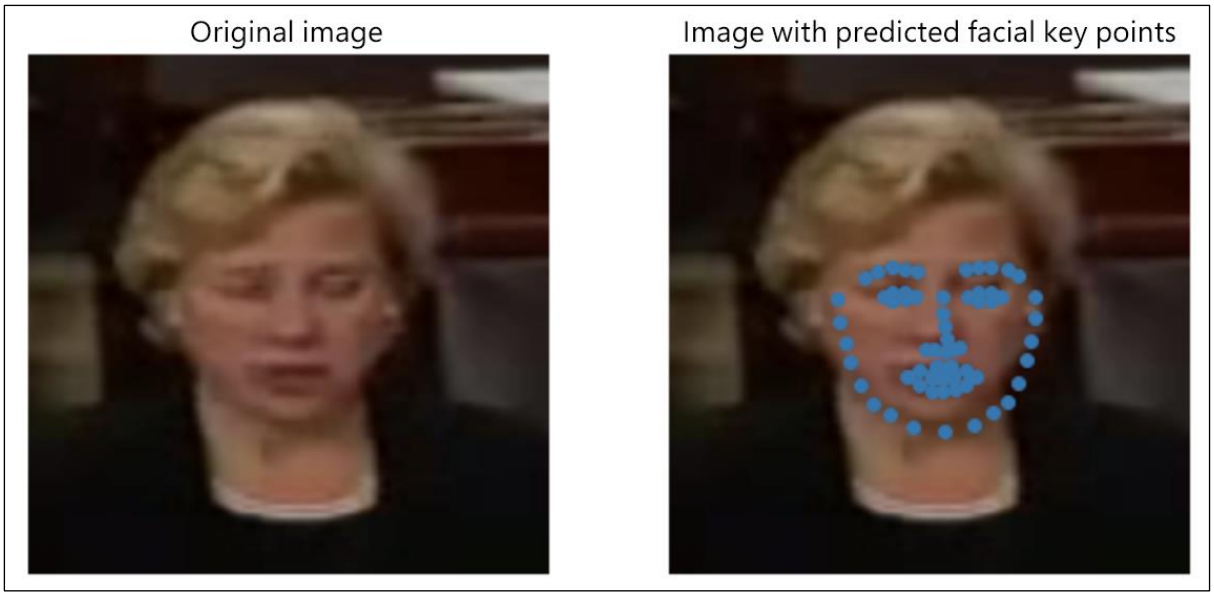

Training and validation accuracy with ResNet18
and 1K training data points
Training and validation accuracy with ResNet34
and 1K training data points
Training and validation accuracy with ResNet50
and 1K training data points
Training and validation accuracy with ResNet101
and 1K training data points
Training and validation accuracy with ResNet152
and 1K training data points
Accuracy
Epochs
Training accuracy
Validation accuracy

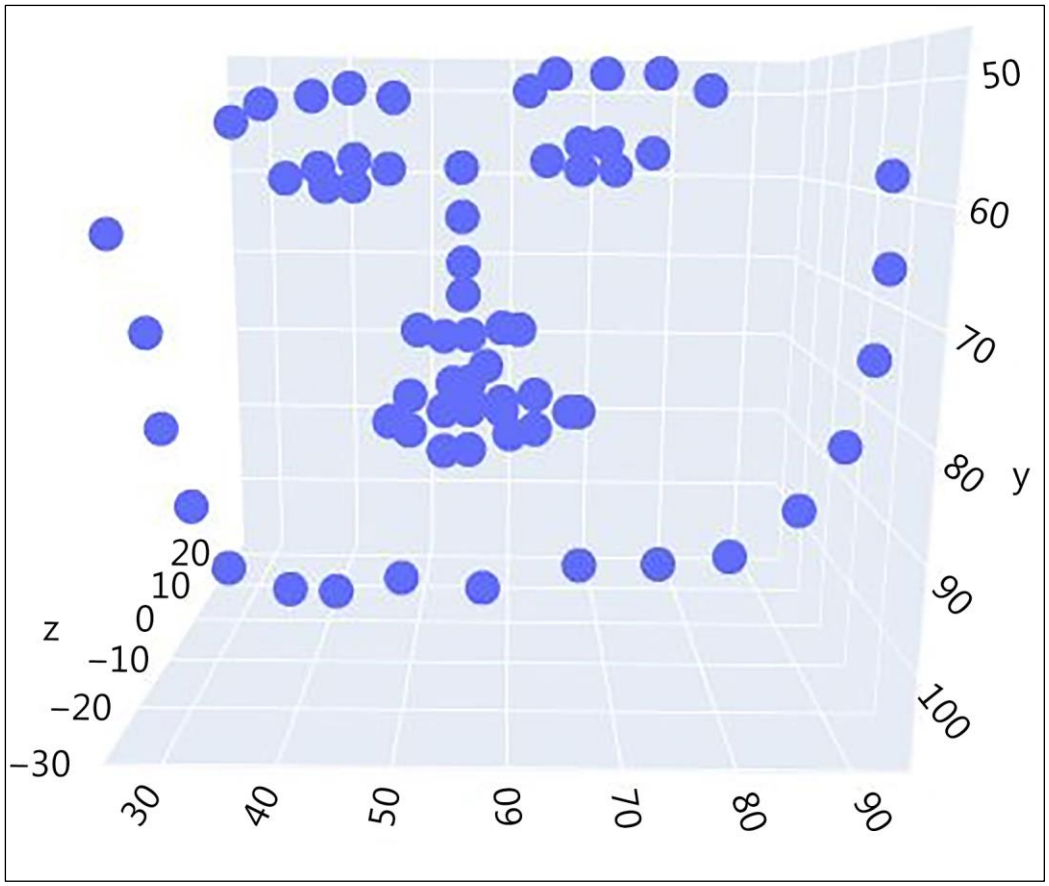


	Unnamed: 0	0	1	2	3	4	5	6
0	Luis_Fonsi_21.jpg	45.0	98.0	47.0	106.0	49.0	110.0	53.0
1	Lincoln_Chafee_52.jpg	41.0	83.0	43.0	91.0	45.0	100.0	47.0
2	Valerie_Harper_30.jpg	56.0	69.0	56.0	77.0	56.0	86.0	56.0
3	Angelo_Reyes_22.jpg	61.0	80.0	58.0	95.0	58.0	108.0	58.0
4	Kristen_Breitweiser_11.jpg	58.0	94.0	58.0	104.0	60.0	113.0	62.0

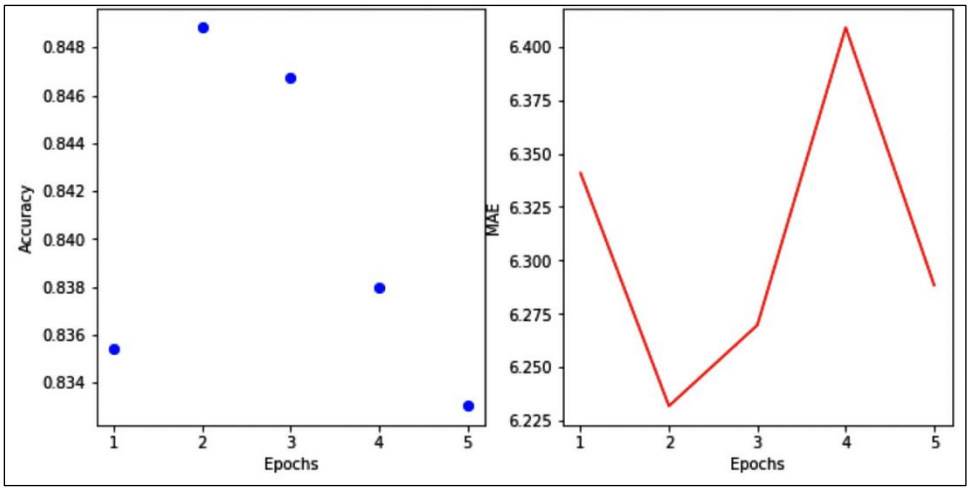
5 rows × 137 columns

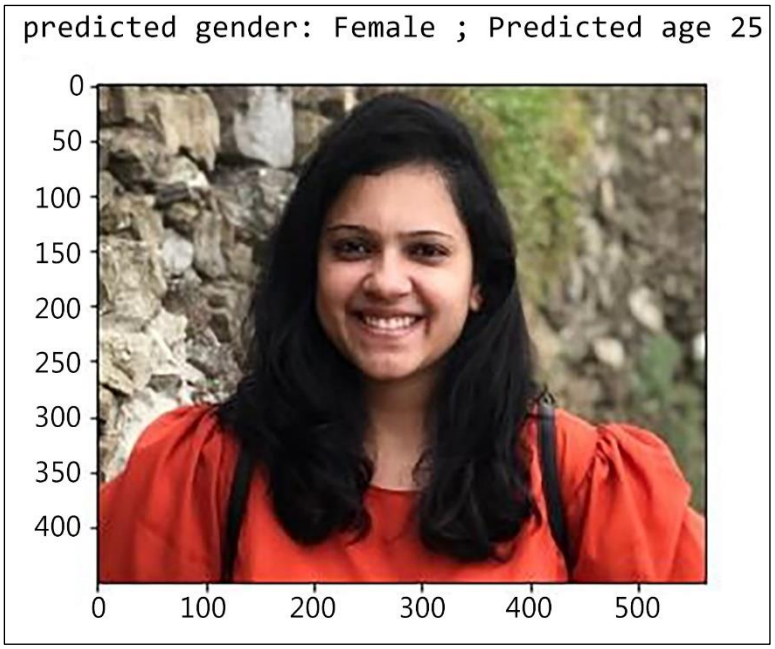



Original image
Image with predicted facial key points

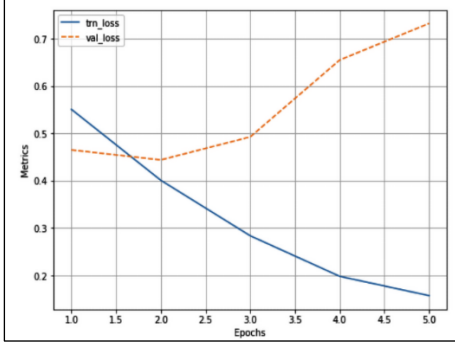


	file	age	gender	race	service_test
0	train/1.jpg	59	Male	East Asian	True
1	train/2.jpg	39	Female	Indian	False
2	train/3.jpg	11	Female	Black	False
3	train/4.jpg	26	Female	Indian	True
4	train/5.jpg	26	Female	Indian	True

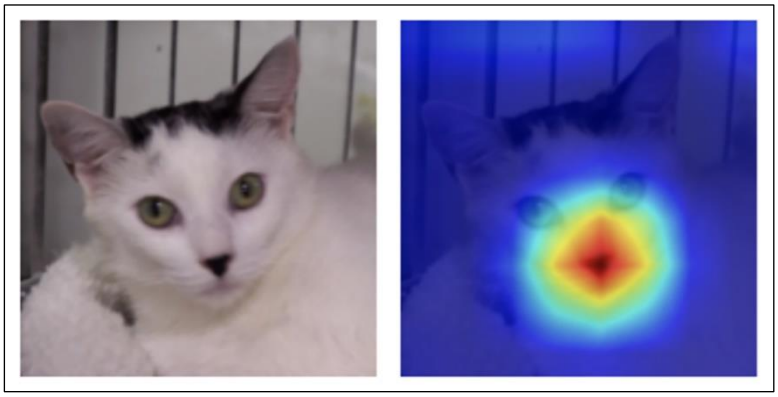


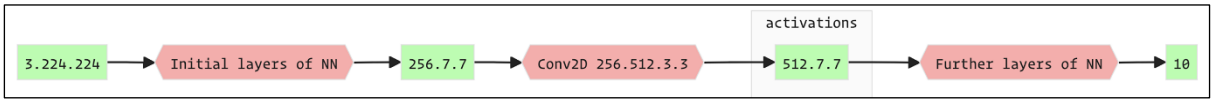


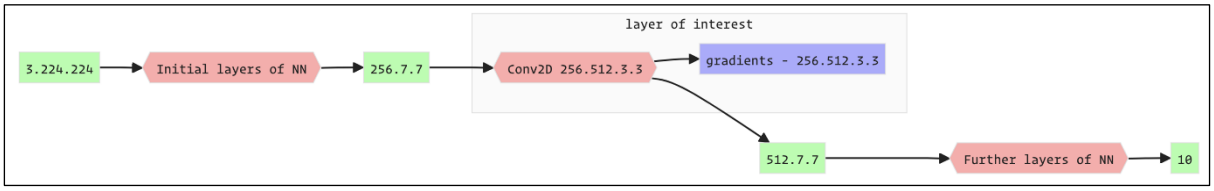

```
EPOCH: 1.000    trn_loss: 0.551 val_loss: 0.465 val_gender_acc: 0.834   val_age_mae: 6.238      (779.37s - 3117.47s remaining)
EPOCH: 2.000    trn_loss: 0.401 val_loss: 0.444 val_gender_acc: 0.847   val_age_mae: 6.229      (1555.70s - 2333.56s remaining)
EPOCH: 3.000    trn_loss: 0.284 val_loss: 0.493 val_gender_acc: 0.846   val_age_mae: 6.340      (2335.09s - 1556.73s remaining)
EPOCH: 4.000    trn_loss: 0.198 val_loss: 0.655 val_gender_acc: 0.842   val_age_mae: 6.339      (3110.83s - 777.71s remaining)
EPOCH: 4.994    val_loss: 0.375 val_gender_acc: 0.969   val_age_mae: 6.541      (3887.91s - 4.54s remaining)
  0%|          | 0/6 [00:00<?, ?it/s]/usr/local/lib/python3.6/dist-packages/numpy/core/fromnumeric.py:3335: RuntimeWarning: Mean
  out=out, **kwargs)
/usr/local/lib/python3.6/dist-packages/numpy/core/_methods.py:161: RuntimeWarning: invalid value encountered in double_scalars
  ret = ret.dtype.type(ret / rcount)
100%|██████████| 6/6 [00:00<00:00, 291.40it/s]EPOCH: 5.000      trn_loss: 0.157 val_loss: 0.733 val_gender_acc: 0.847   val_age_
```

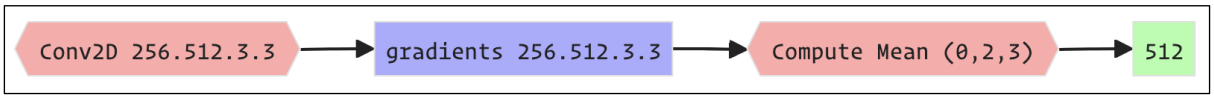


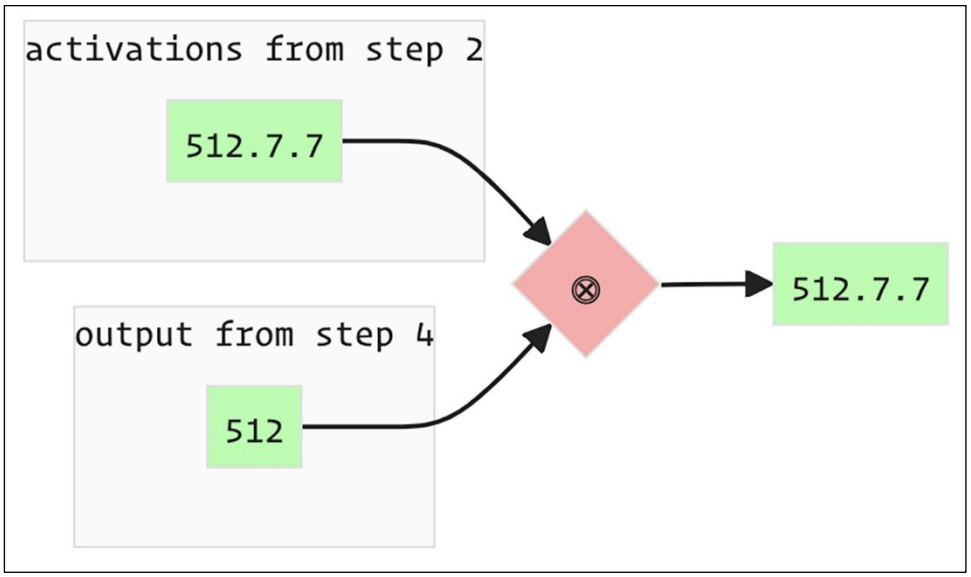
Chapter 6:

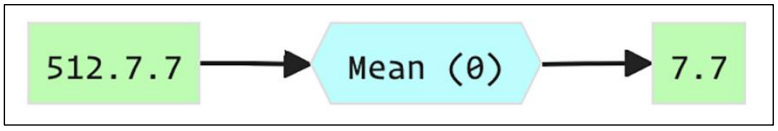


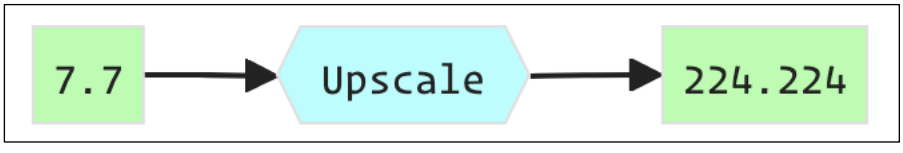

activations
3.224.224
Initial layers of NN
256.7.7
Conv2D 256.512.3.3
512.7.7
Further layers of NN
10

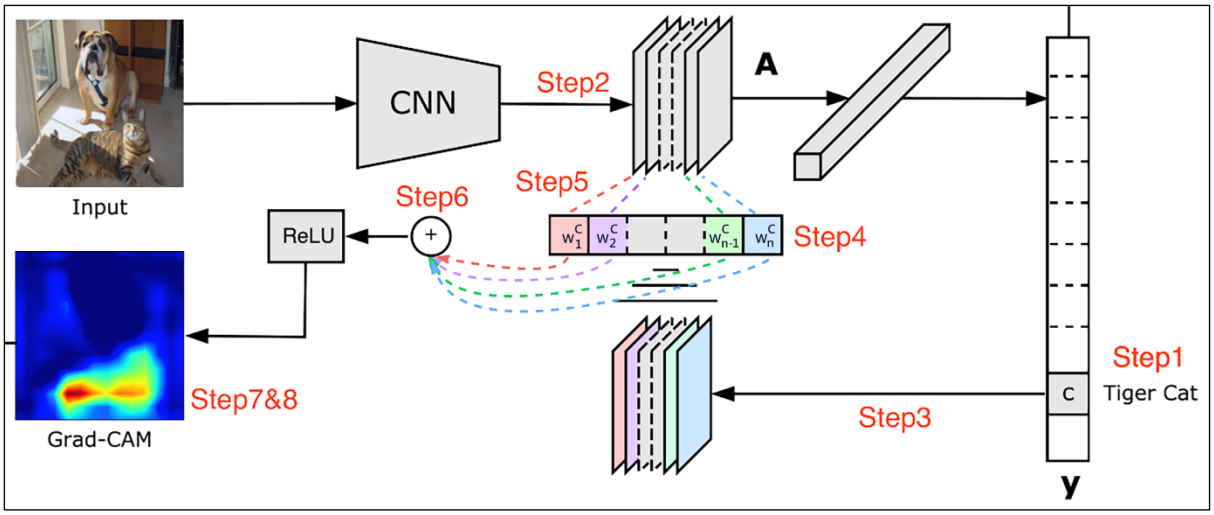

layer of interest
3.224.224
Initial layers of NN
256.7.7
Conv2D 256.512.3.3
gradients - 256.512.3.3
512.7.7
Further layers of NN
10

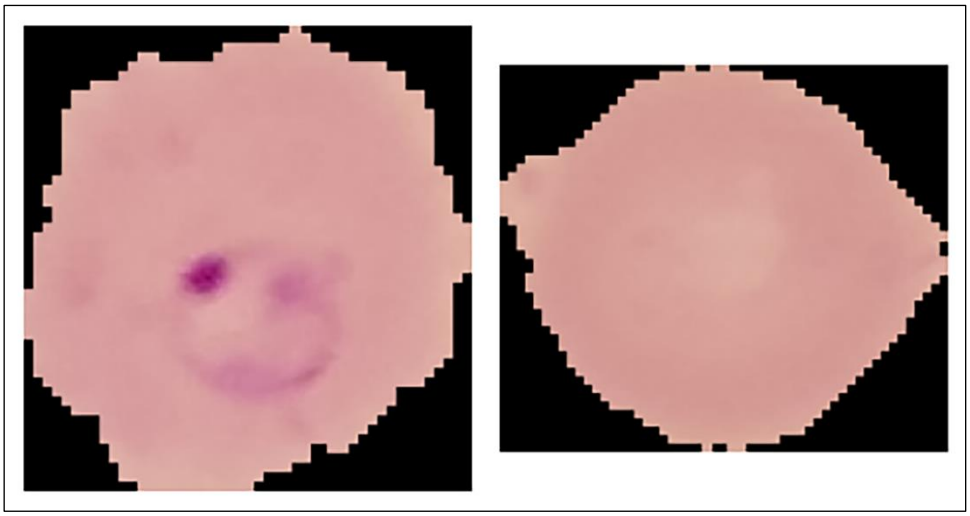

Conv2D 256.512.3.3
gradients 256.512.3.3
Compute Mean (0,2,3)
512

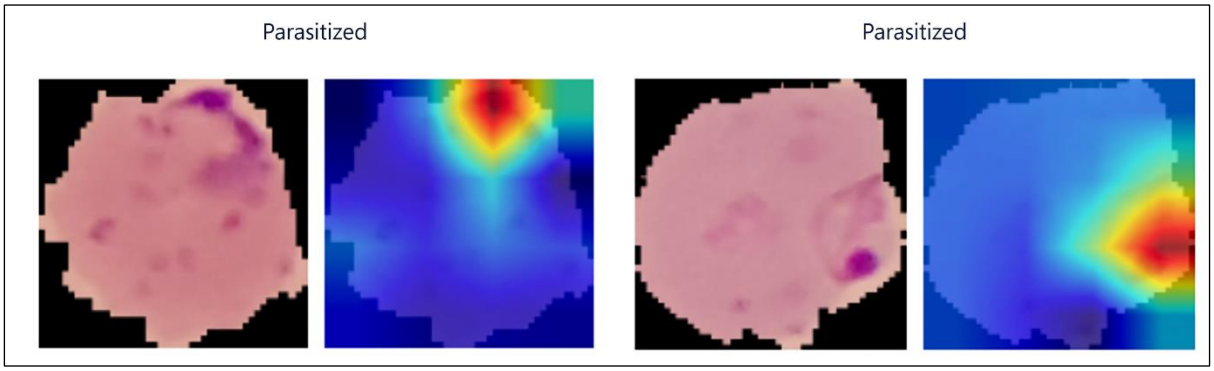

activations from step 2
512.7.7
output from step 4
512
512.7.7


512.7.7
Mean (0)
7.7


7.7
Upscale
224.224


Input
CNN
Step2
A
Step1
Tiger Cat
C
y
Step3
Step4
Step5
Step6
ReLU
Step7&8
Grad-CAM

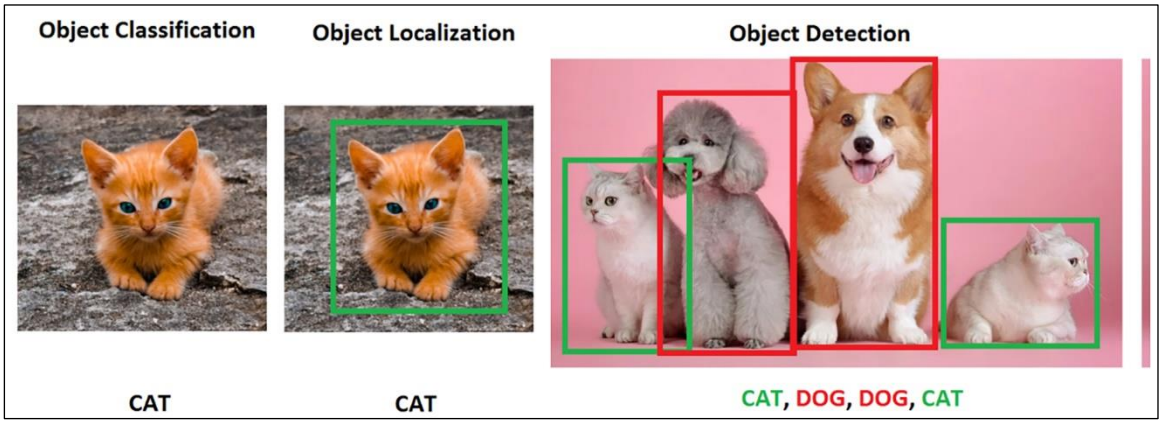


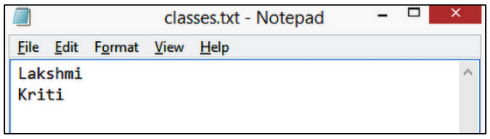

Parasitized
Parasitized

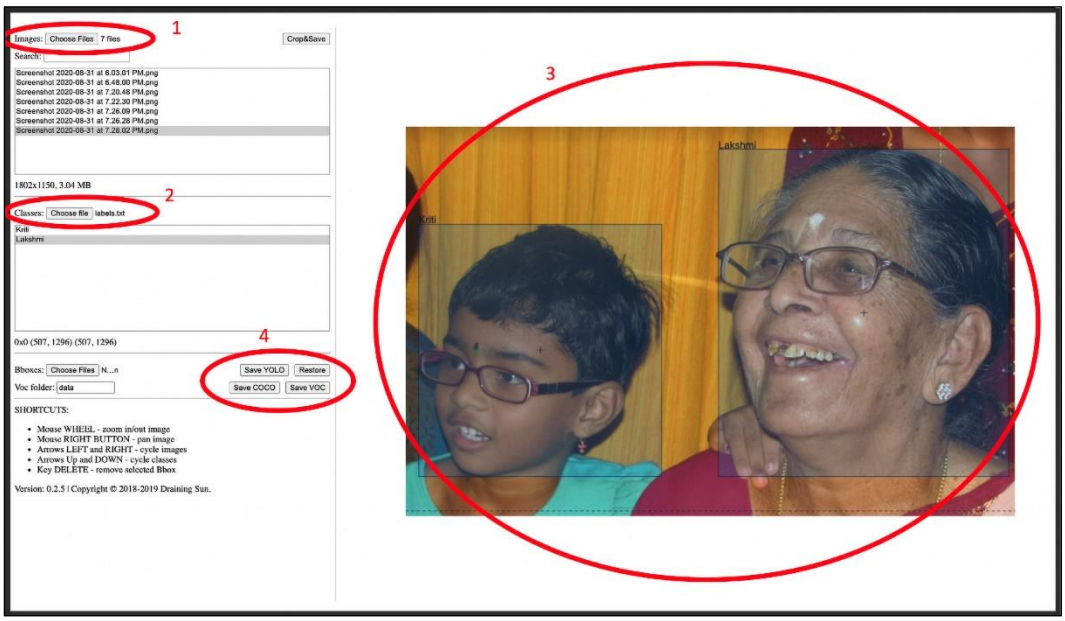
Augment	*Batch-Norm*	*Train Accuracy*	*Validation Accuracy*
No	No	95.9	94.5
No	Yes	99.3	97.7
Yes	Yes	97.7	97.6

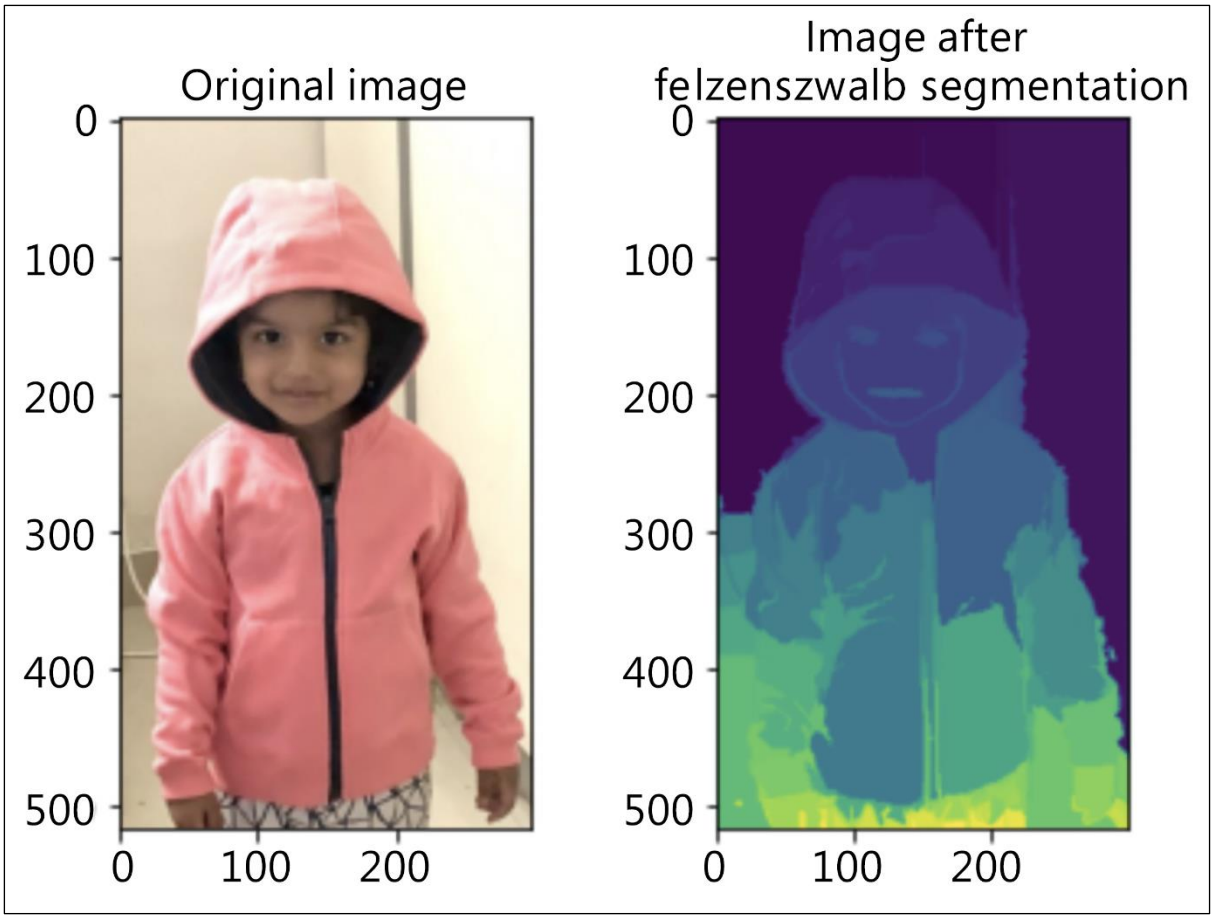
	Predicted	
Actual	0	1
0	TN	FP
1	FN	TP

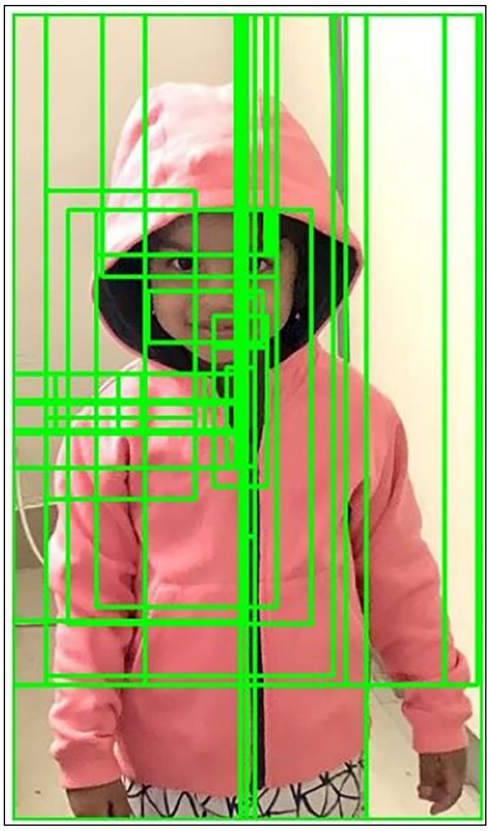
Chapter 7:

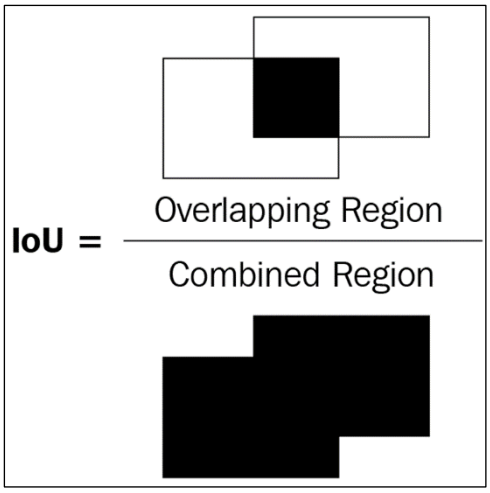

Object Classification
Object Localization
Object Detection
CAT
CAT
CAT, DOG, DOG, CAT

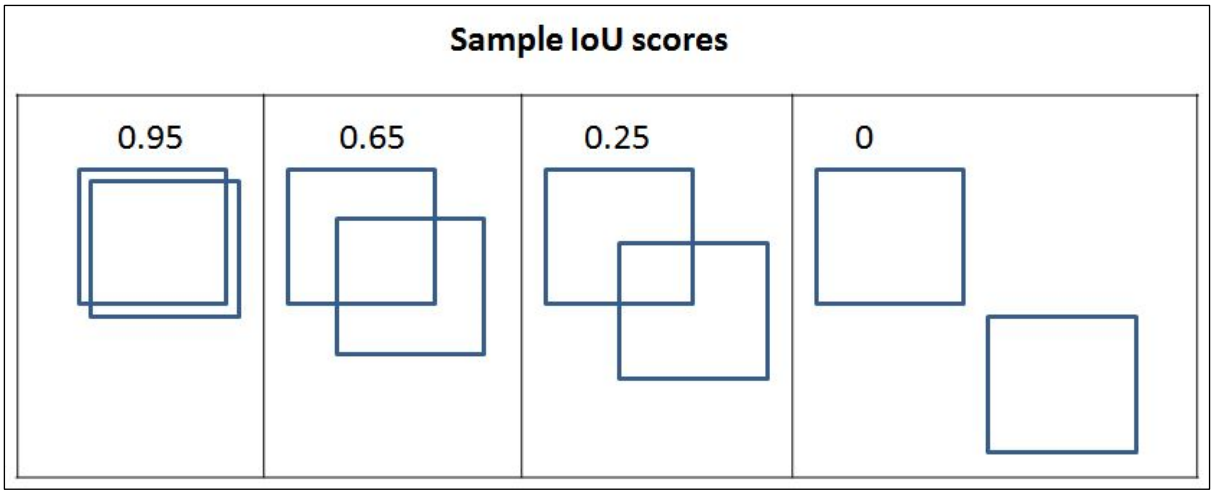

classes.txt - Notepad
File Edit Format View Help
Lakshmi
Kriti

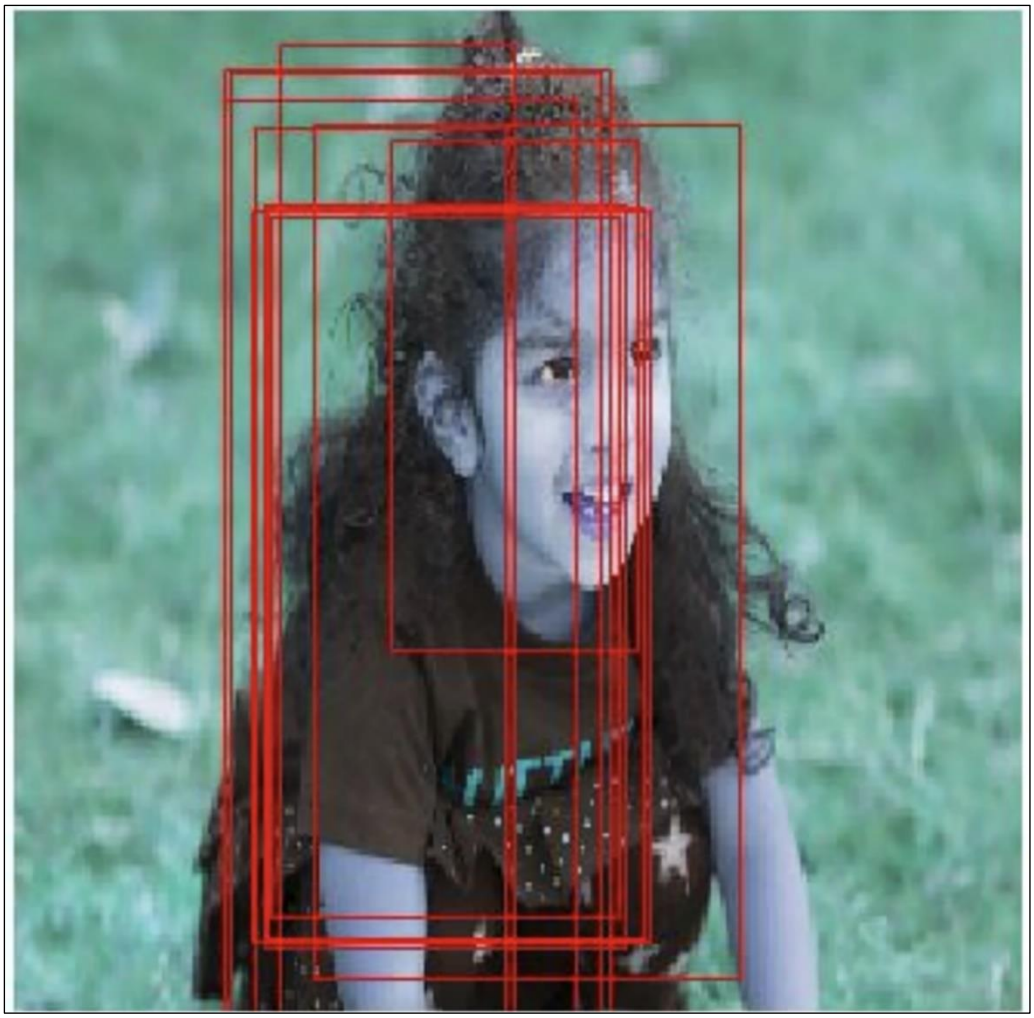

1
2
3
4
Images: Choose Files 7 files
Crop&Save
Search:
Screenshot 2020-08-31 at 6.03.01 PM.png
Screenshot 2020-08-31 at 6.48.00 PM.png
Screenshot 2020-08-31 at 7.20.48 PM.png
Screenshot 2020-08-31 at 7.22.30 PM.png
Screenshot 2020-08-31 at 7.26.09 PM.png
Screenshot 2020-08-31 at 7.26.28 PM.png
Screenshot 2020-08-31 at 7.26.02 PM.png
1802x1150, 3.04 MB
Classes: Choose file labels.txt
Kriti
Lakshmi
0x0 (507, 1296) (507, 1296)
Bboxes: Choose Files N...n
Save YOLO
Restore
Voc folder: data
Save COCO
Save VOC
SHORTCUTS:
Mouse WHEEL - zoom in/out image
Mouse RIGHT BUTTON - pan image
Arrows LEFT and RIGHT - cycle images
Arrows Up and DOWN - cycle classes
Key DELETE - remove selected Bbox
Version: 0.2.5 | Copyright © 2018-2019 Draining Sun.
Lakshmi
Kriti

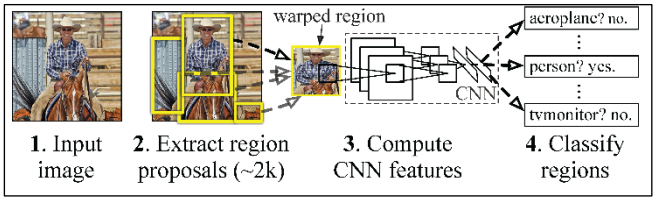

Original image
Image after
felzenszwalb segmentation
0
100
200
300
400
500
0
100
200

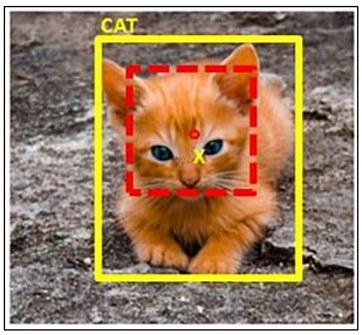



IoU =
Overlapping Region
Combined Region

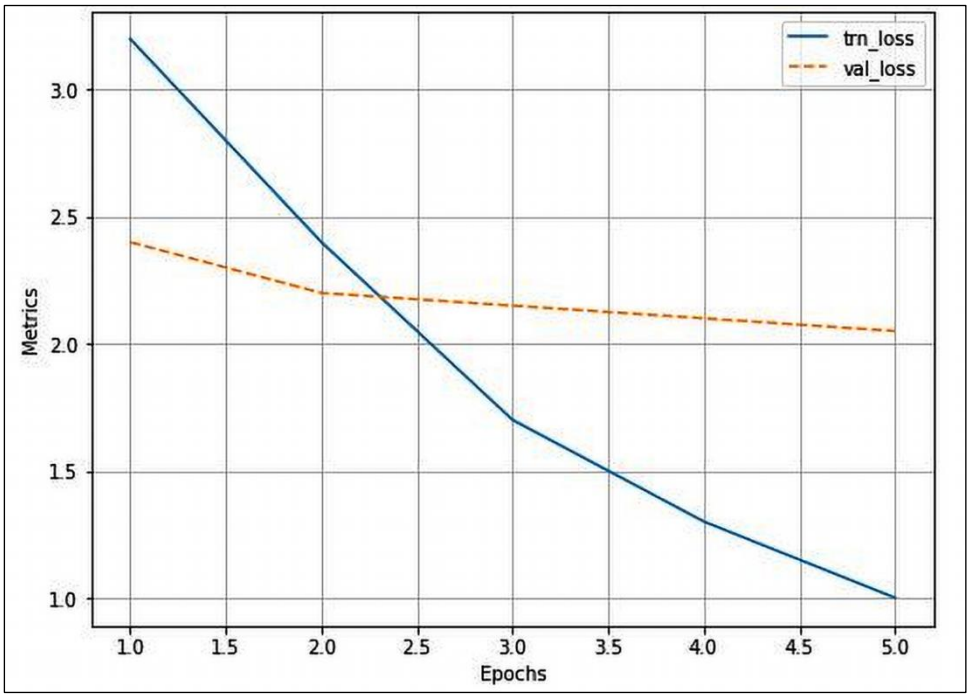

Sample IoU scores
0.95
0.65
0.25
0



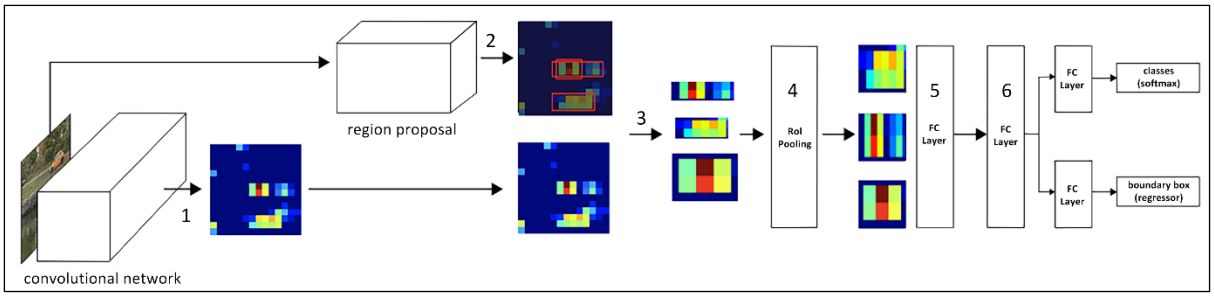


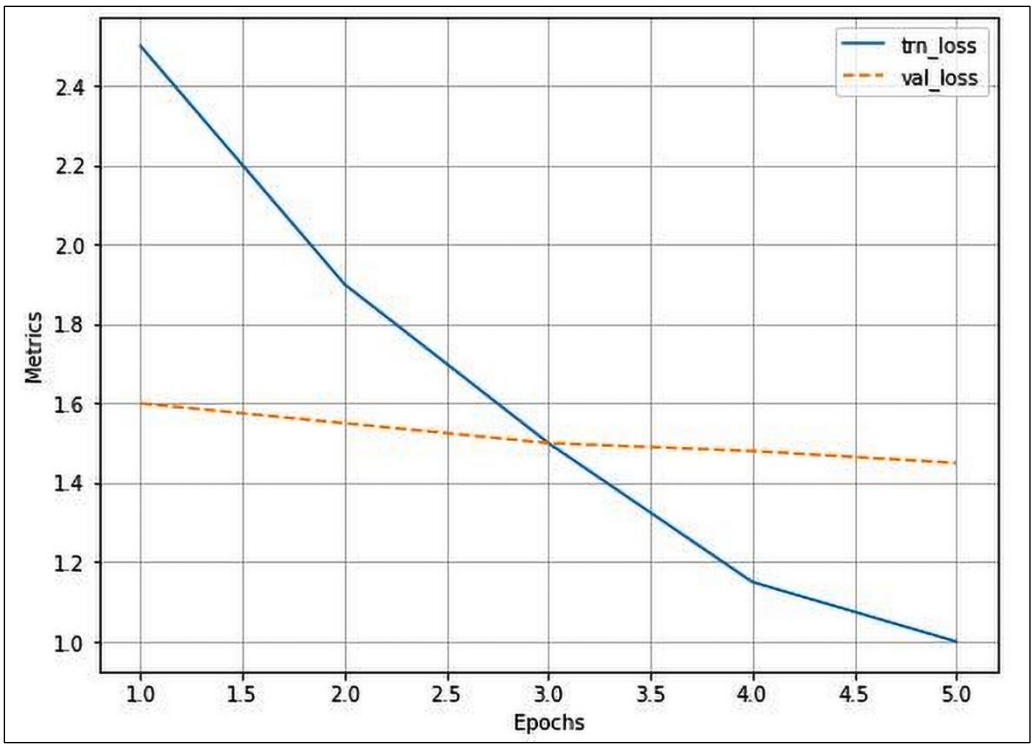


	ImageID	Source	LabelName	Confidence	XMin	XMax	YMin	YMax	IsOccluded	IsTruncated	IsGroupOf	IsDepiction	IsInside
20	002f8241bd829022	xclick	Bus	1	0.257812	0.515625	0.485417	0.891667	1	0	0	0	0
21	002f8241bd829022	xclick	Bus	1	0.535937	0.907813	0.347917	0.997917	1	1	0	0	0
191	013b99371484d3d5	xclick	Bus	1	0.154688	0.920312	0.102083	0.872917	0	0	0	0	0
322	01f8886b50a031a1	xclick	Truck	1	0.012821	0.987179	0.000000	0.969512	0	0	0	0	0
405	02717d30304f4849	xclick	Bus	1	0.106250	0.926562	0.266667	0.635417	0	0	0	0	0



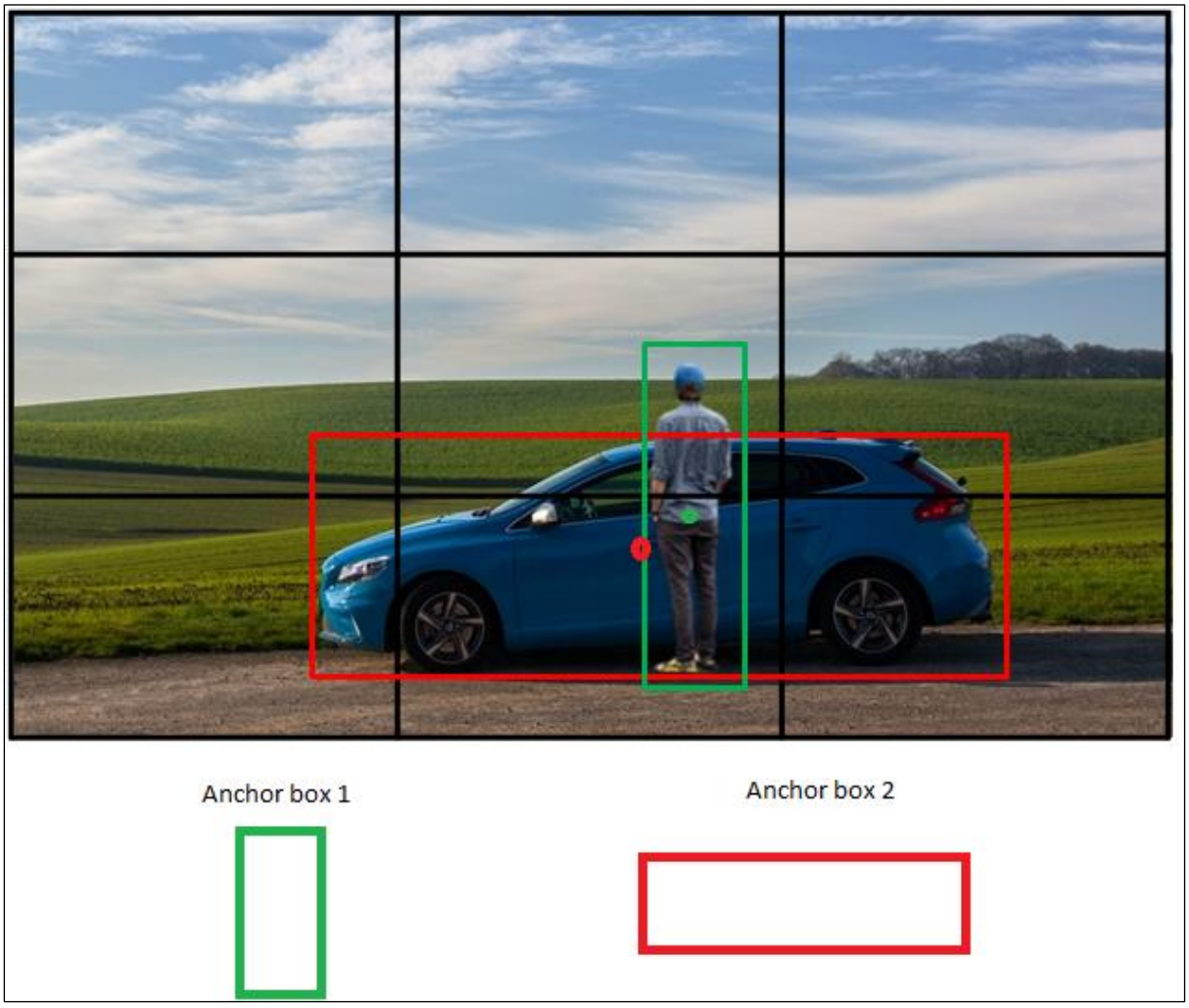
((16, 60, 217, 133), 'Bus', 0.974549)

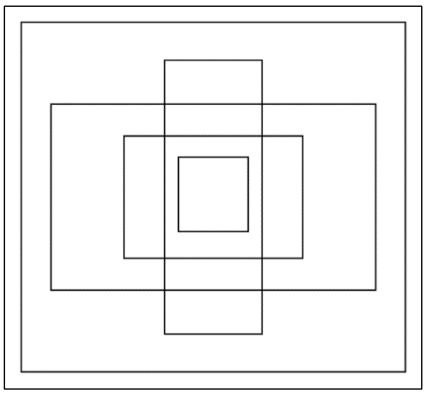

region proposal
convolutional network
1
2
3
4
RoI Pooling
5
FC Layer
6
FC Layer
FC Layer
FC Layer
classes (softmax)
boundary box (regressor)

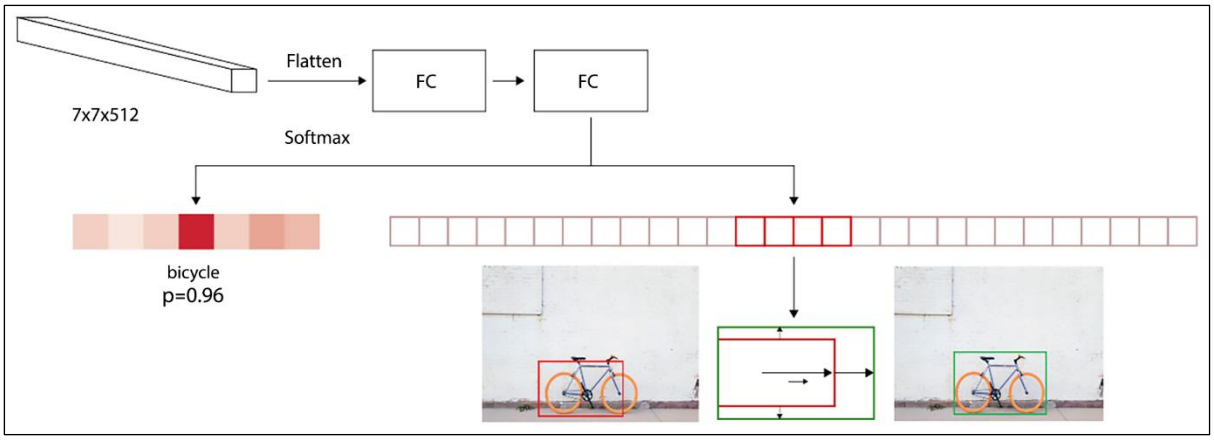

trn_loss
val_loss
Metrics
Epochs
2.4
2.2
2.0
1.8
1.6
1.4
1.2
1.0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0

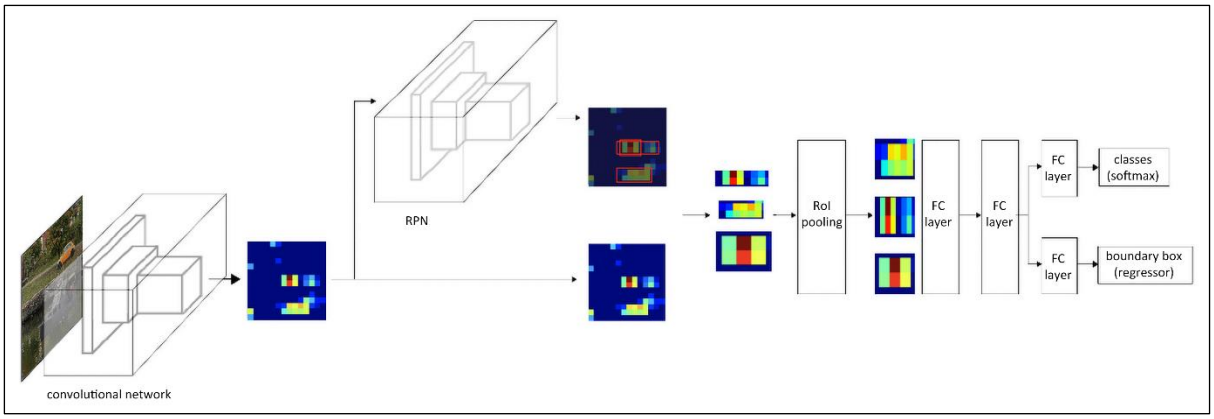

Truck
Truck

Chapter 8:







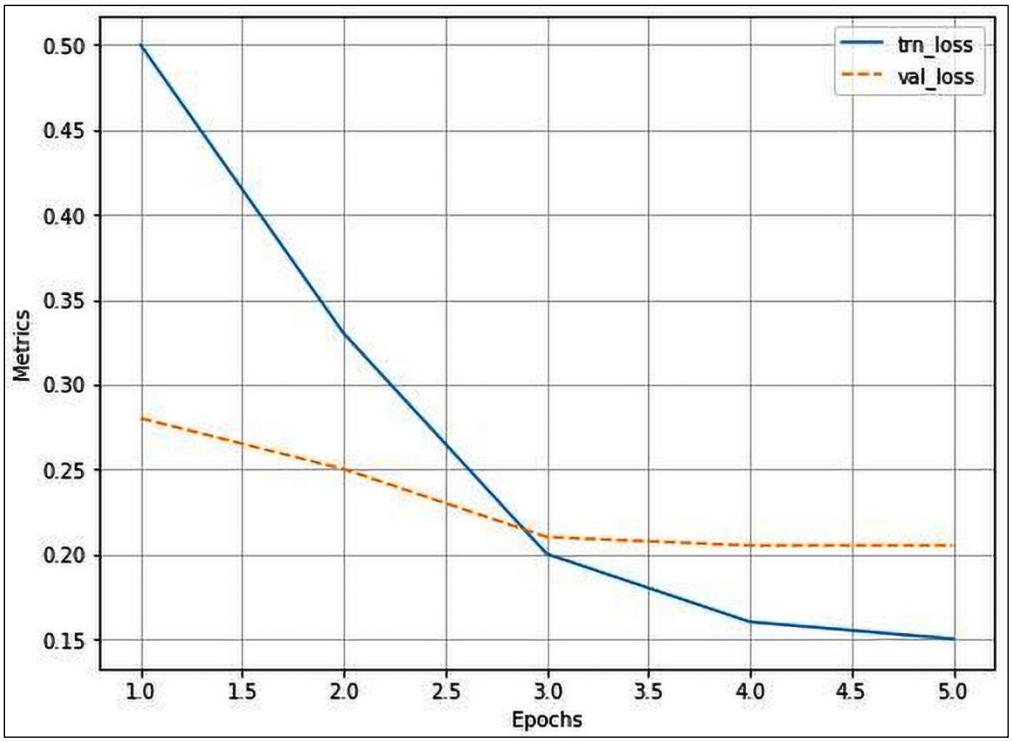


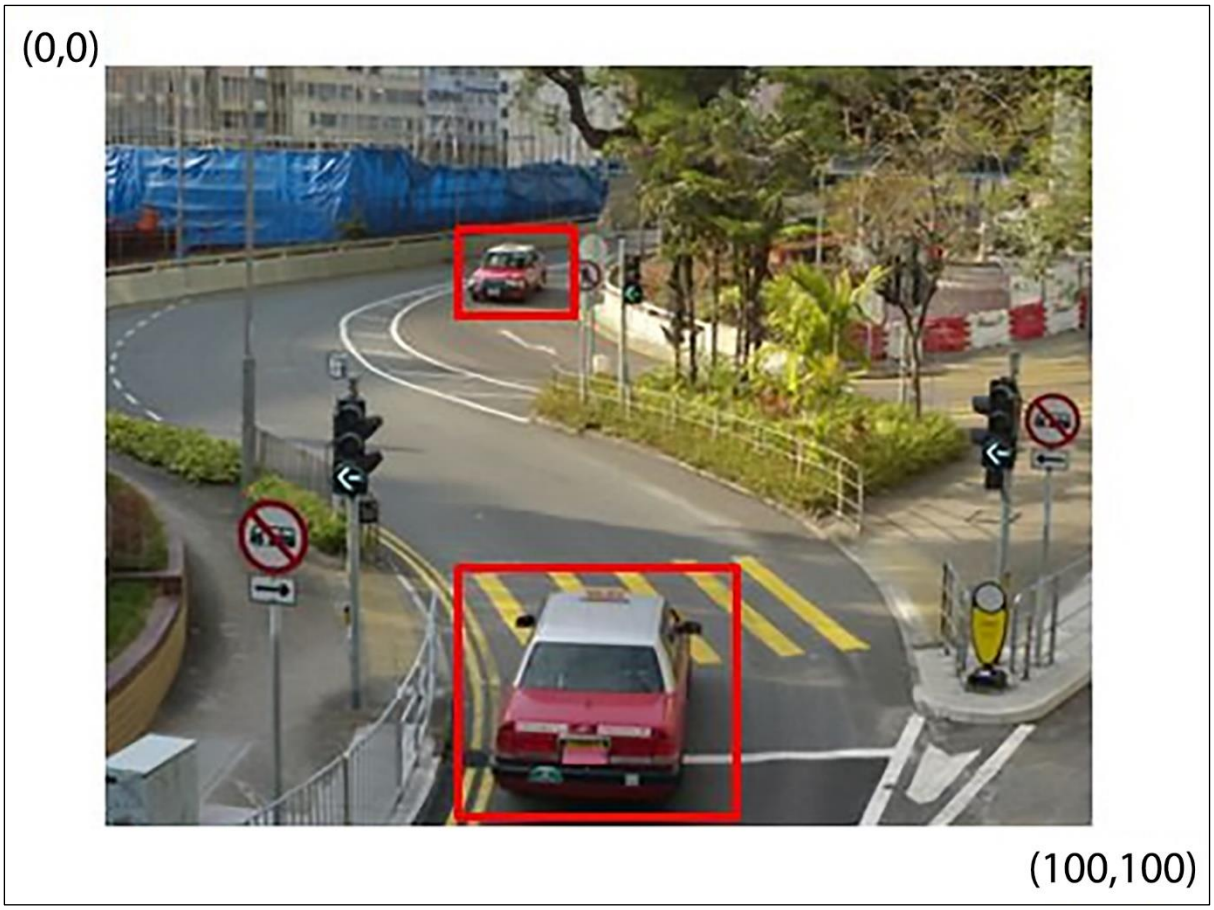

```
==========================================================================
Layer (type:depth-idx)                             Param #
==========================================================================
├─GeneralizedRCNNTransform: 1-1                    --
├─BackboneWithFPN: 1-2                             (26,799,296)
├─RegionProposalNetwork: 1-3                       593,935
├─RoIHeads: 1-4                                    13,905,930
==========================================================================
Total params: 41,299,161
Trainable params: 14,499,865
Non-trainable params: 26,799,296
==========================================================================
```

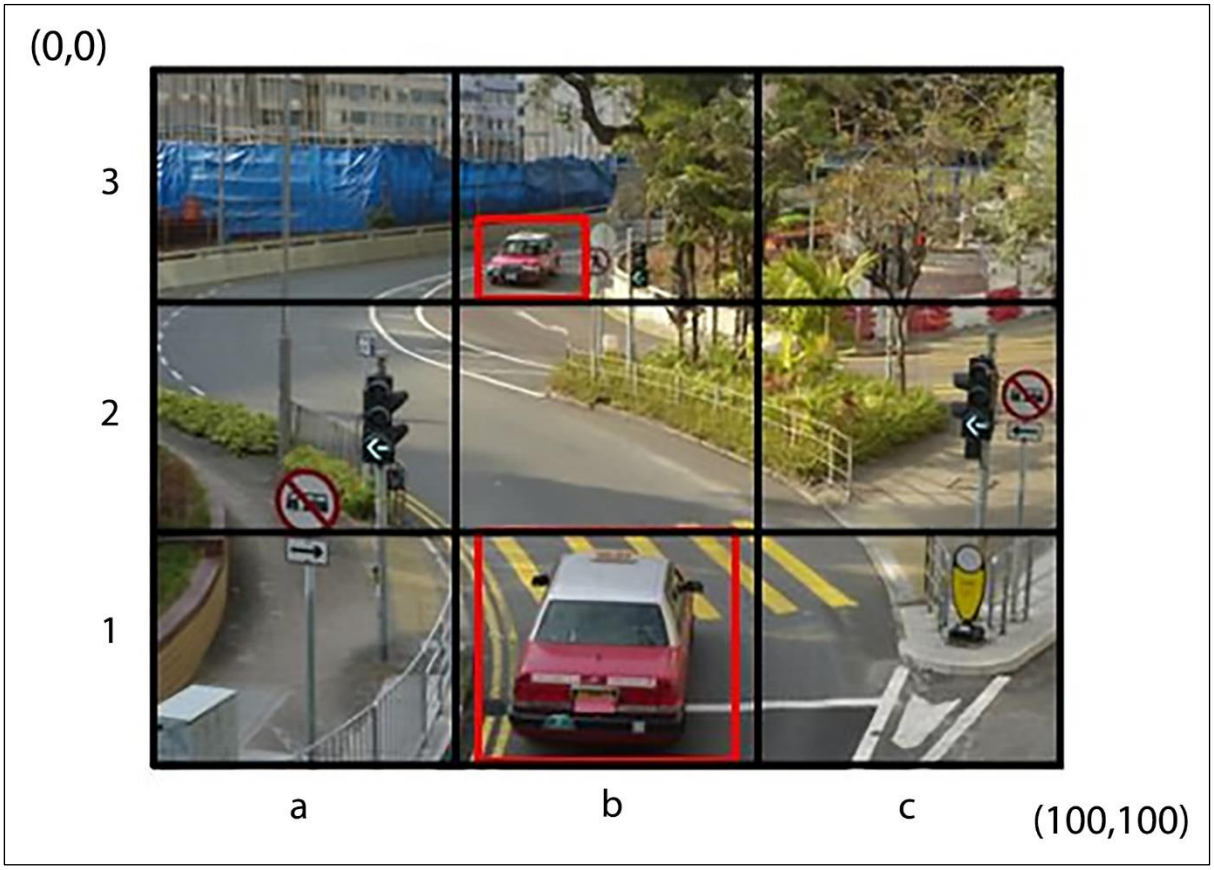
```
(transform): GeneralizedRCNNTransform(
    Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
    Resize(min_size=(800,), max_size=1333, mode='bilinear')
)
```

```
(rpn): RegionProposalNetwork(
  (anchor_generator): AnchorGenerator()
  (head): RPNHead(
    (conv): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (cls_logits): Conv2d(256, 3, kernel_size=(1, 1), stride=(1, 1))
    (bbox_pred): Conv2d(256, 12, kernel_size=(1, 1), stride=(1, 1))
  )
)
```

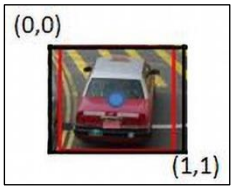
```
(roi_heads): RoIHeads(
  (box_roi_pool): MultiScaleRoIAlign()
  (box_head): TwoMLPHead(
    (fc6): Linear(in_features=12544, out_features=1024, bias=True)
    (fc7): Linear(in_features=1024, out_features=1024, bias=True)
  )
  (box_predictor): FastRCNNPredictor(
    (cls_score): Linear(in_features=1024, out_features=2, bias=True)
    (bbox_pred): Linear(in_features=1024, out_features=8, bias=True)
  )
)
```

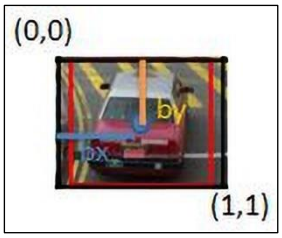



(0,0)
(100,100)


(0,0)
3
2
1
a
b
c
(100,100)

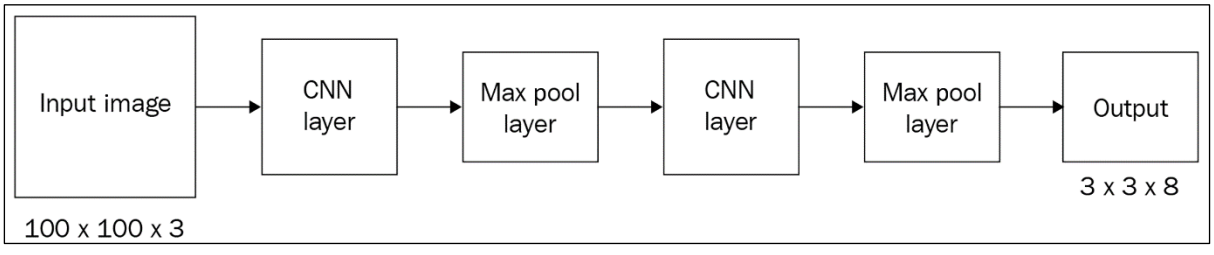
y=	pc
	bx
	by
	bw
	bh
	c1
	c2
	c3

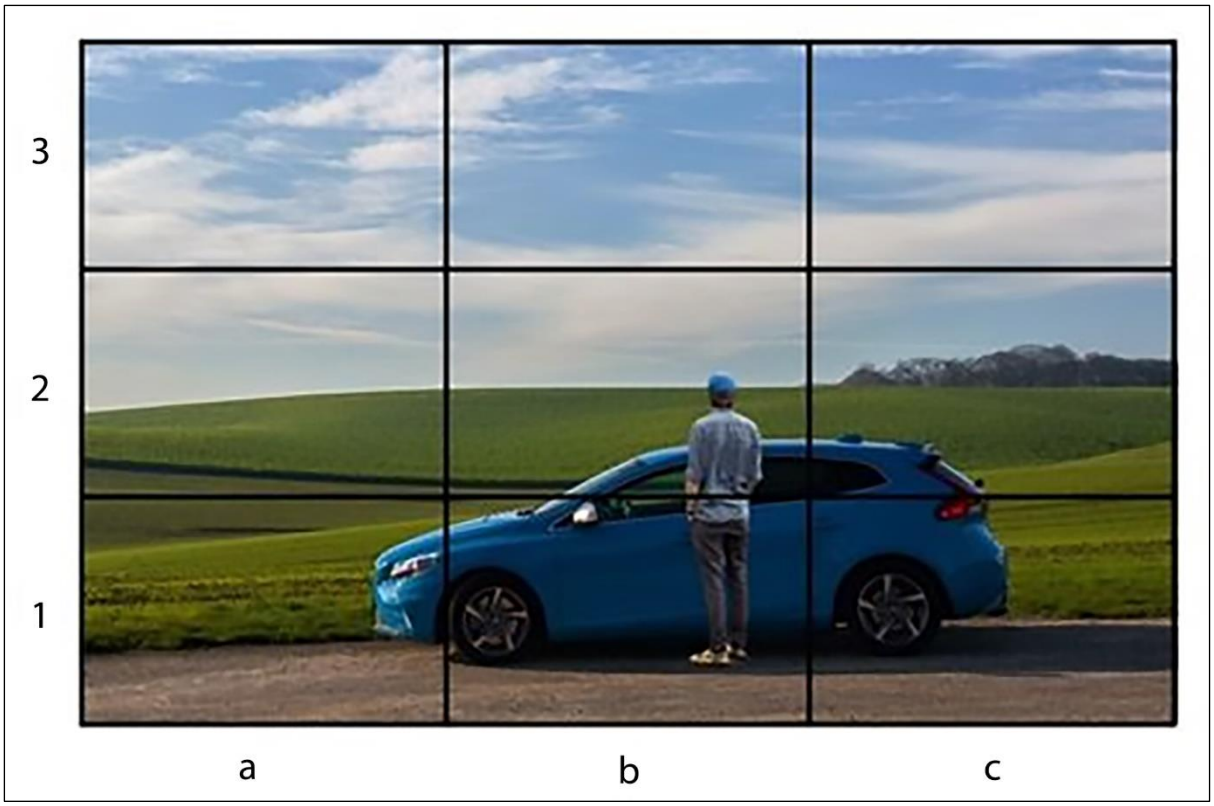


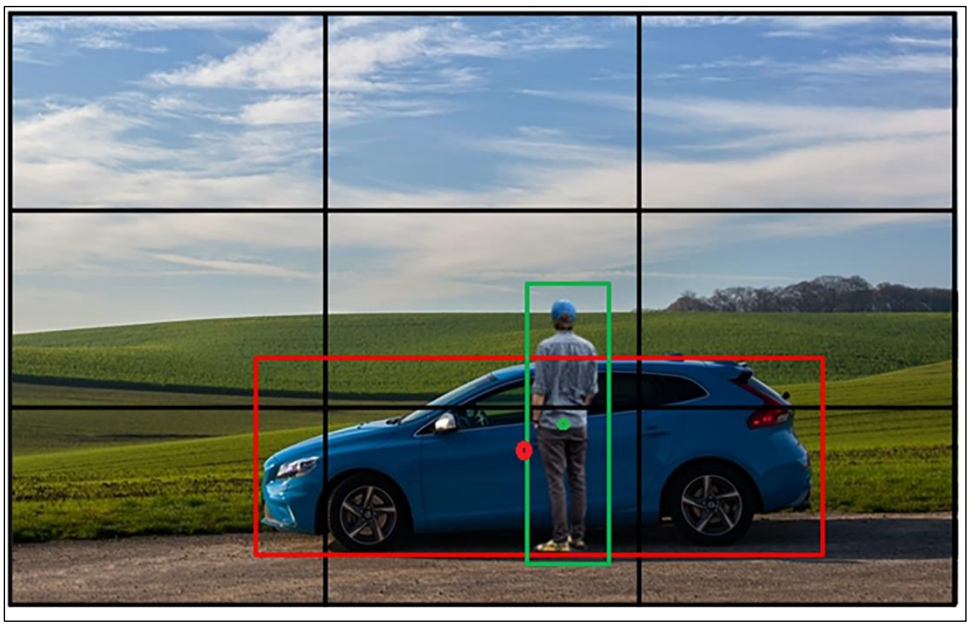


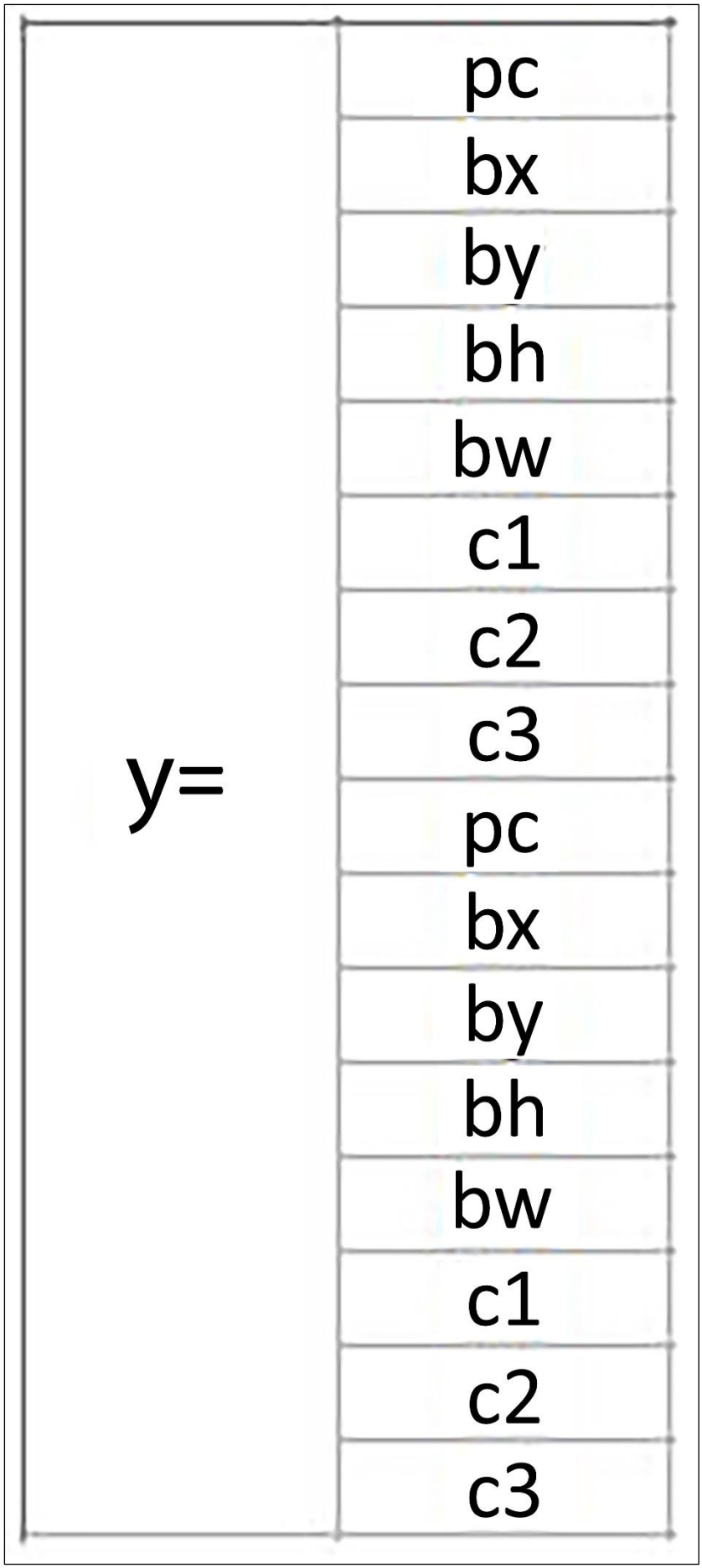
y=	0
	?
	?
	?
	?
	?
	?
	?

$$y = \begin{bmatrix} 1 \\ 0.5 \\ 0.5 \\ 0.95 \\ 0.8 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

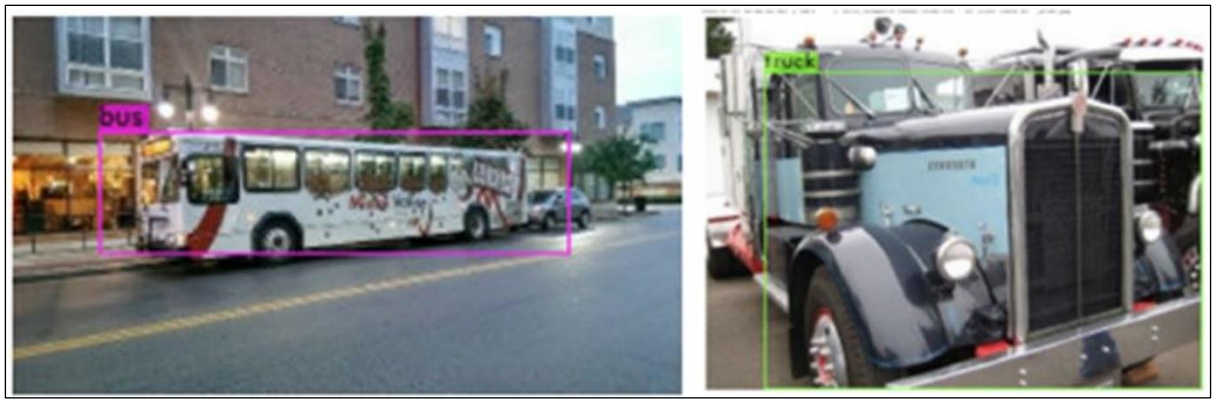


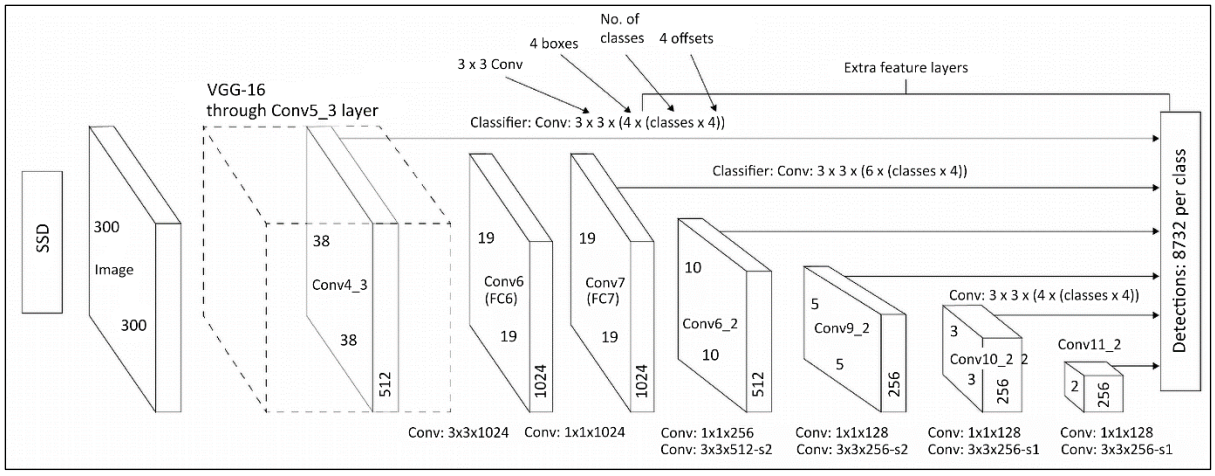

3
2
1
a
b
c

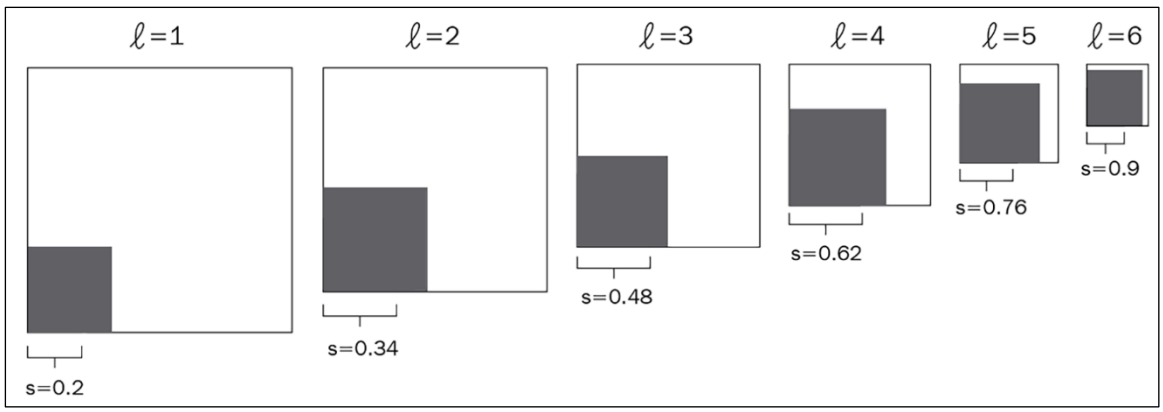


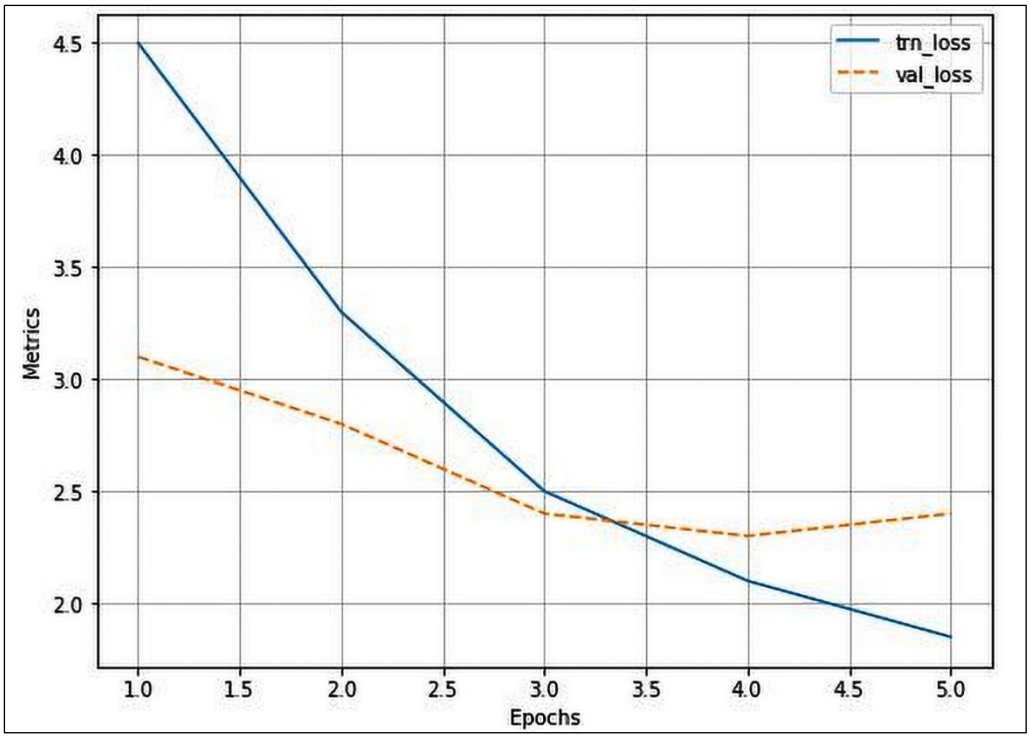

y=
pc
bx
by
bh
bw
c1
c2
c3
pc
bx
by
bh
bw
c1
c2
c3

```
data/person.jpg: Predicted in 54.532000 milli-seconds.
dog: 99%
person: 100%
horse: 98%
```



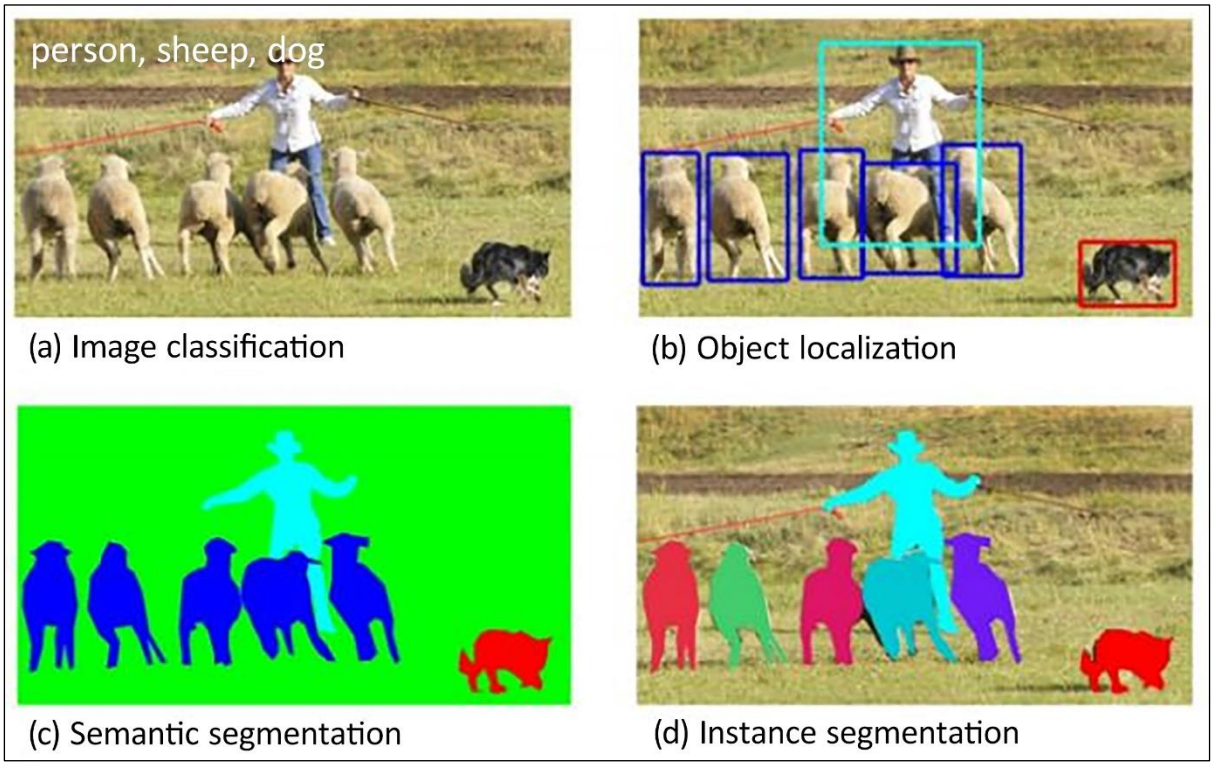


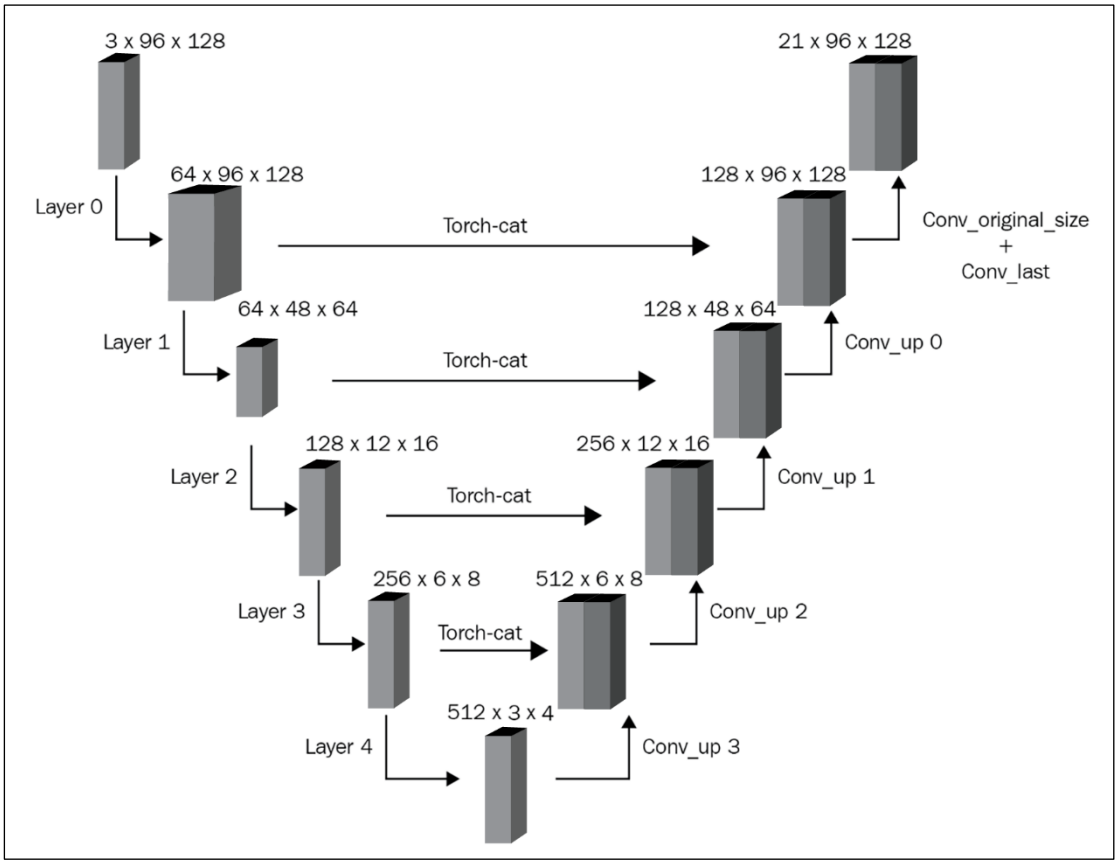



trn_loss
val_loss
Metrics
Epochs
4.5
4.0
3.5
3.0
2.5
2.0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0


Truck @ 1.00
Bus @ 0.99
Bus @ 1.00
SCHOOL BUS

Chapter 9:


person, sheep, dog
(a) Image classification
(b) Object localization
(c) Semantic segmentation
(d) Instance segmentation


3 x 96 x 128
Layer 0
64 x 96 x 128
Torch-cat
21 x 96 x 128
128 x 96 x 128
Conv_original_size
+
Conv_last
Layer 1
64 x 48 x 64
Torch-cat
128 x 48 x 64
Conv_up 0
Layer 2
128 x 12 x 16
Torch-cat
256 x 12 x 16
Conv_up 1
Layer 3
256 x 6 x 8
Torch-cat
512 x 6 x 8
Conv_up 2
Layer 4
512 x 3 x 4
Conv_up 3

Input array

1	1	1
1	1	1
1	1	1

Input array adjusted for stride

1	0	1	0	1
0	0	0	0	0
1	0	1	0	1
0	0	0	0	0
1	0	1	0	1

Input array adjusted for stride and padding

0	0	0	0	0	0	0
0	1	0	1	0	1	0
0	0	0	0	0	0	0
0	1	0	1	0	1	0
0	0	0	0	0	0	0
0	1	0	1	0	1	0
0	0	0	0	0	0	0

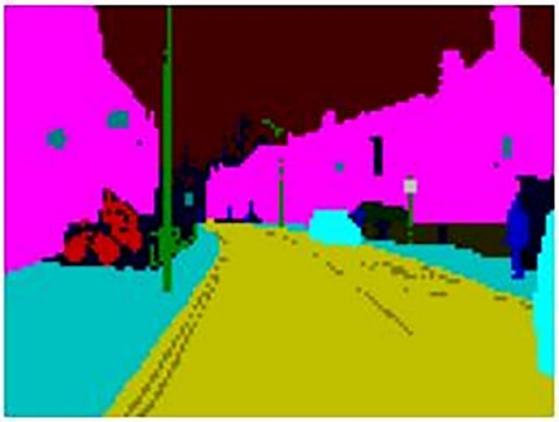
Filter/kernel

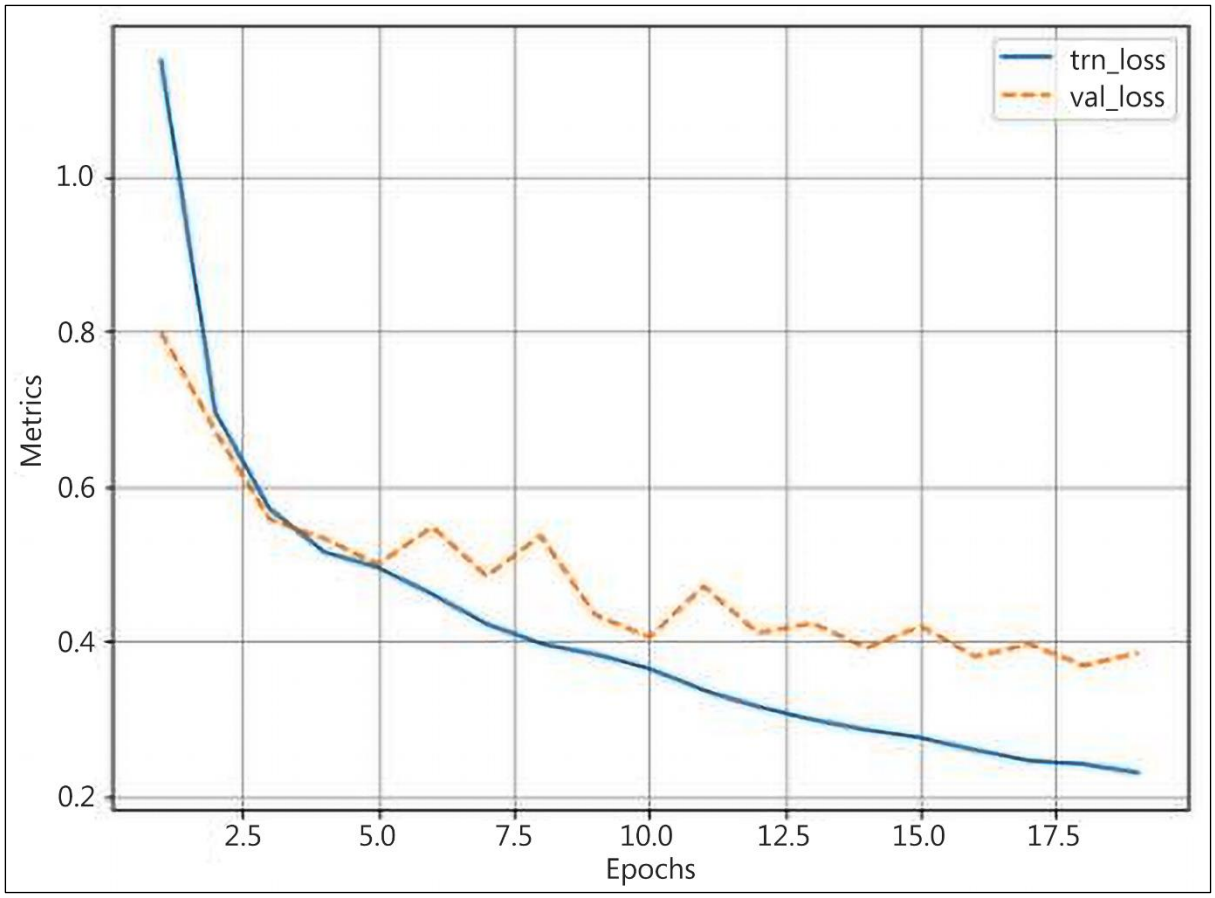
1	1
1	1

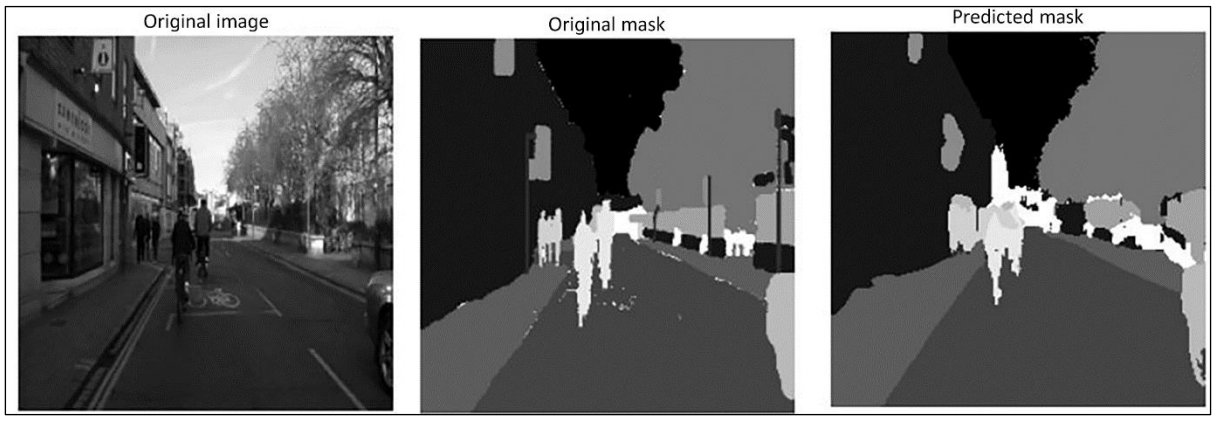
Output array

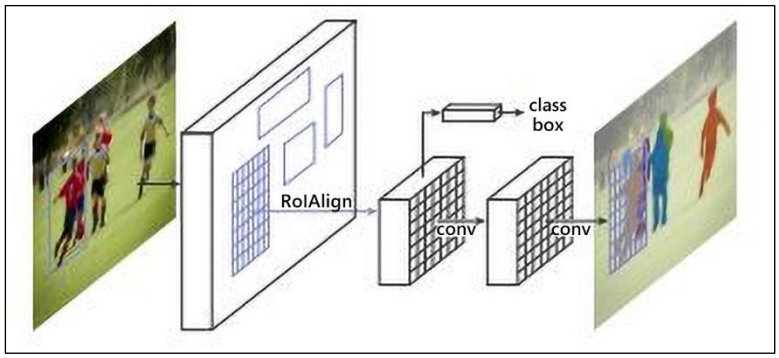
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1

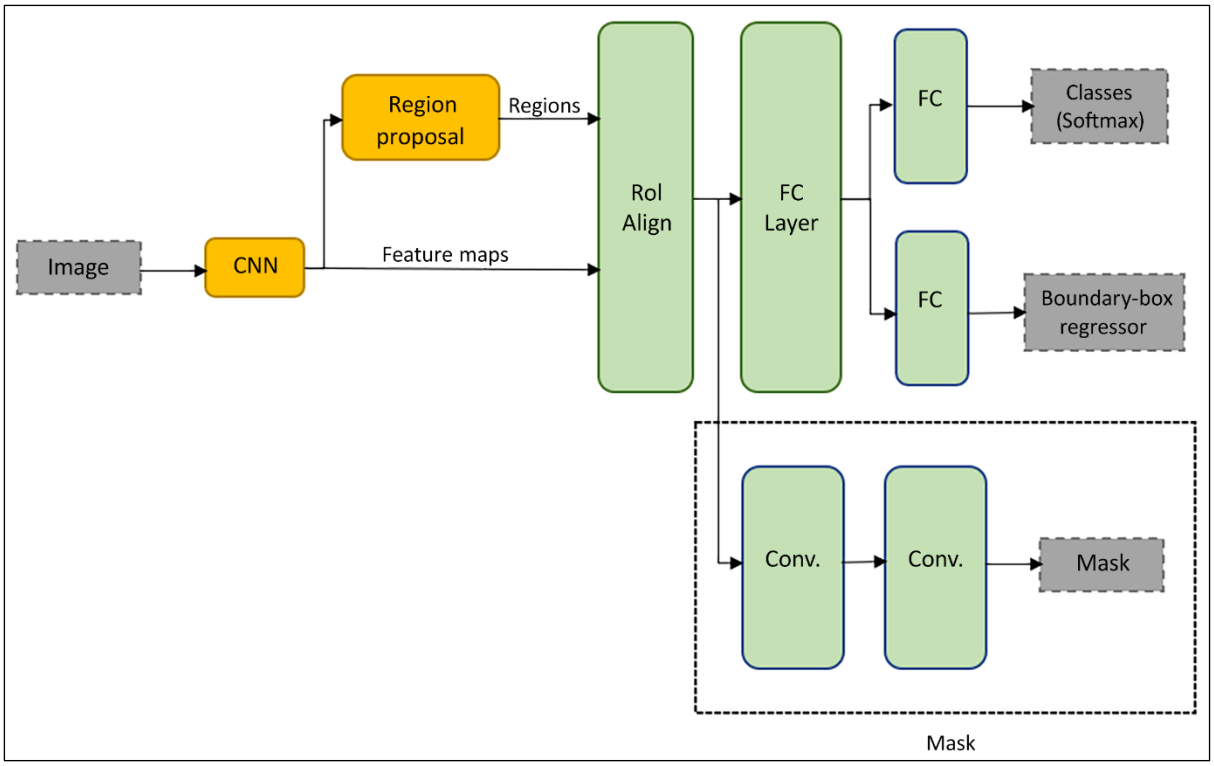
```
help(nn.ConvTranspose2d)

|  Args:
|      in_channels (int): Number of channels in the input image
|      out_channels (int): Number of channels produced by the convolution
|      kernel_size (int or tuple): Size of the convolving kernel
|      stride (int or tuple, optional): Stride of the convolution. Default: 1
|      padding (int or tuple, optional): ``dilation * (kernel_size - 1) - padding`` zero-padding
|          will be added to both sides of each dimension in the input. Default: 0
|      output_padding (int or tuple, optional): Additional size added to one side
|          of each dimension in the output shape. Default: 0
|      groups (int, optional): Number of blocked connections from input channels to output channels. Default: 1
|      bias (bool, optional): If ``True``, adds a learnable bias to the output. Default: ``True``
|      dilation (int or tuple, optional): Spacing between kernel elements. Default: 1
```

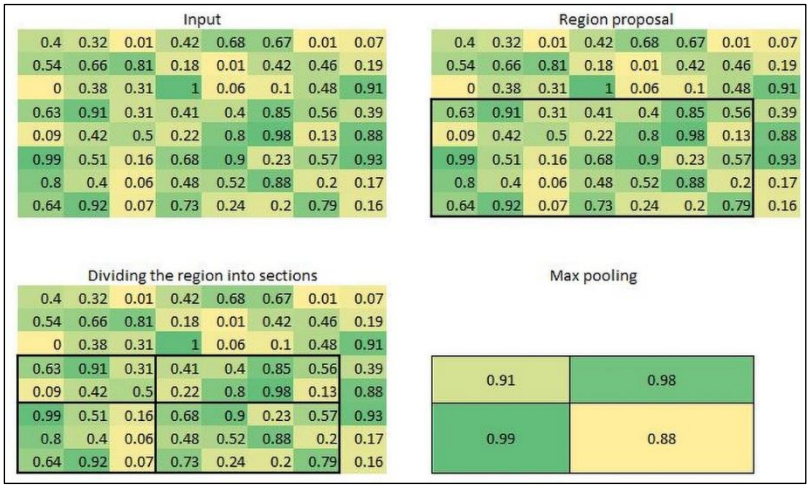



trn_loss
val_loss
Metrics
1.0
0.8
0.6
0.4
0.2
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Epochs

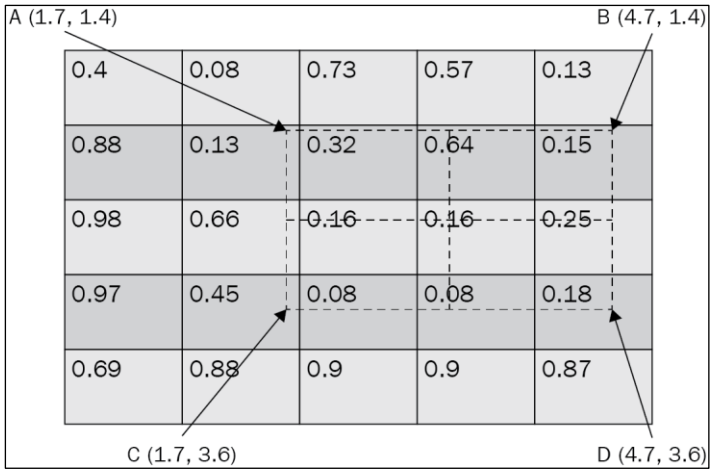

Original image
Original mask
Predicted mask

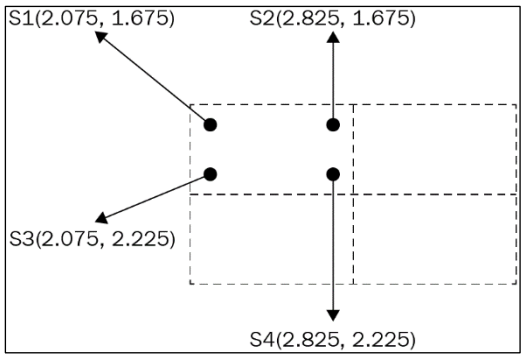

RoIAlign
class
box
conv
conv

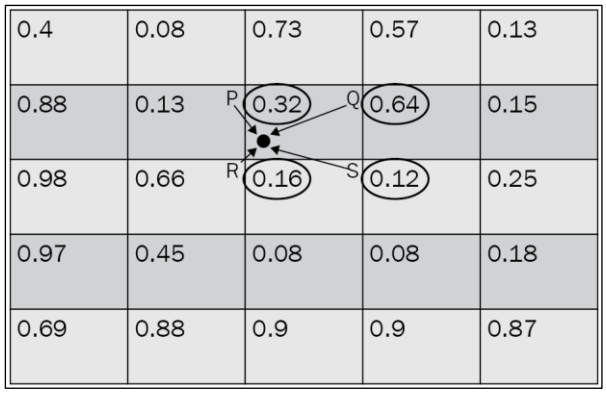


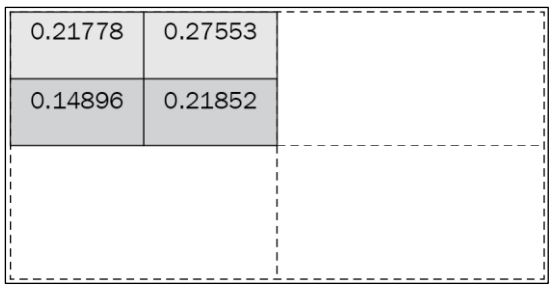


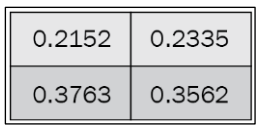
0.4	0.08	0.73	0.57	0.13
0.88	0.13	0.32	0.64	0.15
0.98	0.66	0.16	0.16	0.25
0.97	0.45	0.08	0.08	0.18
0.69	0.88	0.9	0.9	0.87

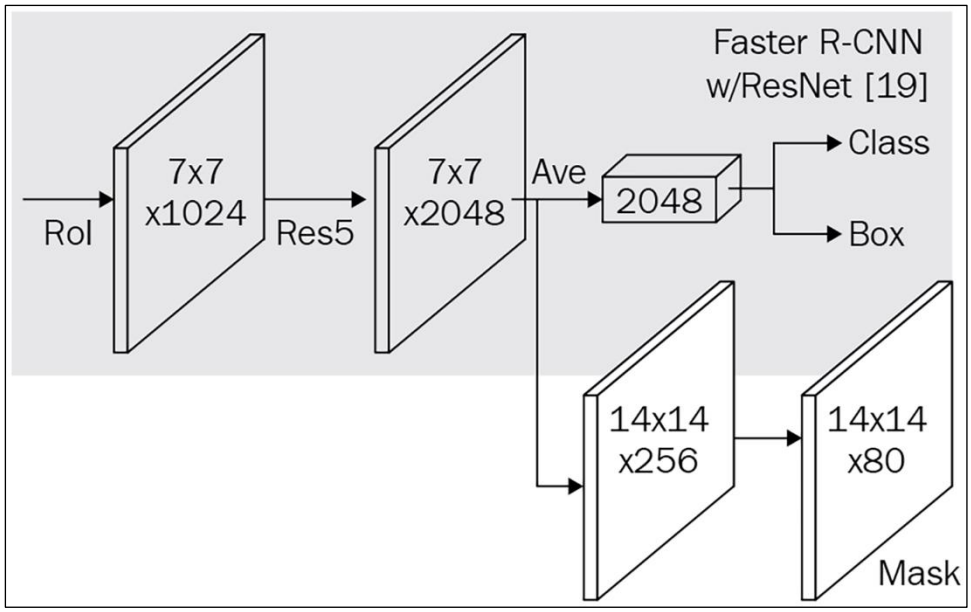

A (1.7, 1.4)
B (4.7, 1.4)
0.4
0.08
0.73
0.57
0.13
0.88
0.13
0.32
0.64
0.15
0.98
0.66
0.16
0.16
0.25
0.97
0.45
0.08
0.08
0.18
0.69
0.88
0.9
0.9
0.87
C (1.7, 3.6)
D (4.7, 3.6)

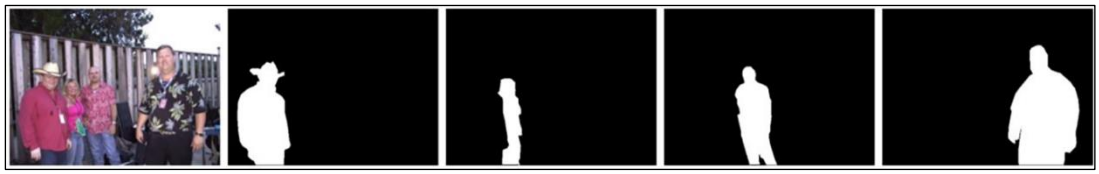

S1(2.075, 1.675)
S2(2.825, 1.675)
S3(2.075, 2.225)
S4(2.825, 2.225)


0.4
0.08
0.73
0.57
0.13
0.88
0.13
P
0.32
Q
0.64
0.15
0.98
0.66
R
0.16
S
0.12
0.25
0.97
0.45
0.08
0.08
0.18
0.69
0.88
0.9
0.9
0.87

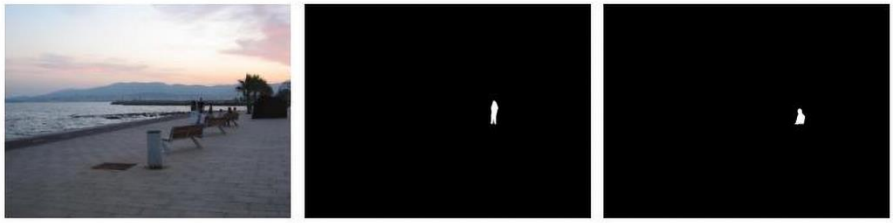

0.21778
0.27553
0.14896
0.21852


0.2152
0.2335
0.3763
0.3562





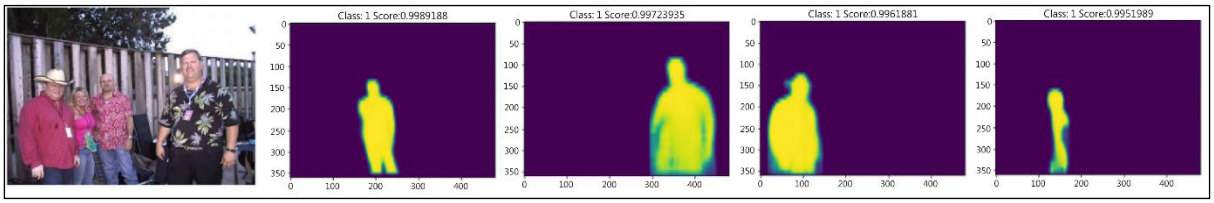
```
================================================================
Tensor  Shape: torch.Size([3, 512, 683])        Min: 0.000      Max: 1.000      Mean: 0.555     dtype: torch.float32
================================================================
Dict Of 6 items
================================================================
        BOXES:
        Tensor  Shape: torch.Size([2, 4])       Min: 233.000    Max: 480.000    Mean: 362.750   dtype: torch.float32
================================================================
================================================================
        LABELS:
        Tensor  Shape: torch.Size([2])  Min: 1.000      Max: 1.000      Mean: 1.000     dtype: torch.int64
================================================================
================================================================
        MASKS:
        Tensor  Shape: torch.Size([2, 512, 683])        Min: 0.000      Max: 1.000      Mean: 0.002     dtype: torch.uint8
================================================================
================================================================
        IMAGE_ID:
        Tensor  Shape: torch.Size([1])  Min: 0.000      Max: 0.000      Mean: 0.000     dtype: torch.int64
================================================================
================================================================
        AREA:
        Tensor  Shape: torch.Size([2])  Min: 864.000    Max: 935.000    Mean: 899.500   dtype: torch.float32
================================================================
... ... 1 more items
```

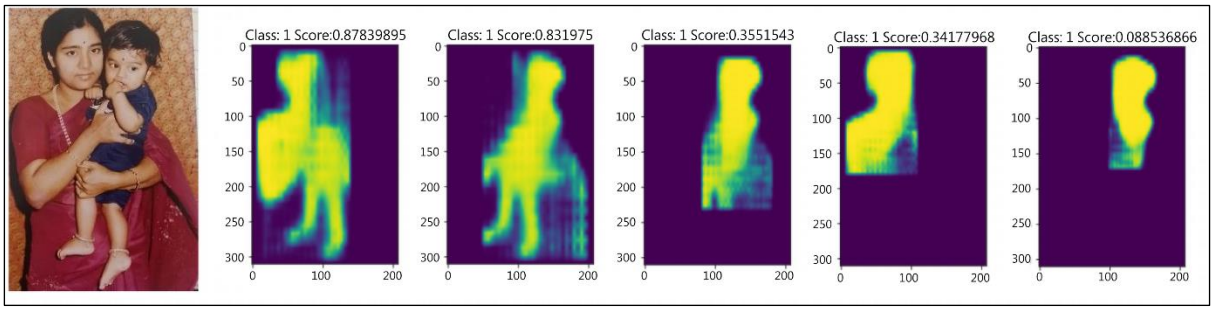


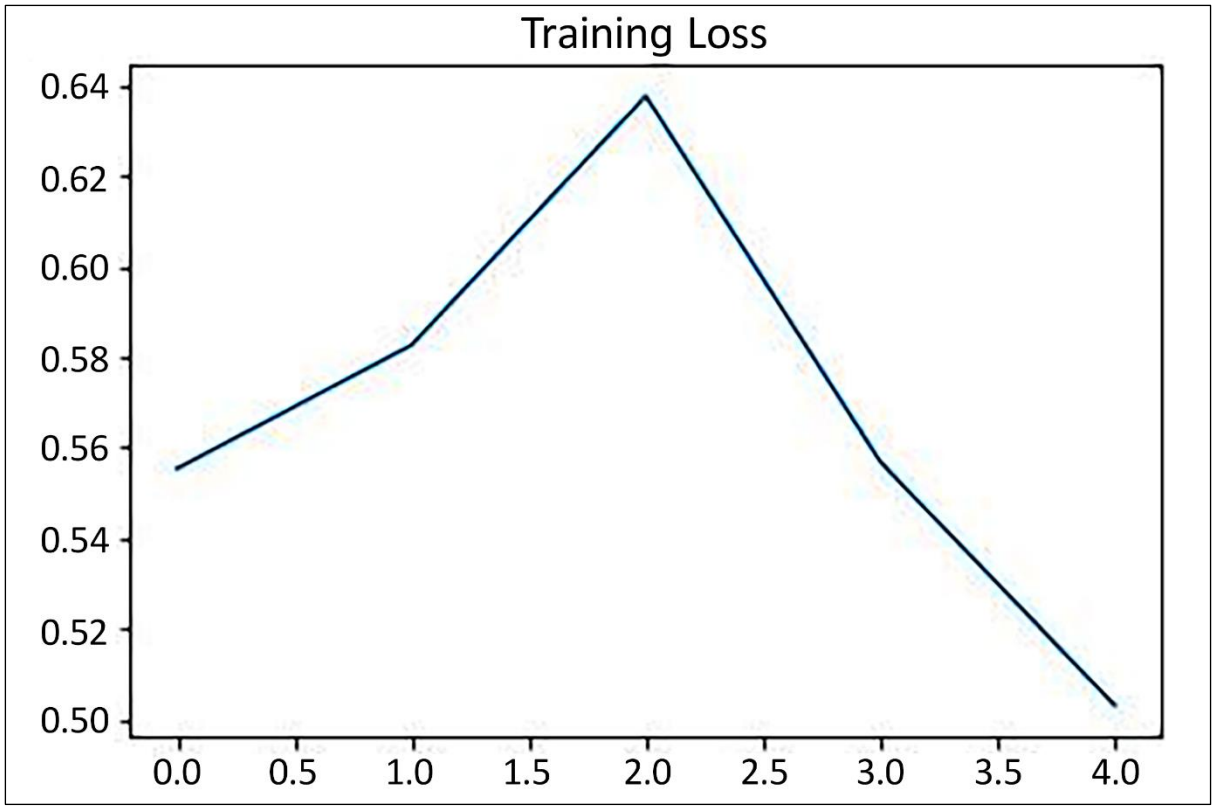
```
(roi_heads): RoIHeads(
  (box_roi_pool): MultiScaleRoIAlign()
  (box_head): TwoMLPHead(
    (fc6): Linear(in_features=12544, out_features=1024, bias=True)
    (fc7): Linear(in_features=1024, out_features=1024, bias=True)
  )
  (box_predictor): FastRCNNPredictor(
    (cls_score): Linear(in_features=1024, out_features=2, bias=True)
    (bbox_pred): Linear(in_features=1024, out_features=8, bias=True)
  )
  (mask_roi_pool): MultiScaleRoIAlign()
  (mask_head): MaskRCNNHeads(
    (mask_fcn1): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (relu1): ReLU(inplace=True)
    (mask_fcn2): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (relu2): ReLU(inplace=True)
    (mask_fcn3): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (relu3): ReLU(inplace=True)
    (mask_fcn4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (relu4): ReLU(inplace=True)
  )
  (mask_predictor): MaskRCNNPredictor(
    (conv5_mask): ConvTranspose2d(256, 256, kernel_size=(2, 2), stride=(2, 2))
    (relu): ReLU(inplace=True)
    (mask_fcn_logits): Conv2d(256, 2, kernel_size=(1, 1), stride=(1, 1))
```

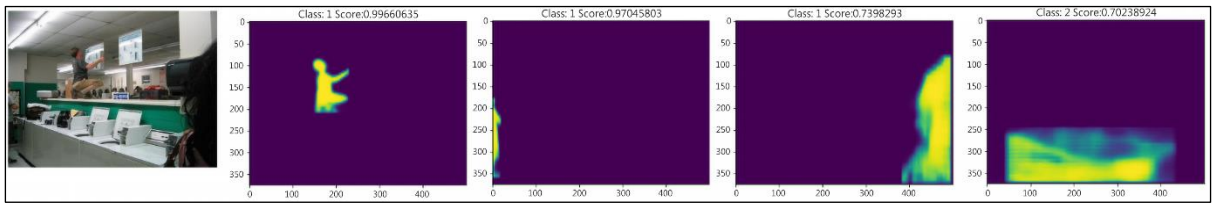
```
====================================================================
Dict Of 4 items
====================================================================
       BOXES:
       Tensor  Shape: torch.Size([100, 4])     Min: 0.000      Max: 1024.000   Mean: 385.767   dtype: torch.float32
====================================================================
====================================================================
       LABELS:
       Tensor  Shape: torch.Size([100])        Min: 1.000      Max: 1.000      Mean: 1.000     dtype: torch.int64
====================================================================
====================================================================
       SCORES:
       Tensor  Shape: torch.Size([100])        Min: 0.491      Max: 0.648      Mean: 0.531     dtype: torch.float32
====================================================================
====================================================================
       MASKS:
       Tensor  Shape: torch.Size([100, 1, 692, 1024])  Min: 0.000      Max: 1.000      Mean: 0.012     dtype: torch.float32
====================================================================
```


Training Loss
0.70
0.65
0.60
0.55
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

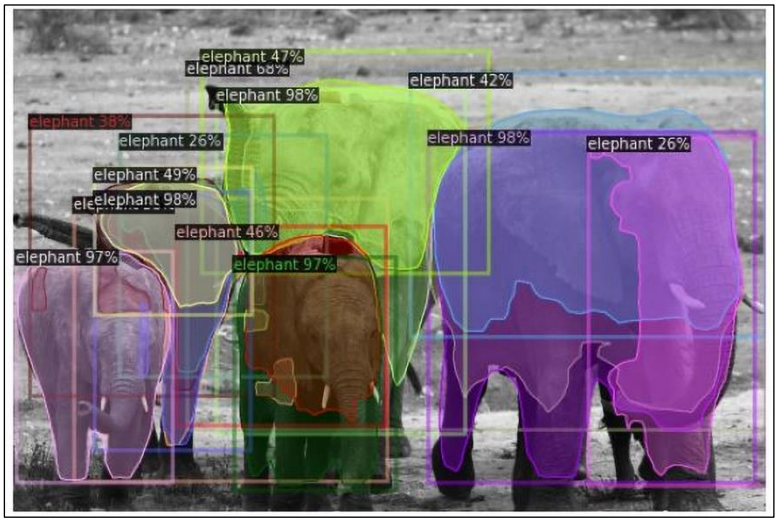

Class: 1 Score:0.9989188
Class: 1 Score:0.99723935
Class: 1 Score:0.9961881
Class: 1 Score:0.9951989

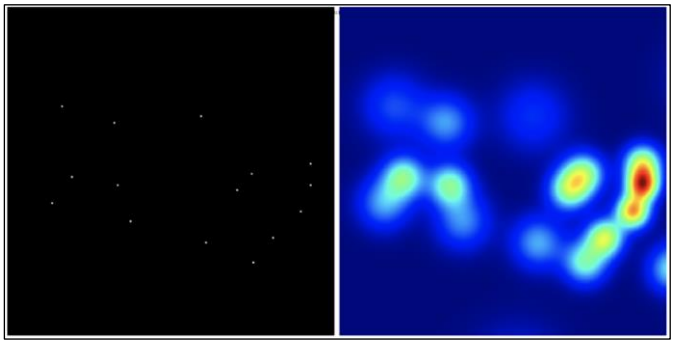

Class: 1 Score:0.87839895
Class: 1 Score:0.831975
Class: 1 Score:0.3551543
Class: 1 Score:0.34177968
Class: 1 Score:0.088536866


Training Loss
0.64
0.62
0.60
0.58
0.56
0.54
0.52
0.50
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

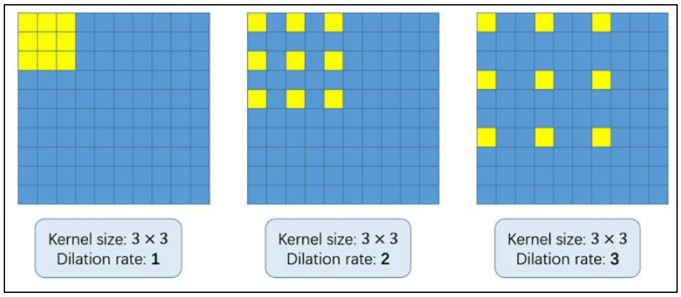

Class: 1 Score:0.99660635
Class: 1 Score:0.97045803
Class: 1 Score:0.7398293
Class: 2 Score:0.70238924

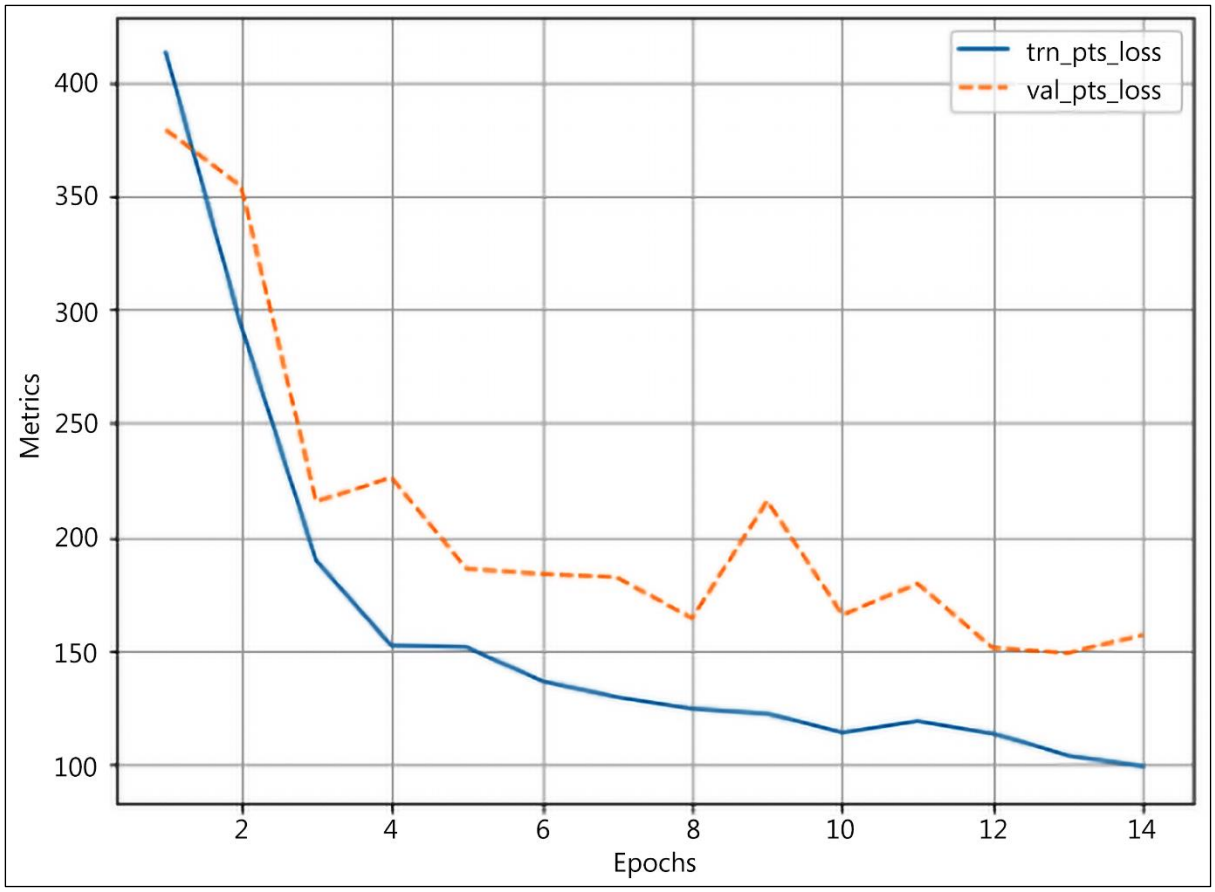
Chapter 10:

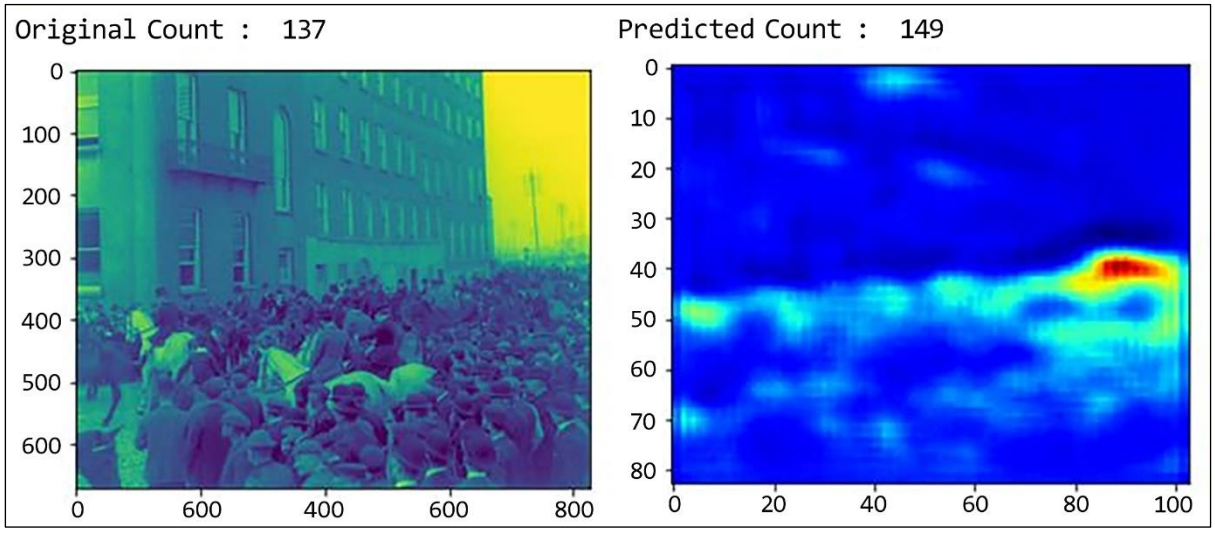

elephant 47%
elephant 42%
elephant 98%
elephant 38%
elephant 26%
elephant 98%
elephant 26%
elephant 49%
elephant 98%
elephant 46%
elephant 97%
elephant 97%

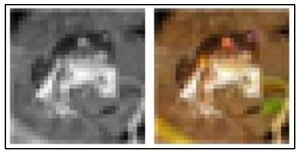


Configurations of CSRNet			
A	B	C	D
input(unfixed-resolution color image)			
front-end (fine-tuned from VGG-16)			
conv3-64-1 conv3-64-1			
max-pooling			
conv3-128-1 conv3-128-1			
max-pooling			
conv3-256-1 conv3-256-1 conv3-256-1			
max-pooling			
conv3-512-1 conv3-512-1 conv3-512-1			
back-end (four different configurations)			
conv3-512-1 conv3-512-1 conv3-512-1 conv3-256-1 conv3-128-1 conv3-64-1	conv3-512-2 conv3-512-2 conv3-512-2 conv3-256-2 conv3-128-2 conv3-64-2	conv3-512-2 conv3-512-2 conv3-512-2 conv3-256-4 conv3-128-4 conv3-64-4	conv3-512-4 conv3-512-4 conv3-512-4 conv3-256-4 conv3-128-4 conv3-64-4
conv1-1-1			

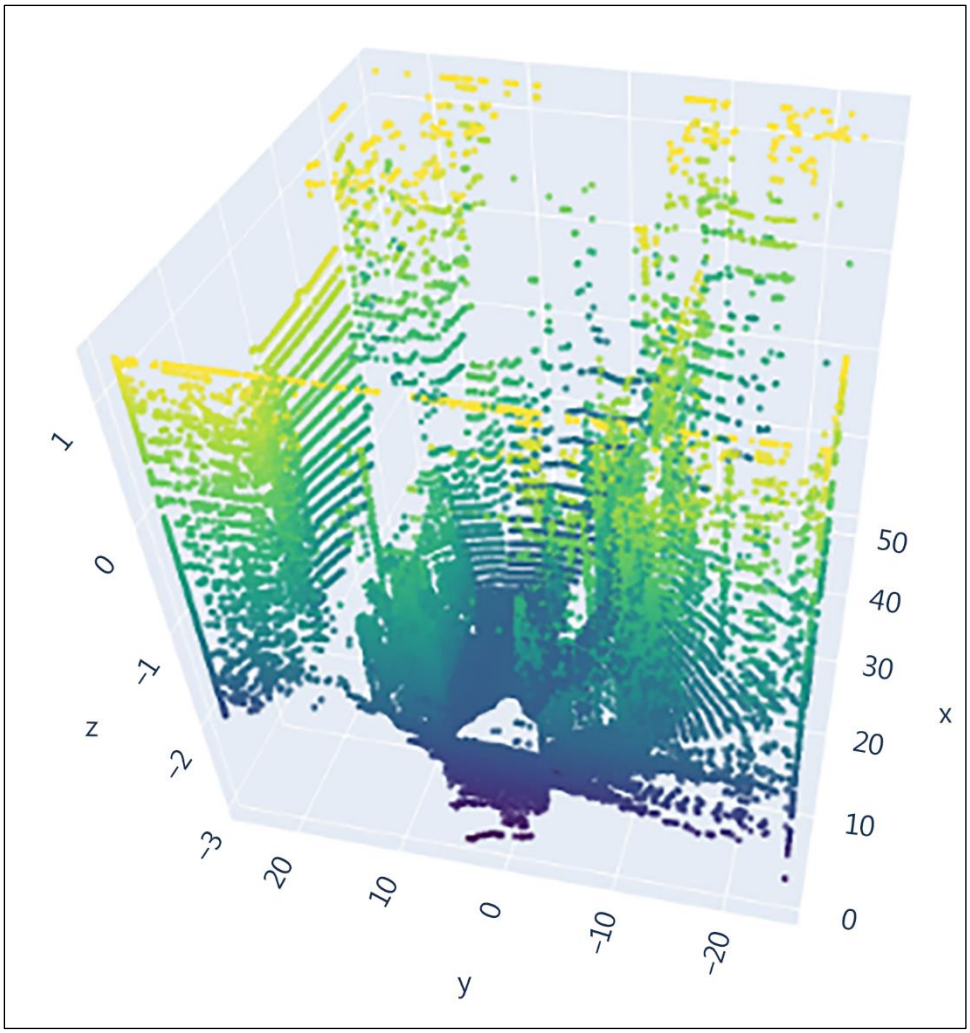


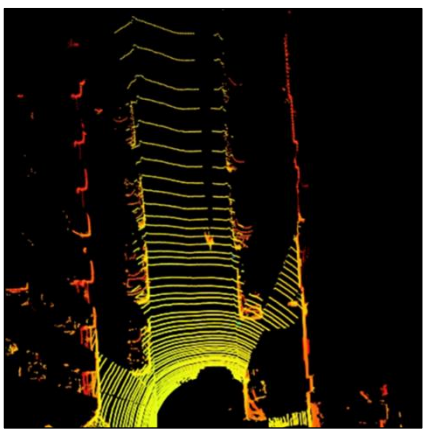

trn_pts_loss
val_pts_loss
Metrics
Epochs
400
350
300
250
200
150
100
2
4
6
8
10
12
14

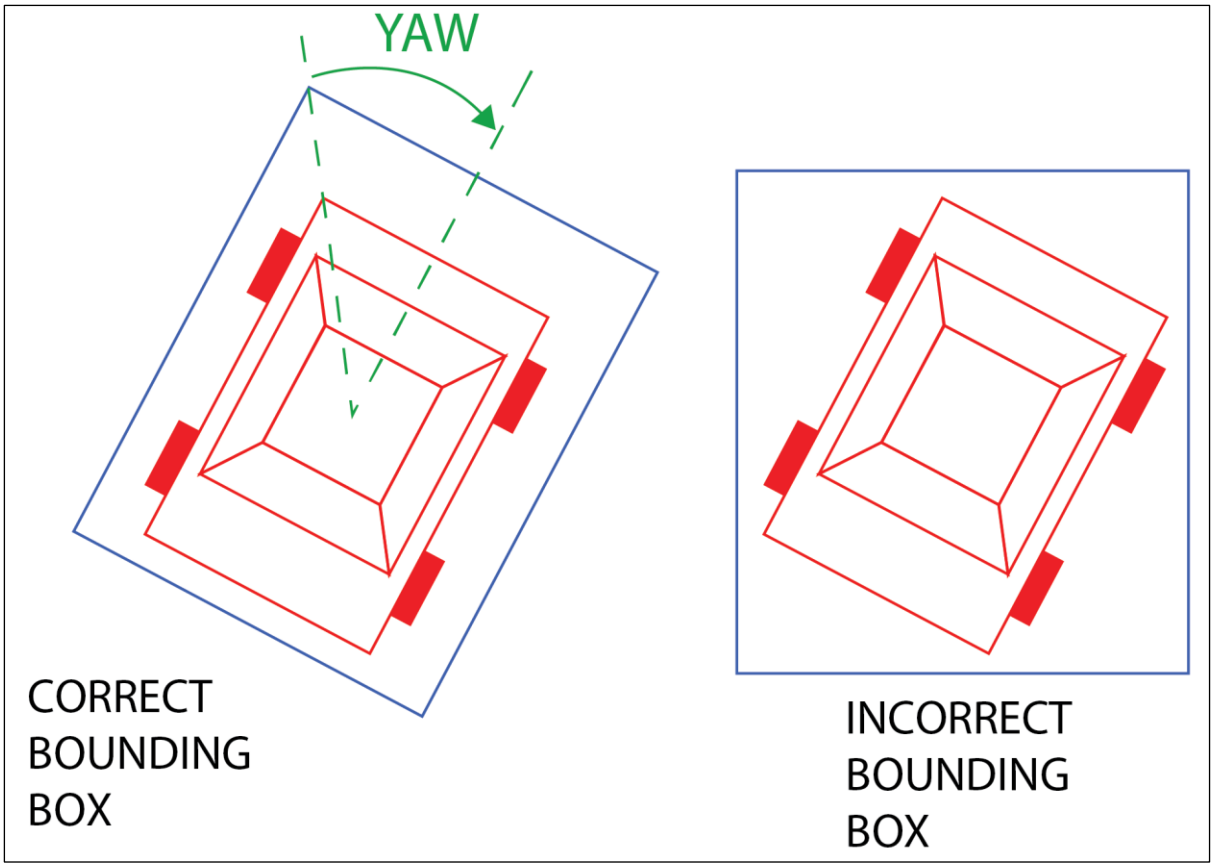

Original Count : 137
Predicted Count : 149
0
100
200
300
400
500
600
0
600
400
600
800
0
10
20
30
40
50
60
70
80
0
20
40
60
80
100

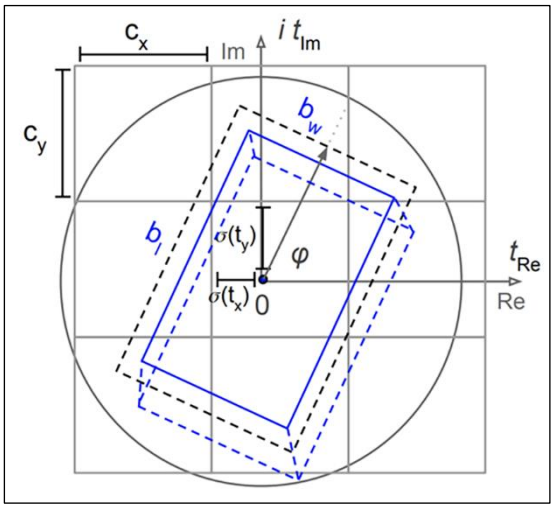


```
array([[62.502,  8.628,  2.343,  0.   ],
       [62.468,  8.824,  2.342,  0.   ],
       [66.793, 10.832,  2.497,  0.   ],
       ...,
       [ 3.75 , -1.418, -1.753,  0.25 ],
       [ 3.759, -1.409, -1.756,  0.32 ],
       [ 3.767, -1.398, -1.758,  0.   ]], dtype=float32)
```



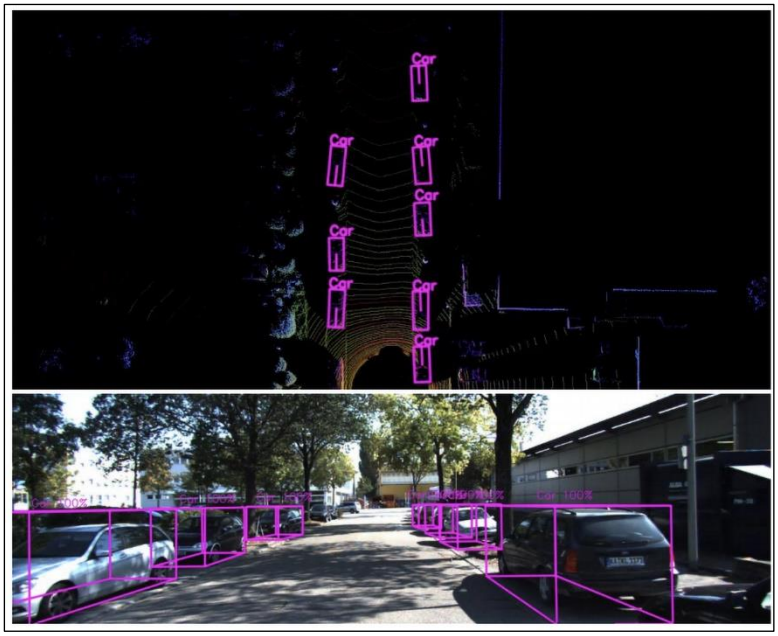


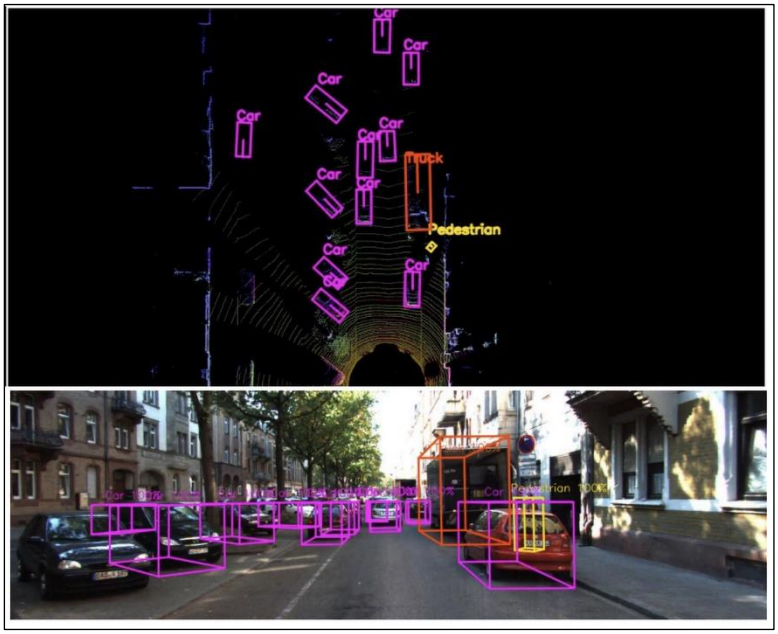


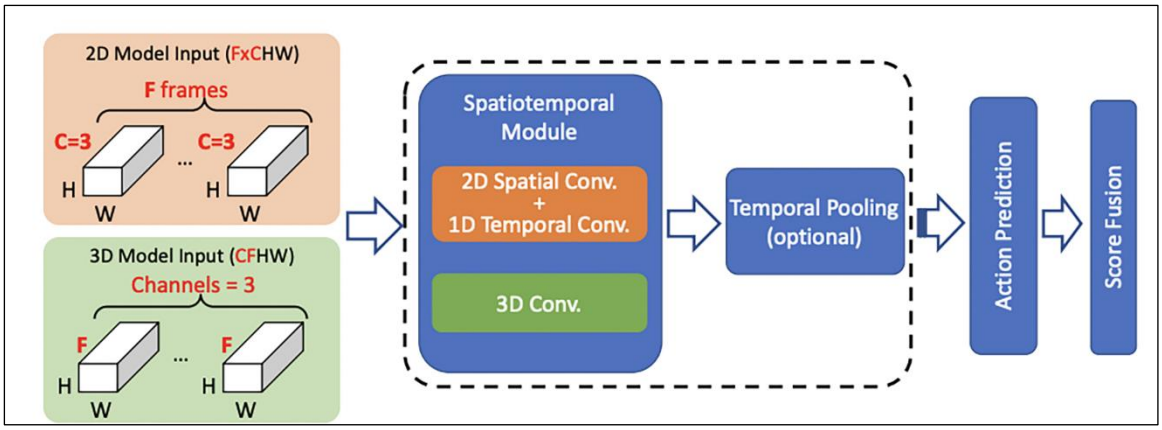



```
/Complex-YOLOv4-Pytorch  master !1  base
tree dataset
dataset
└── [redacted]
    ├── classes_names.txt
    ├── ImageSets
    │   ├── test.txt
    │   ├── train.txt
    │   └── val.txt
    ├── testing
    │   ├── calib
    │   ├── image_2
    │   └── velodyne
    └── training
        ├── calib
        ├── image_2
        ├── label_2
        └── velodyne
```

COLUMN	Example	Description	Range
type	Pedestrian	Class	Car/Pedestrian
truncation	0	Is object leaving image boundaries	0/1
occlusion	0	Is object occluded (0=fully visible, 1=partial, 2=mostly, 3=unknown)	0,1,2,3
alpha	-0.2	Observation angle	-pi to pi
x1	712.4	bbox	Image-coordinates
y1	143	bbox	Image-coordinates
x2	810.73	bbox	Image-coordinates
y2	307.92	bbox	Image-coordinates
h	1.89	height	meters
w	0.48	width	meters
l	1.2	length	meters
x	1.84	object location from camera	meters
y	1.47	object location from camera	meters
z	8.41	object location from camera	meters
ry	0.01	rotation of object around its own y-axis	-pi to pi



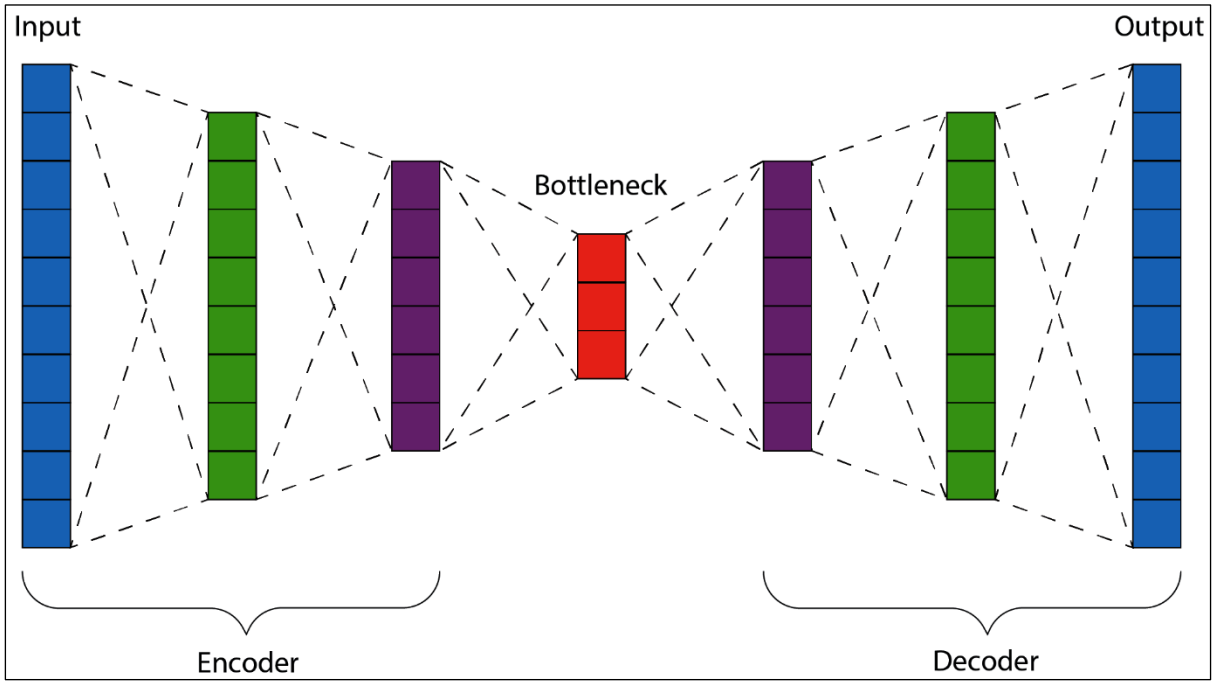




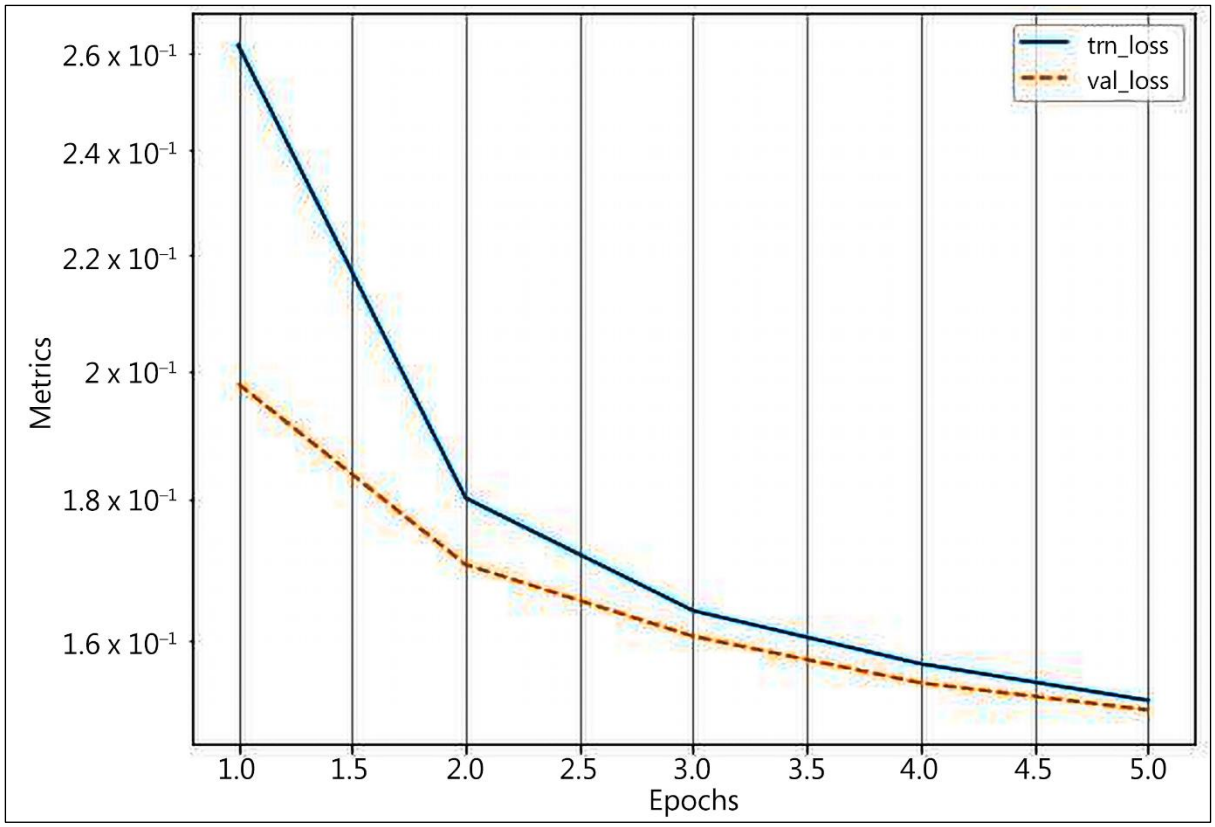

```
The top-5 labels with corresponding scores are:
arm wrestling:  1.0
rock scissors paper:  6.434453414527752e-09
shaking hands:  2.7599860175087088e-09
clapping:  1.3454612979302283e-09
massaging feet:  5.555100823784187e-10
```

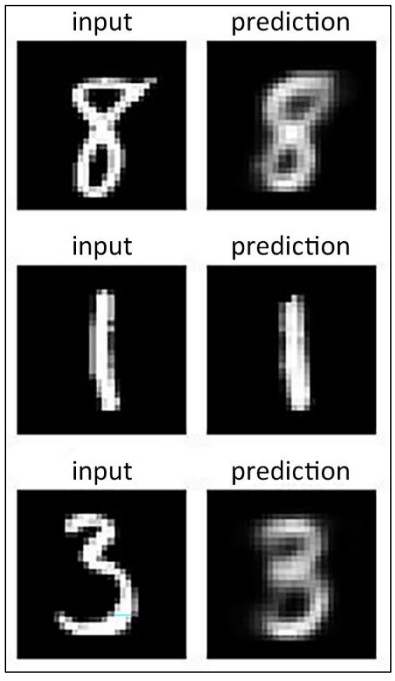
```
D32_1gwq35E.mp4 0
iRuyZSKhHRg.mp4 1
oXy-e_P_cAI.mp4 0
34XczvTaRiI.mp4 1
h2YqqUhnR34.mp4 0
O46YA8tI530.mp4 0
kFC3KY2bOP8.mp4 1
```

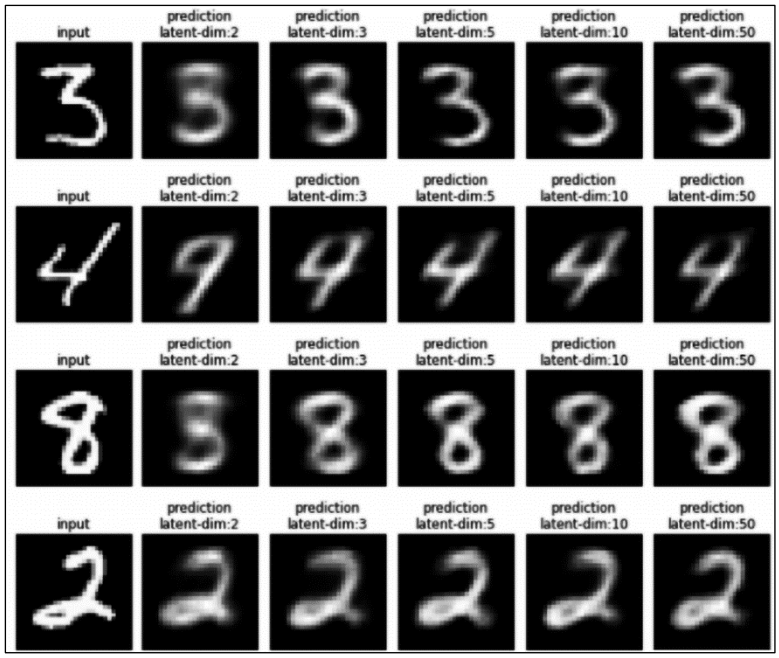
Chapter 11:

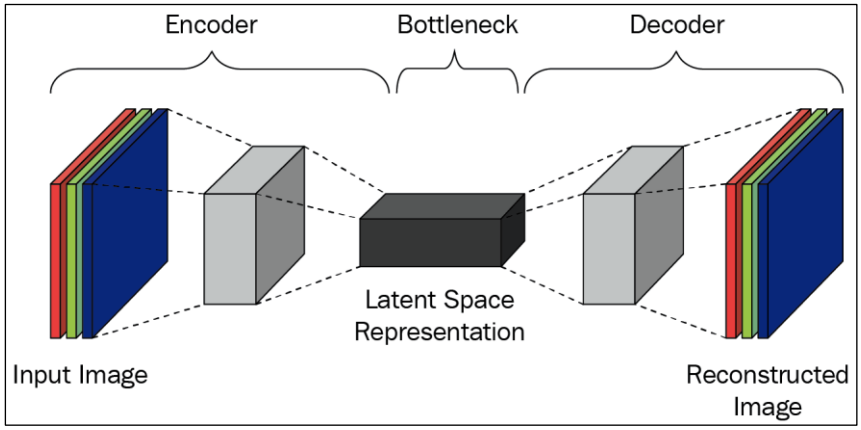



```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
├─Sequential: 1-1                        [-1, 3]                   --
|    └─Linear: 2-1                       [-1, 128]                 100,480
|    └─ReLU: 2-2                         [-1, 128]                 --
|    └─Linear: 2-3                       [-1, 64]                  8,256
|    └─ReLU: 2-4                         [-1, 64]                  --
|    └─Linear: 2-5                       [-1, 3]                   195
├─Sequential: 1-2                        [-1, 784]                 --
|    └─Linear: 2-6                       [-1, 64]                  256
|    └─ReLU: 2-7                         [-1, 64]                  --
|    └─Linear: 2-8                       [-1, 128]                 8,320
|    └─ReLU: 2-9                         [-1, 128]                 --
|    └─Linear: 2-10                      [-1, 784]                 101,136
|    └─Tanh: 2-11                        [-1, 784]                 --
==========================================================================================
Total params: 218,643
Trainable params: 218,643
Non-trainable params: 0
Total mult-adds (M): 0.43
```

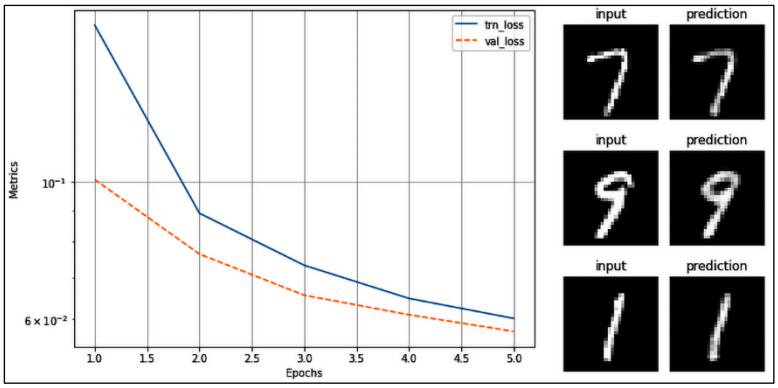

trn_loss
val_loss
Metrics
2.6 x 10⁻¹
2.4 x 10⁻¹
2.2 x 10⁻¹
2 x 10⁻¹
1.8 x 10⁻¹
1.6 x 10⁻¹
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Epochs

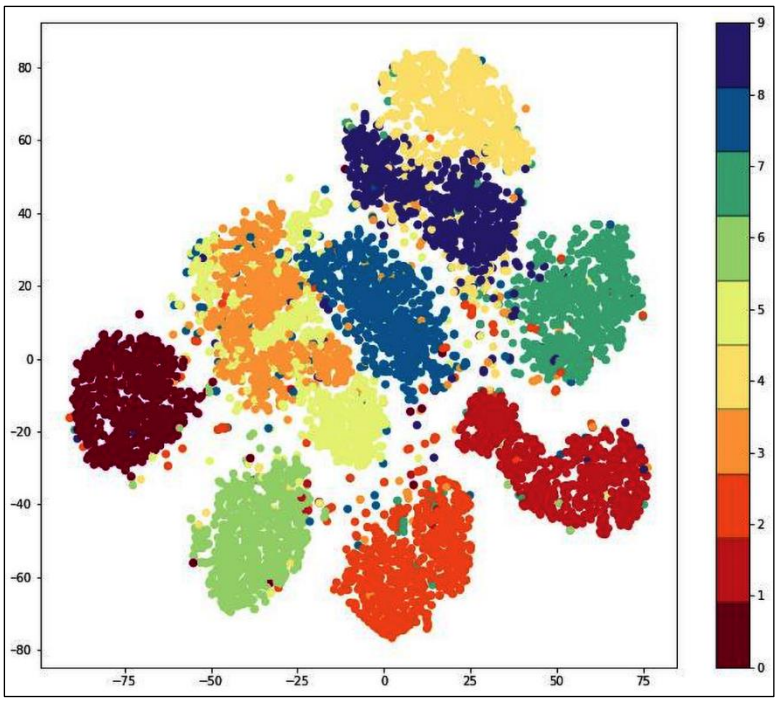

input
prediction
input
prediction
input
prediction

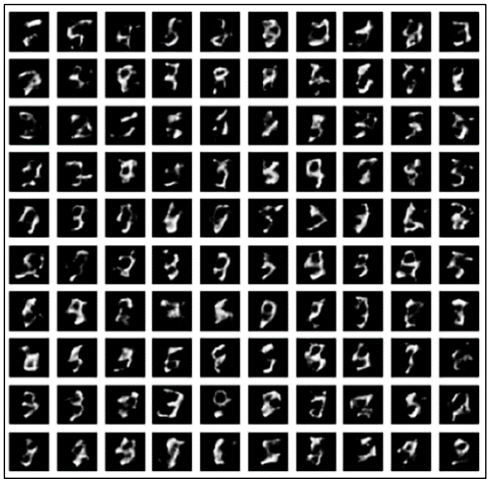


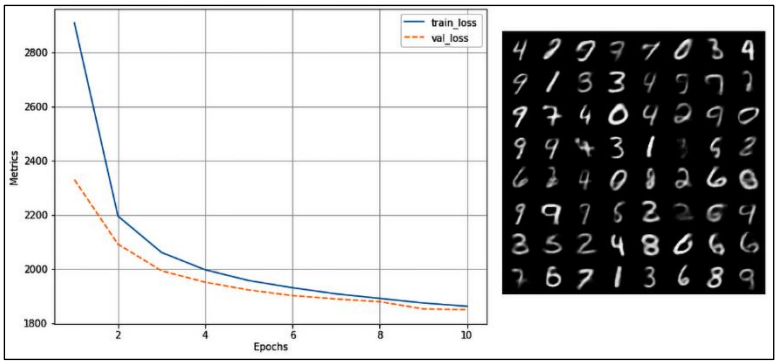


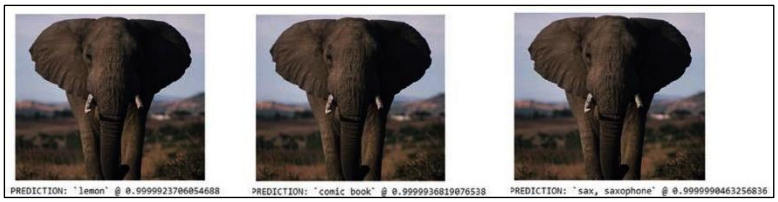

```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
├─Sequential: 1-1                        [-1, 64, 2, 2]            --
|    └─Conv2d: 2-1                       [-1, 32, 10, 10]          320
|    └─ReLU: 2-2                         [-1, 32, 10, 10]          --
|    └─MaxPool2d: 2-3                    [-1, 32, 5, 5]            --
|    └─Conv2d: 2-4                       [-1, 64, 3, 3]            18,496
|    └─ReLU: 2-5                         [-1, 64, 3, 3]            --
|    └─MaxPool2d: 2-6                    [-1, 64, 2, 2]            --
├─Sequential: 1-2                        [-1, 1, 28, 28]           --
|    └─ConvTranspose2d: 2-7              [-1, 32, 5, 5]            18,464
|    └─ReLU: 2-8                         [-1, 32, 5, 5]            --
|    └─ConvTranspose2d: 2-9              [-1, 16, 15, 15]          12,816
|    └─ReLU: 2-10                        [-1, 16, 15, 15]          --
|    └─ConvTranspose2d: 2-11             [-1, 1, 28, 28]           65
|    └─Tanh: 2-12                        [-1, 1, 28, 28]           --
==========================================================================================
Total params: 50,161
Trainable params: 50,161
Non-trainable params: 0
Total mult-adds (M): 3.64
```

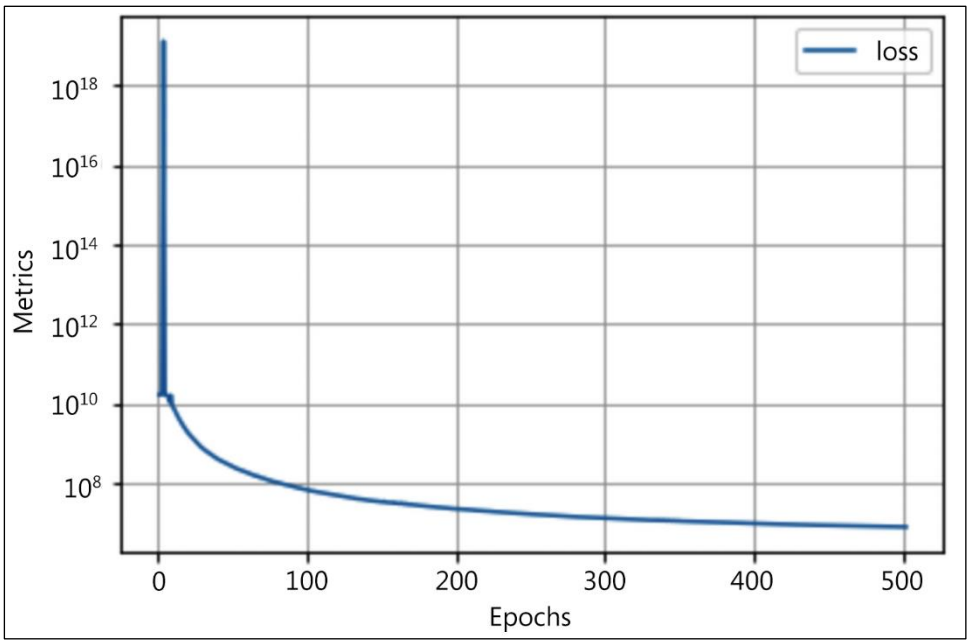

trn_loss
val_loss
Metrics
10^{-1}
6×10^{-2}
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Epochs
input
prediction
input
prediction
input
prediction

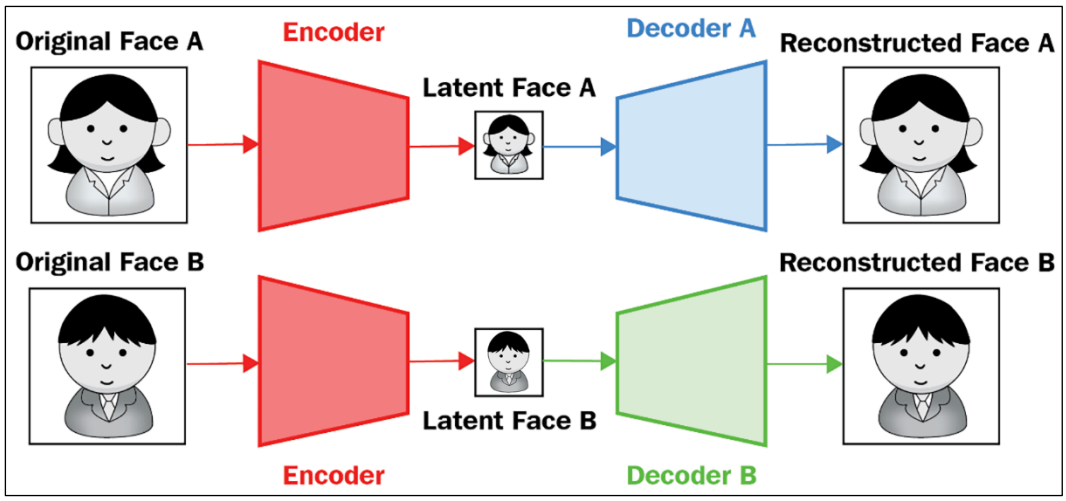

80
60
40
20
0
-20
-40
-60
-80
−75
−50
−25
0
25
50
75
9
8
7
6
5
4
3
2
1
0

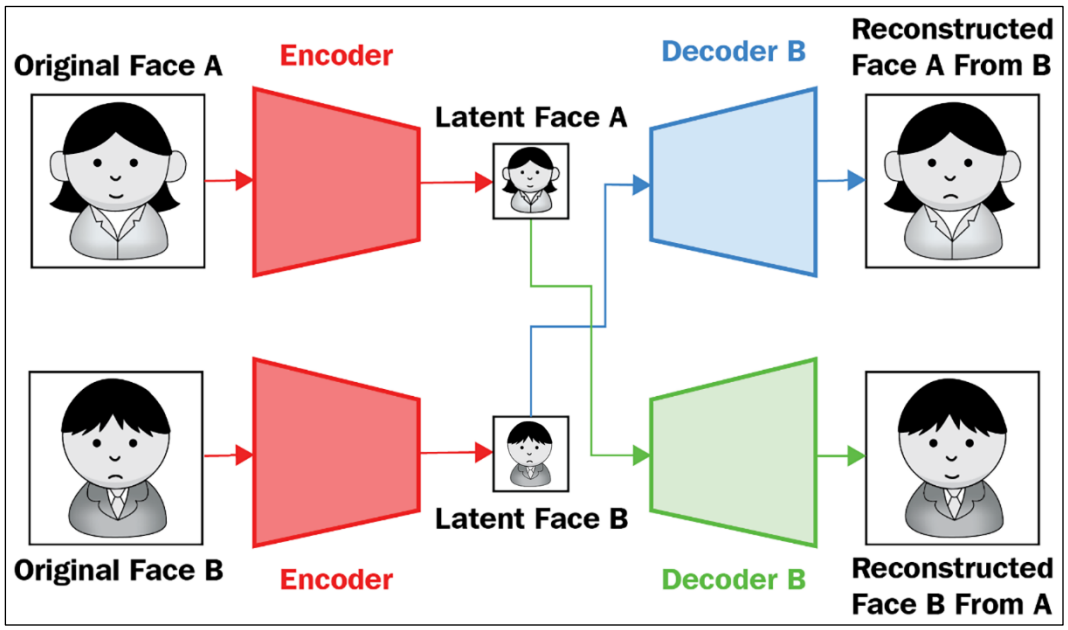


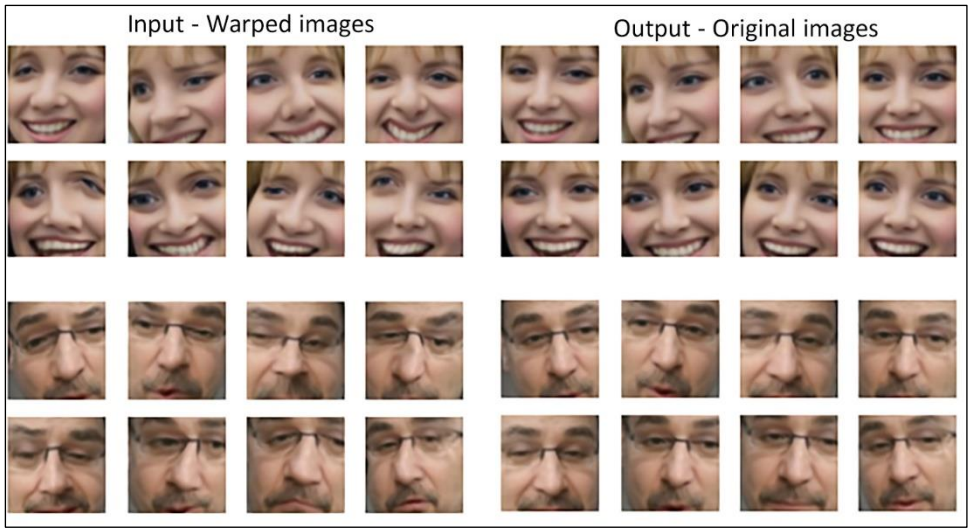

train_loss
val_loss
Metrics
Epochs
2800
2600
2400
2200
2000
1800
2
4
6
8
10


PREDICTION: `lemon` @ 0.999992370605468
PREDICTION: `comic book` @ 0.9999936819076538
PREDICTION: `sax, saxophone` @ 0.9999990463256836

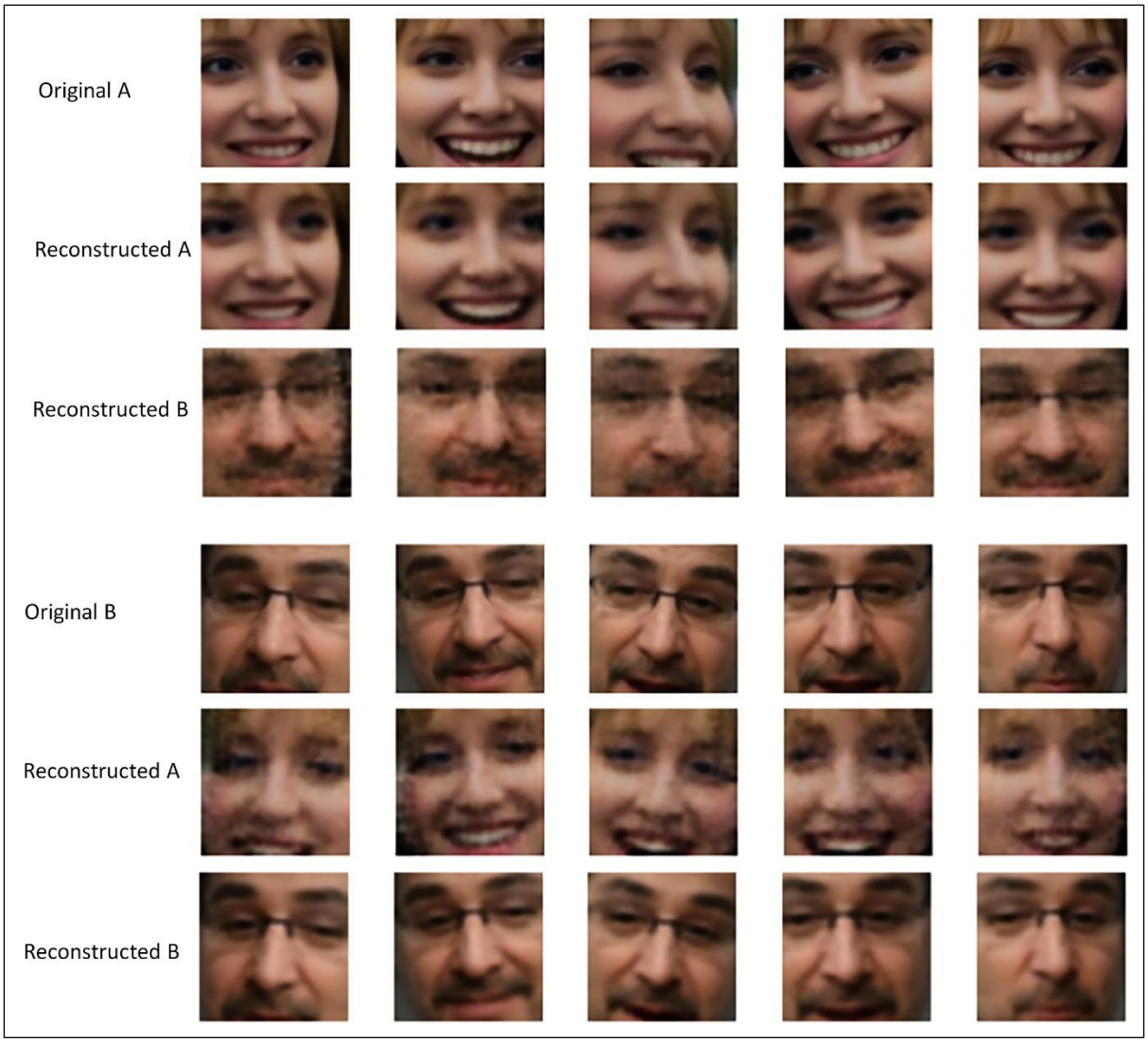

loss
Metrics
Epochs
10^{18}
10^{16}
10^{14}
10^{12}
10^{10}
10^{8}
0
100
200
300
400
500

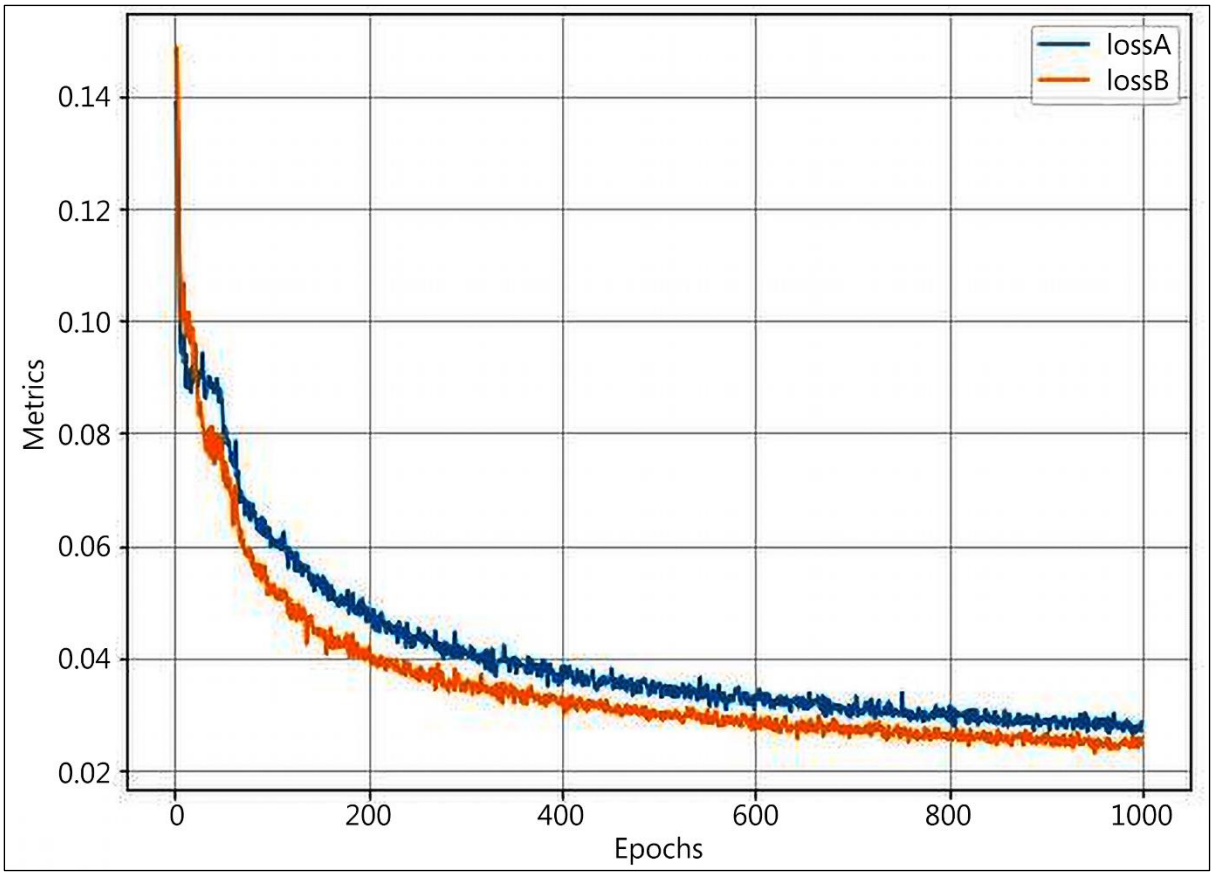

Original Face A
Encoder
Latent Face A
Decoder A
Reconstructed Face A
Original Face B
Reconstructed Face B
Latent Face B
Encoder
Decoder B


Original Face A
Encoder
Latent Face A
Decoder B
Reconstructed Face A From B
Original Face B
Encoder
Latent Face B
Decoder B
Reconstructed Face B From A

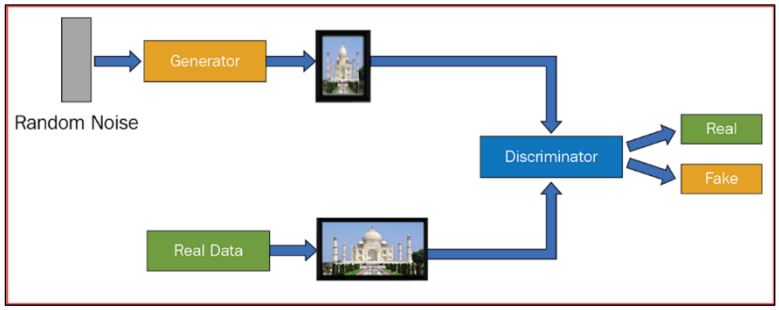


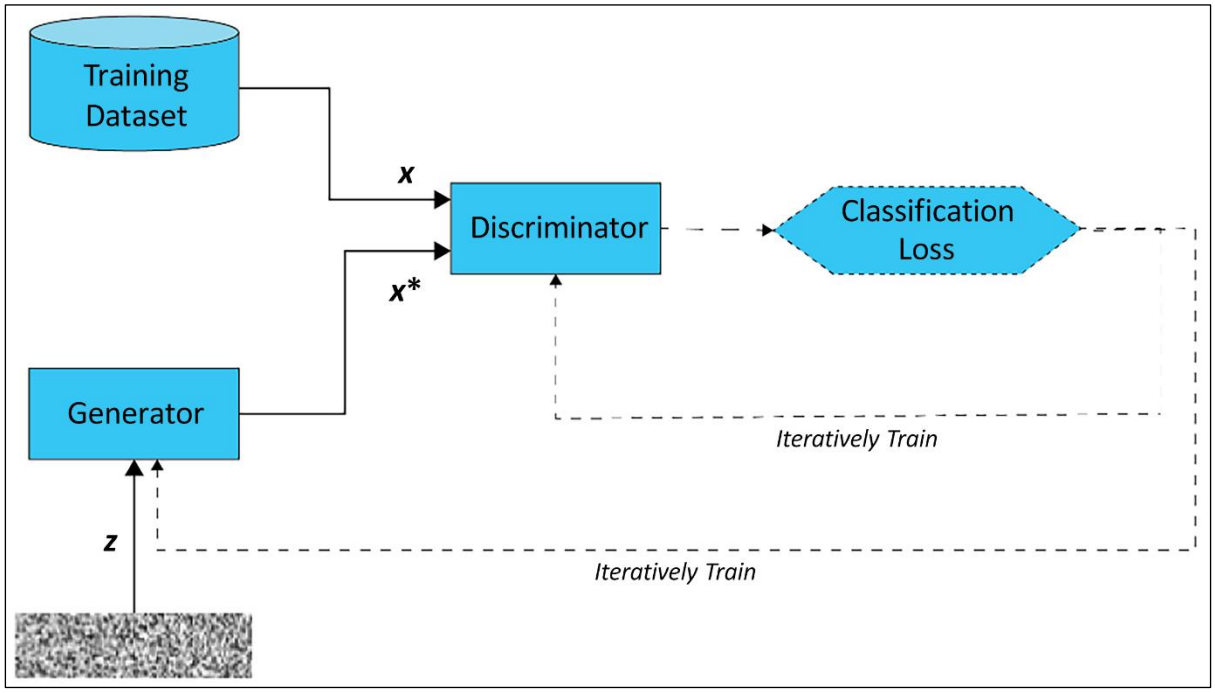

```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
├─Sequential: 1-1                        [-1, 512, 8, 8]           --
|    └─Sequential: 2-1                   [-1, 128, 32, 32]         --
|    |    └─Conv2d: 3-1                  [-1, 128, 32, 32]         9,728
|    |    └─LeakyReLU: 3-2               [-1, 128, 32, 32]         --
|    └─Sequential: 2-2                   [-1, 256, 16, 16]         --
|    |    └─Conv2d: 3-3                  [-1, 256, 16, 16]         819,456
|    |    └─LeakyReLU: 3-4               [-1, 256, 16, 16]         --
|    └─Sequential: 2-3                   [-1, 512, 8, 8]           --
|    |    └─Conv2d: 3-5                  [-1, 512, 8, 8]           3,277,312
|    |    └─LeakyReLU: 3-6               [-1, 512, 8, 8]           --
|    └─Sequential: 2-4                   [-1, 1024, 4, 4]          --
|    |    └─Conv2d: 3-7                  [-1, 1024, 4, 4]          13,108,224
|    |    └─LeakyReLU: 3-8               [-1, 1024, 4, 4]          --
|    └─Flatten: 2-5                      [-1, 16384]               --
|    └─Linear: 2-6                       [-1, 1024]                16,778,240
|    └─Linear: 2-7                       [-1, 16384]               16,793,600
|    └─Reshape: 2-8                      [-1, 1024, 4, 4]          --
|    └─Sequential: 2-9                   [-1, 512, 8, 8]           --
|    |    └─ConvTranspose2d: 3-9         [-1, 512, 8, 8]           2,097,664
|    |    └─LeakyReLU: 3-10              [-1, 512, 8, 8]           --
├─Sequential: 1-2                        [-1, 3, 64, 64]           --
|    └─Sequential: 2-10                  [-1, 256, 16, 16]         --
|    |    └─ConvTranspose2d: 3-11        [-1, 256, 16, 16]         524,544
|    |    └─LeakyReLU: 3-12              [-1, 256, 16, 16]         --
|    └─Sequential: 2-11                  [-1, 128, 32, 32]         --
|    |    └─ConvTranspose2d: 3-13        [-1, 128, 32, 32]         131,200
|    |    └─LeakyReLU: 3-14              [-1, 128, 32, 32]         --
|    └─Sequential: 2-12                  [-1, 64, 64, 64]          --
|    |    └─ConvTranspose2d: 3-15        [-1, 64, 64, 64]          32,832
|    |    └─LeakyReLU: 3-16              [-1, 64, 64, 64]          --
|    └─Conv2d: 2-13                      [-1, 3, 64, 64]           1,731
|    └─Sigmoid: 2-14                     [-1, 3, 64, 64]           --
==========================================================================================
Total params: 53,574,531
Trainable params: 53,574,531
Non-trainable params: 0
Total mult-adds (G): 1.29
```


Original A
Reconstructed A
Reconstructed B
Original B
Reconstructed A
Reconstructed B


lossA
lossB
0.14
0.12
0.10
0.08
0.06
0.04
0.02
Metrics
0
200
400
600
800
1000
Epochs

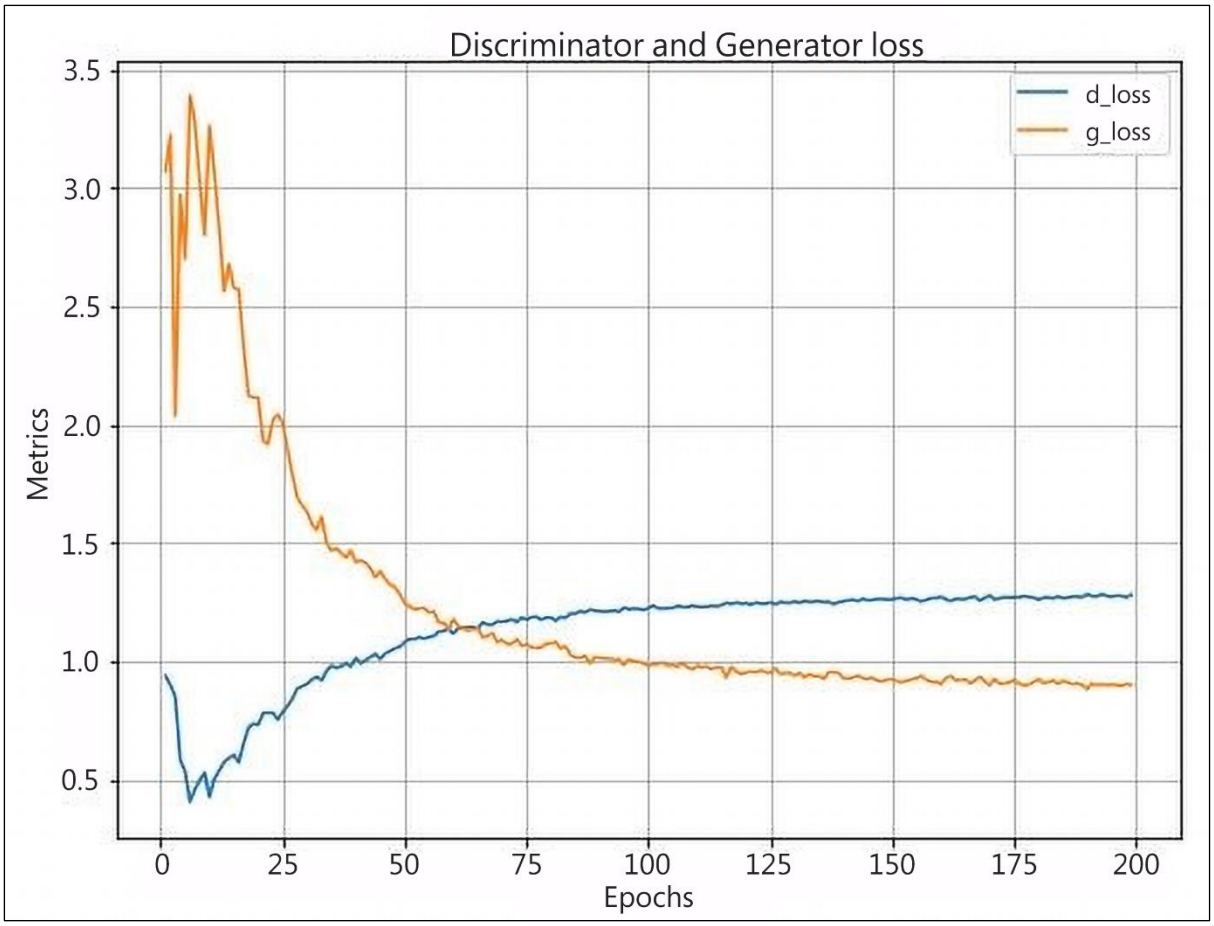
Chapter 12:





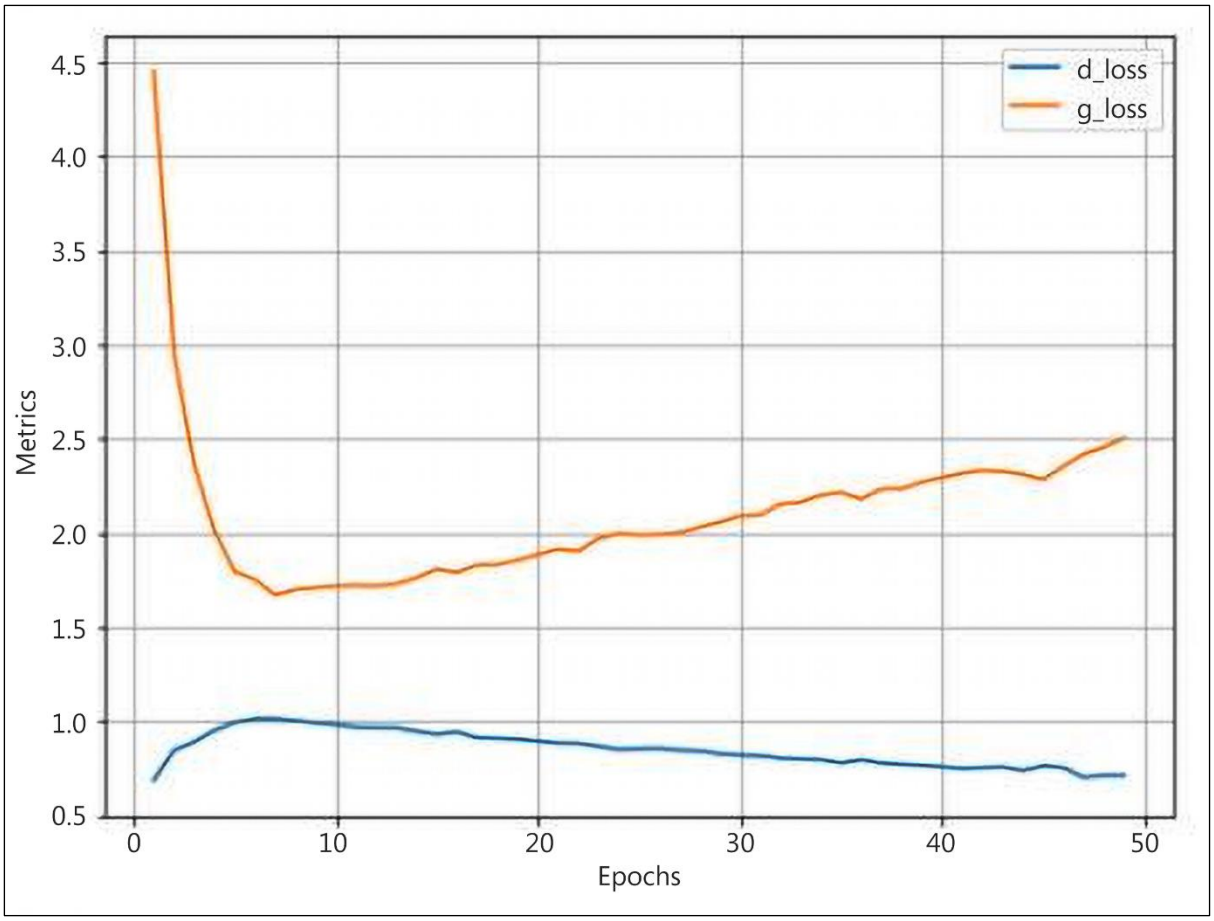

```
==========================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================
├─Sequential: 1-1                        [-1, 1]                   --
|    └─Linear: 2-1                       [-1, 1024]                803,840
|    └─LeakyReLU: 2-2                    [-1, 1024]                --
|    └─Dropout: 2-3                      [-1, 1024]                --
|    └─Linear: 2-4                       [-1, 512]                 524,800
|    └─LeakyReLU: 2-5                    [-1, 512]                 --
|    └─Dropout: 2-6                      [-1, 512]                 --
|    └─Linear: 2-7                       [-1, 256]                 131,328
|    └─LeakyReLU: 2-8                    [-1, 256]                 --
|    └─Dropout: 2-9                      [-1, 256]                 --
|    └─Linear: 2-10                      [-1, 1]                   257
|    └─Sigmoid: 2-11                     [-1, 1]                   --
==========================================================================
Total params: 1,460,225
Trainable params: 1,460,225
Non-trainable params: 0
Total mult-adds (M): 2.92
```

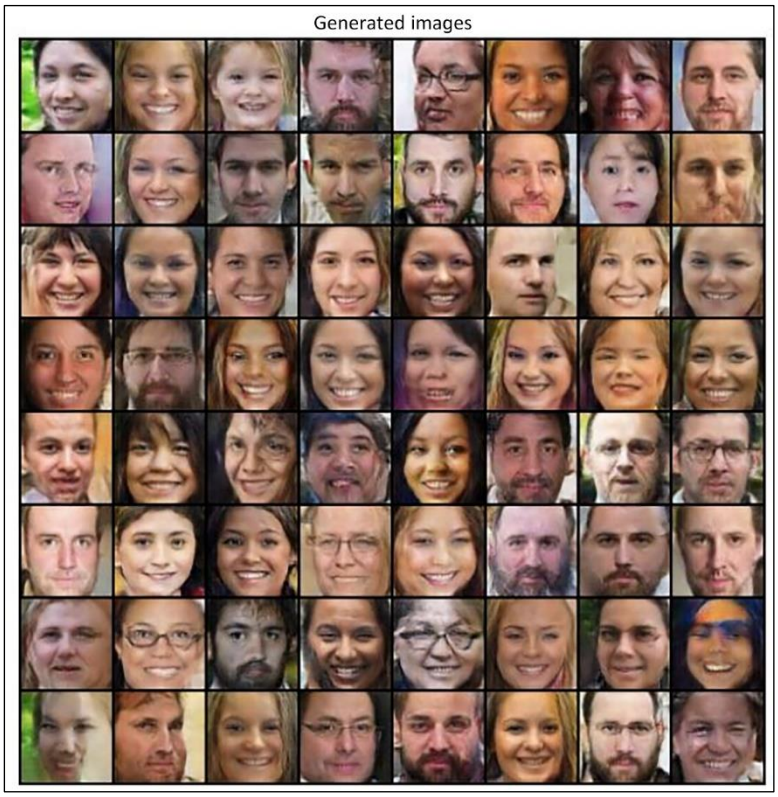
```
==========================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================
├─Sequential: 1-1                        [-1, 784]                 --
|    └─Linear: 2-1                       [-1, 256]                 25,856
|    └─LeakyReLU: 2-2                    [-1, 256]                 --
|    └─Linear: 2-3                       [-1, 512]                 131,584
|    └─LeakyReLU: 2-4                    [-1, 512]                 --
|    └─Linear: 2-5                       [-1, 1024]                525,312
|    └─LeakyReLU: 2-6                    [-1, 1024]                --
|    └─Linear: 2-7                       [-1, 784]                 803,600
|    └─Tanh: 2-8                         [-1, 784]                 --
==========================================================================
Total params: 1,486,352
Trainable params: 1,486,352
Non-trainable params: 0
Total mult-adds (M): 2.97
```



```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
├─Sequential: 1-1                        [-1, 1, 1, 1]             --
|    └─Conv2d: 2-1                       [-1, 64, 32, 32]          3,072
|    └─LeakyReLU: 2-2                    [-1, 64, 32, 32]          --
|    └─Conv2d: 2-3                       [-1, 128, 16, 16]         131,072
|    └─BatchNorm2d: 2-4                  [-1, 128, 16, 16]         256
|    └─LeakyReLU: 2-5                    [-1, 128, 16, 16]         --
|    └─Conv2d: 2-6                       [-1, 256, 8, 8]           524,288
|    └─BatchNorm2d: 2-7                  [-1, 256, 8, 8]           512
|    └─LeakyReLU: 2-8                    [-1, 256, 8, 8]           --
|    └─Conv2d: 2-9                       [-1, 512, 4, 4]           2,097,152
|    └─BatchNorm2d: 2-10                 [-1, 512, 4, 4]           1,024
|    └─LeakyReLU: 2-11                   [-1, 512, 4, 4]           --
|    └─Conv2d: 2-12                      [-1, 1, 1, 1]             8,192
|    └─Sigmoid: 2-13                     [-1, 1, 1, 1]             --
==========================================================================================
Total params: 2,765,568
Trainable params: 2,765,568
Non-trainable params: 0
Total mult-adds (M): 106.58
==========================================================================================
```

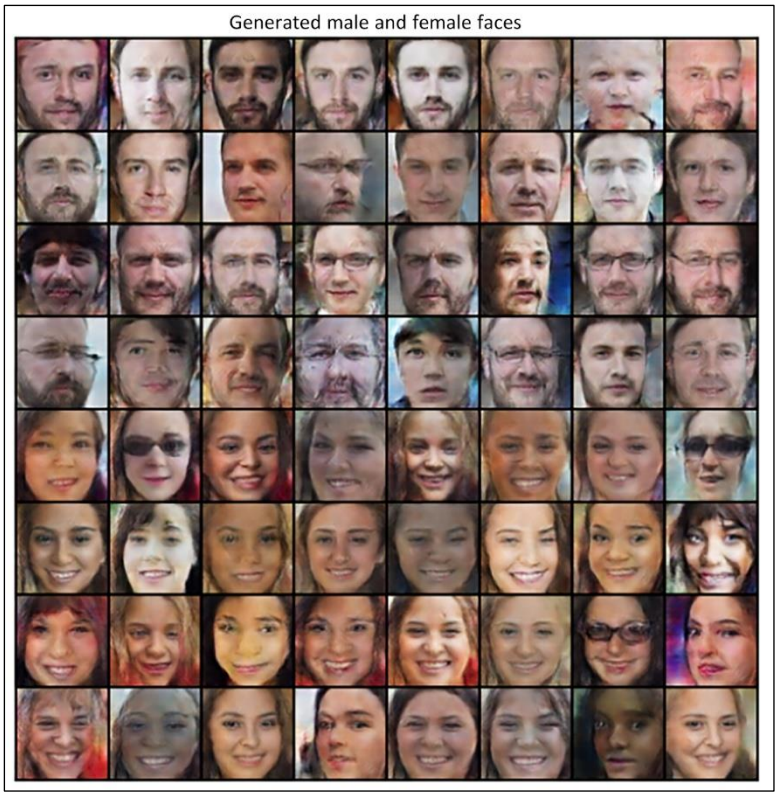
```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
├─Sequential: 1-1                        [-1, 3, 64, 64]           --
|    └─ConvTranspose2d: 2-1              [-1, 512, 4, 4]           819,200
|    └─BatchNorm2d: 2-2                  [-1, 512, 4, 4]           1,024
|    └─ReLU: 2-3                         [-1, 512, 4, 4]           --
|    └─ConvTranspose2d: 2-4              [-1, 256, 8, 8]           2,097,152
|    └─BatchNorm2d: 2-5                  [-1, 256, 8, 8]           512
|    └─ReLU: 2-6                         [-1, 256, 8, 8]           --
|    └─ConvTranspose2d: 2-7              [-1, 128, 16, 16]         524,288
|    └─BatchNorm2d: 2-8                  [-1, 128, 16, 16]         256
|    └─ReLU: 2-9                         [-1, 128, 16, 16]         --
|    └─ConvTranspose2d: 2-10             [-1, 64, 32, 32]          131,072
|    └─BatchNorm2d: 2-11                 [-1, 64, 32, 32]          128
|    └─ReLU: 2-12                        [-1, 64, 32, 32]          --
|    └─ConvTranspose2d: 2-13             [-1, 3, 64, 64]           3,072
|    └─Tanh: 2-14                        [-1, 3, 64, 64]           --
==========================================================================================
Total params: 3,576,704
Trainable params: 3,576,704
Non-trainable params: 0
Total mult-adds (M): 431.92
==========================================================================================
```


d_loss
g_loss
Metrics
Epochs
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0
10
20
30
40
50

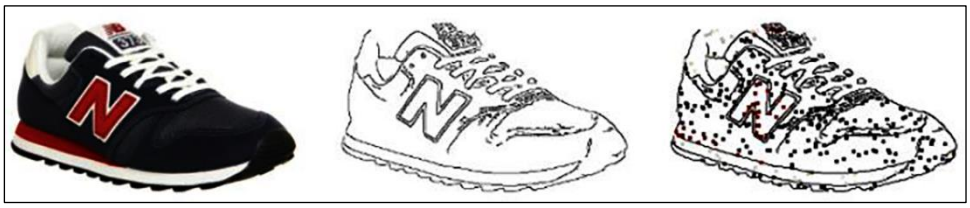

Generated images

```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
├─Sequential: 1-1                        [-1, 256]                 --
│    └─Conv2d: 2-1                       [-1, 64, 32, 32]          3,072
│    └─LeakyReLU: 2-2                    [-1, 64, 32, 32]          --
│    └─Conv2d: 2-3                       [-1, 128, 16, 16]         131,072
│    └─BatchNorm2d: 2-4                  [-1, 128, 16, 16]         256
│    └─LeakyReLU: 2-5                    [-1, 128, 16, 16]         --
│    └─Conv2d: 2-6                       [-1, 256, 8, 8]           524,288
│    └─BatchNorm2d: 2-7                  [-1, 256, 8, 8]           512
│    └─LeakyReLU: 2-8                    [-1, 256, 8, 8]           --
│    └─Conv2d: 2-9                       [-1, 512, 4, 4]           2,097,152
│    └─BatchNorm2d: 2-10                 [-1, 512, 4, 4]           1,024
│    └─LeakyReLU: 2-11                   [-1, 512, 4, 4]           --
│    └─Conv2d: 2-12                      [-1, 64, 2, 2]            524,288
│    └─BatchNorm2d: 2-13                 [-1, 64, 2, 2]            128
│    └─LeakyReLU: 2-14                   [-1, 64, 2, 2]            --
│    └─Flatten: 2-15                     [-1, 256]                 --
├─Embedding: 1-2                         [-1, 32]                  64
├─Sequential: 1-3                        [-1, 1]                   --
│    └─Linear: 2-16                      [-1, 100]                 28,900
│    └─LeakyReLU: 2-17                   [-1, 100]                 --
│    └─Linear: 2-18                      [-1, 1]                   101
│    └─Sigmoid: 2-19                     [-1, 1]                   --
==========================================================================================
Total params: 3,310,857
Trainable params: 3,310,857
Non-trainable params: 0
Total mult-adds (M): 109.25
```

```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
├─Embedding: 1-1                         [-1, 32]                  64
├─Sequential: 1-2                        [-1, 3, 64, 64]           --
│    └─ConvTranspose2d: 2-1              [-1, 512, 4, 4]           1,081,344
│    └─BatchNorm2d: 2-2                  [-1, 512, 4, 4]           1,024
│    └─ReLU: 2-3                         [-1, 512, 4, 4]           --
│    └─ConvTranspose2d: 2-4              [-1, 256, 8, 8]           2,097,152
│    └─BatchNorm2d: 2-5                  [-1, 256, 8, 8]           512
│    └─ReLU: 2-6                         [-1, 256, 8, 8]           --
│    └─ConvTranspose2d: 2-7              [-1, 128, 16, 16]         524,288
│    └─BatchNorm2d: 2-8                  [-1, 128, 16, 16]         256
│    └─ReLU: 2-9                         [-1, 128, 16, 16]         --
│    └─ConvTranspose2d: 2-10             [-1, 64, 32, 32]          131,072
│    └─BatchNorm2d: 2-11                 [-1, 64, 32, 32]          128
│    └─ReLU: 2-12                        [-1, 64, 32, 32]          --
│    └─ConvTranspose2d: 2-13             [-1, 3, 64, 64]           3,072
│    └─Tanh: 2-14                        [-1, 3, 64, 64]           --
==========================================================================================
Total params: 3,838,912
Trainable params: 3,838,912
Non-trainable params: 0
Total mult-adds (M): 436.38
```


Generated male and female faces

Chapter 13:



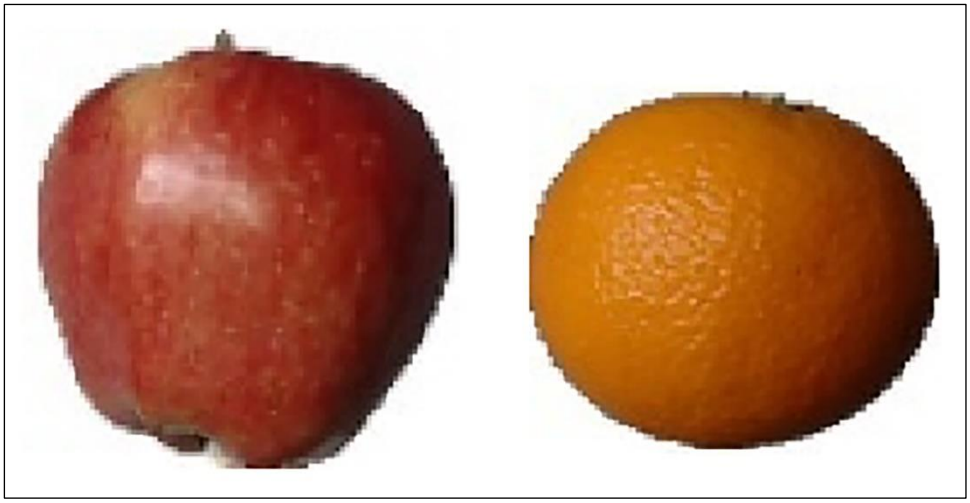
Layer (type)	Output Shape	Param #	Tr. Param #
UNetDown-1	[3, 64, 128, 128]	3,072	3,072
UNetDown-2	[3, 128, 64, 64]	131,072	131,072
UNetDown-3	[3, 256, 32, 32]	524,288	524,288
UNetDown-4	[3, 512, 16, 16]	2,097,152	2,097,152
UNetDown-5	[3, 512, 8, 8]	4,194,304	4,194,304
UNetDown-6	[3, 512, 4, 4]	4,194,304	4,194,304
UNetDown-7	[3, 512, 2, 2]	4,194,304	4,194,304
UNetDown-8	[3, 512, 1, 1]	4,194,304	4,194,304
UNetUp-9	[3, 1024, 2, 2]	4,194,304	4,194,304
UNetUp-10	[3, 1024, 4, 4]	8,388,608	8,388,608
UNetUp-11	[3, 1024, 8, 8]	8,388,608	8,388,608
UNetUp-12	[3, 1024, 16, 16]	8,388,608	8,388,608
UNetUp-13	[3, 512, 32, 32]	4,194,304	4,194,304
UNetUp-14	[3, 256, 64, 64]	1,048,576	1,048,576
UNetUp-15	[3, 128, 128, 128]	262,144	262,144
Upsample-16	[3, 128, 256, 256]	0	0
ZeroPad2d-17	[3, 128, 257, 257]	0	0
Conv2d-18	[3, 3, 256, 256]	6,147	6,147
Tanh-19	[3, 3, 256, 256]	0	0

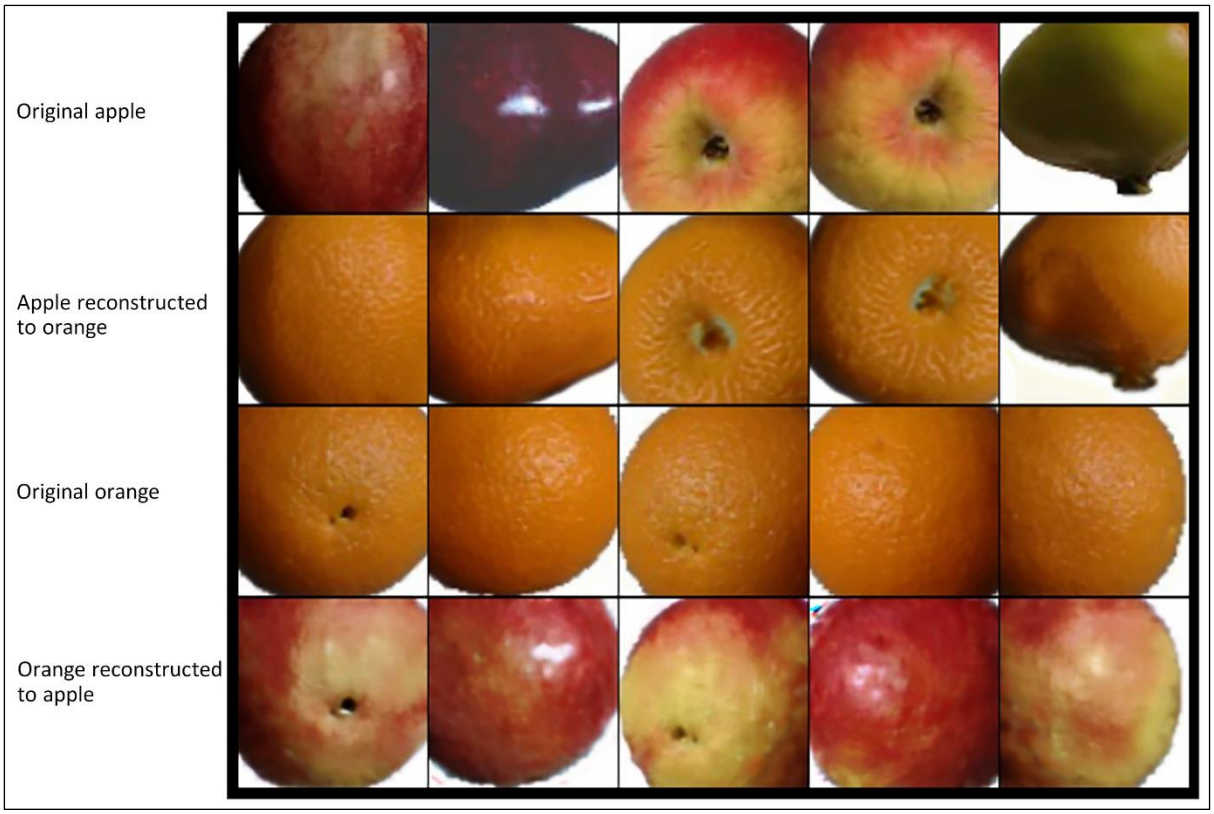
Total params: 54,404,099
Trainable params: 54,404,099
Non-trainable params: 0

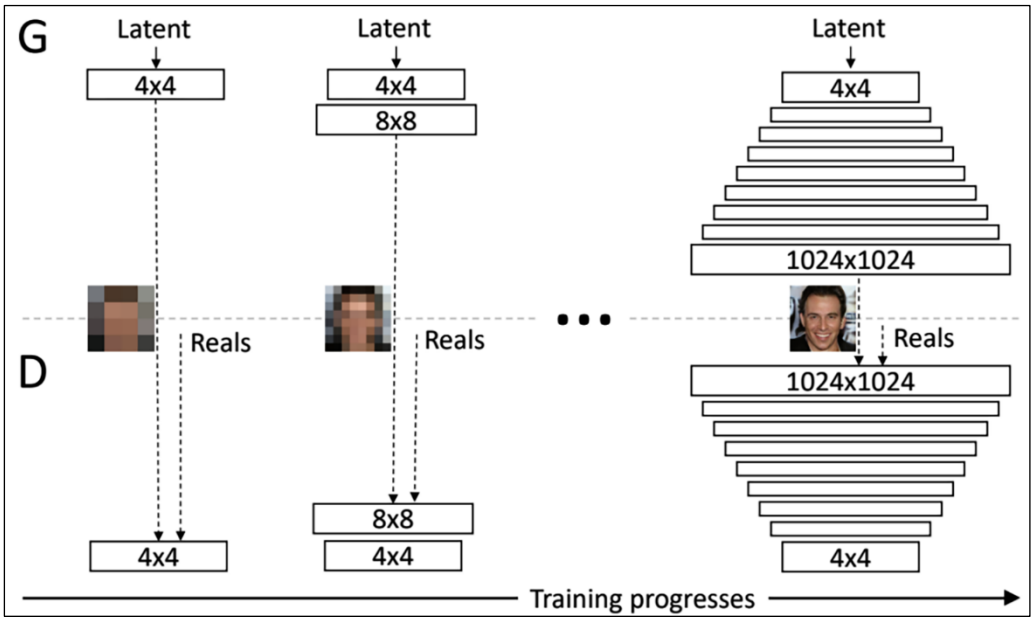
Layer (type)	Output Shape	Param #	Tr. Param #
Conv2d-1	[3, 64, 128, 128]	6,208	6,208
LeakyReLU-2	[3, 64, 128, 128]	0	0
Conv2d-3	[3, 128, 64, 64]	131,200	131,200
InstanceNorm2d-4	[3, 128, 64, 64]	0	0
LeakyReLU-5	[3, 128, 64, 64]	0	0
Conv2d-6	[3, 256, 32, 32]	524,544	524,544
InstanceNorm2d-7	[3, 256, 32, 32]	0	0
LeakyReLU-8	[3, 256, 32, 32]	0	0
Conv2d-9	[3, 512, 16, 16]	2,097,664	2,097,664
InstanceNorm2d-10	[3, 512, 16, 16]	0	0
LeakyReLU-11	[3, 512, 16, 16]	0	0
ZeroPad2d-12	[3, 512, 17, 17]	0	0
Conv2d-13	[3, 1, 16, 16]	8,192	8,192

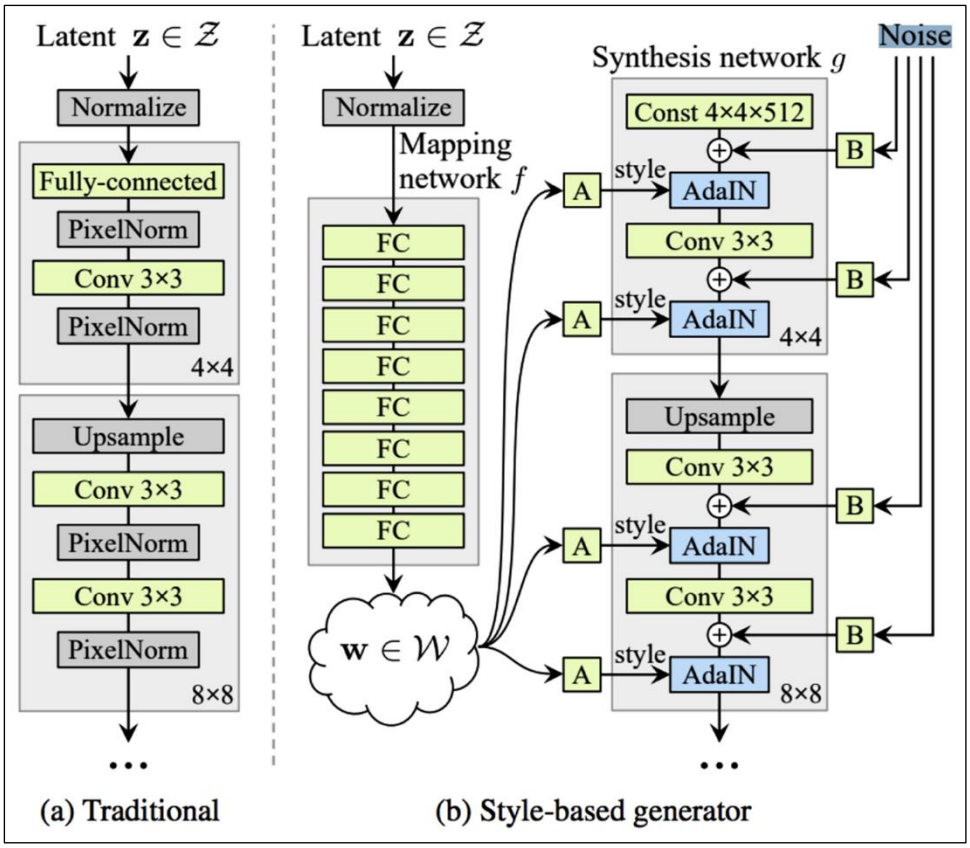
Total params: 2,767,808
Trainable params: 2,767,808
Non-trainable params: 0


Source::Generated::GroundTruth

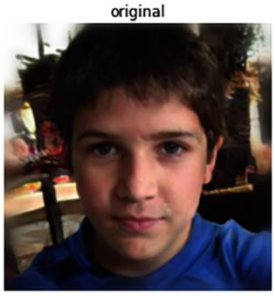



Original apple
Apple reconstructed to orange
Original orange
Orange reconstructed to apple


G
Latent
4x4
Latent
4x4
8x8
Latent
4x4
1024x1024
D
Reals
Reals
Reals
1024x1024
8x8
4x4
4x4
4x4
Training progresses

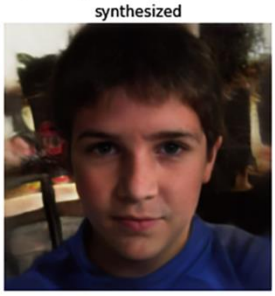

Latent $\mathbf{z} \in \mathcal{Z}$
Normalize
Fully-connected
PixelNorm
Conv 3×3
PixelNorm
4×4
Upsample
Conv 3×3
PixelNorm
Conv 3×3
PixelNorm
8×8
(a) Traditional
Latent $\mathbf{z} \in \mathcal{Z}$
Normalize
Mapping network f
FC
FC
FC
FC
FC
FC
FC
FC
$\mathbf{w} \in \mathcal{W}$
Synthesis network g
Noise
Const 4×4×512
B
A
style
AdaIN
Conv 3×3
4×4
Upsample
8×8
(b) Style-based generator

original

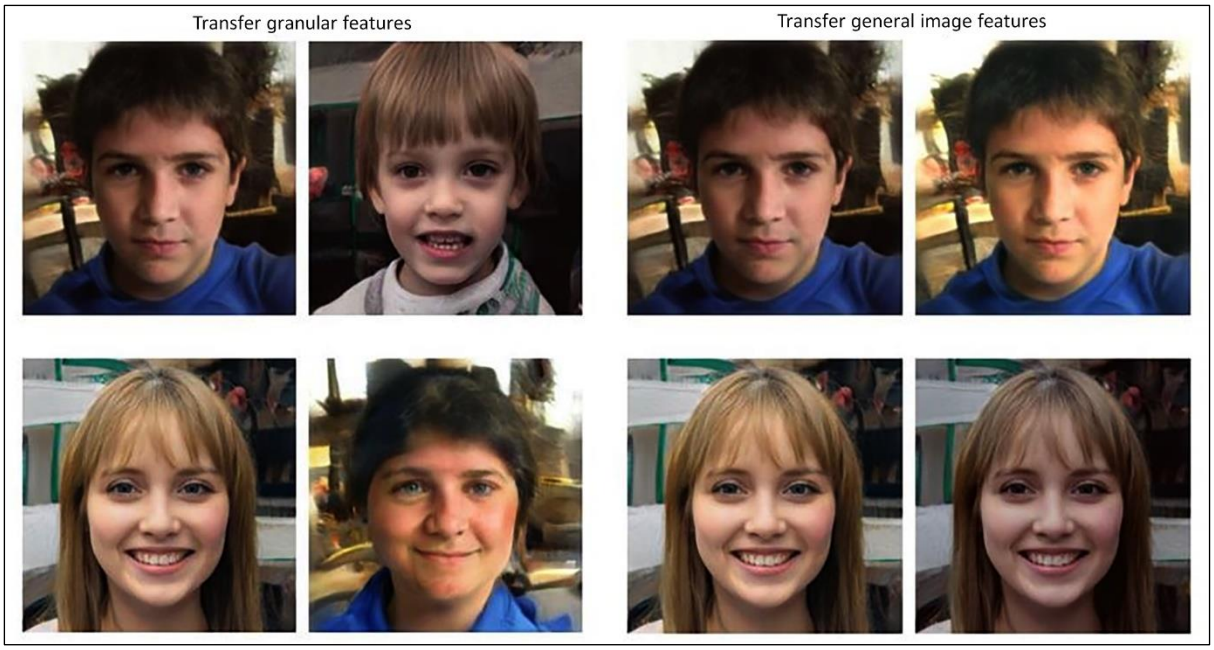


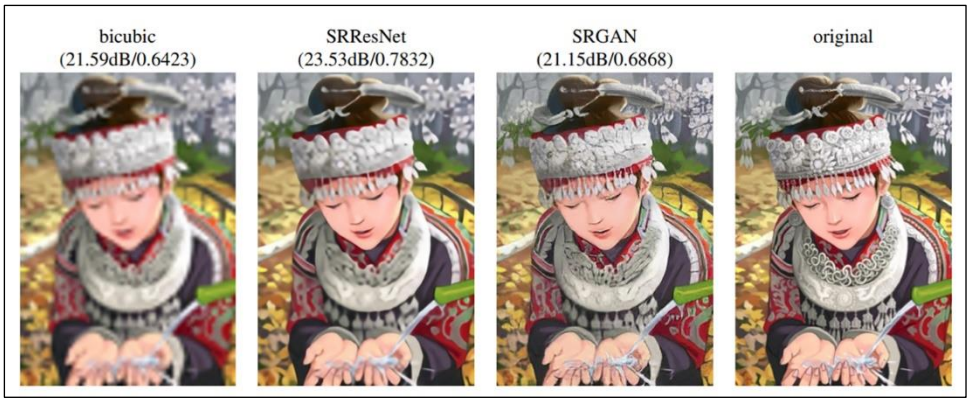
```
Optimizing Latents.
[2020-10-13 04:54:26,699][INFO] Loading pytorch model from `InterFaceGAN/models/pretrain/stylegan_ffhq.pth`.
[2020-10-13 04:54:26,982][INFO] Successfully loaded!
[2020-10-13 04:54:26,983][INFO]   `lod` of the loaded model is 0.0.
Step: 999, Loss: 2.4222376346588135: 100% 1000/1000 [02:06<00:00,  7.89it/s]
```

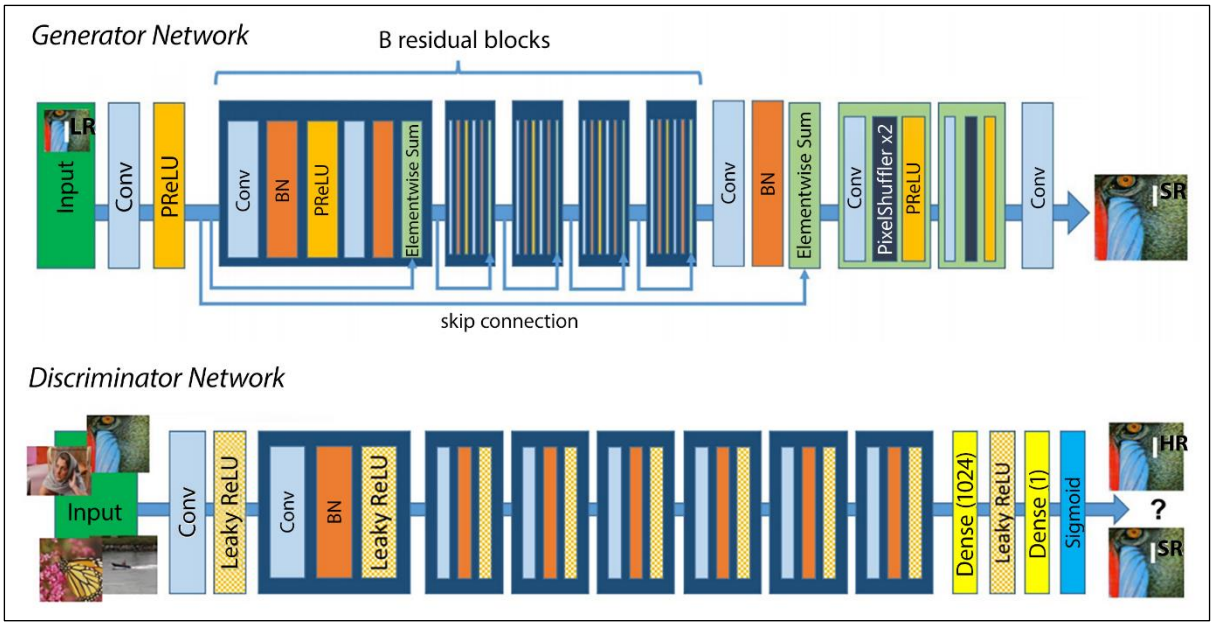
synthesized



Transfer high level features


Transfer granular features
Transfer general image features


bicubic
(21.59dB/0.6423)
SRResNet
(23.53dB/0.7832)
SRGAN
(21.15dB/0.6868)
original

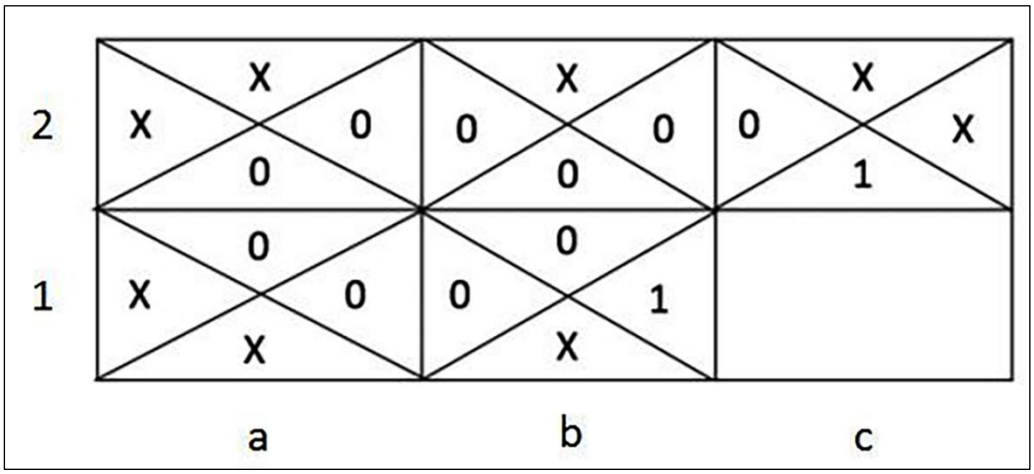

Generator Network
B residual blocks
Input
LR
Conv
PReLU
Conv
BN
PReLU
Elementwise Sum
Conv
BN
Elementwise Sum
Conv
PixelShuffler x2
PReLU
Conv
SR
skip connection
Discriminator Network
Input
Conv
Leaky ReLU
Conv
BN
Leaky ReLU
Dense (1024)
Leaky ReLU
Dense (1)
Sigmoid
HR
?
SR

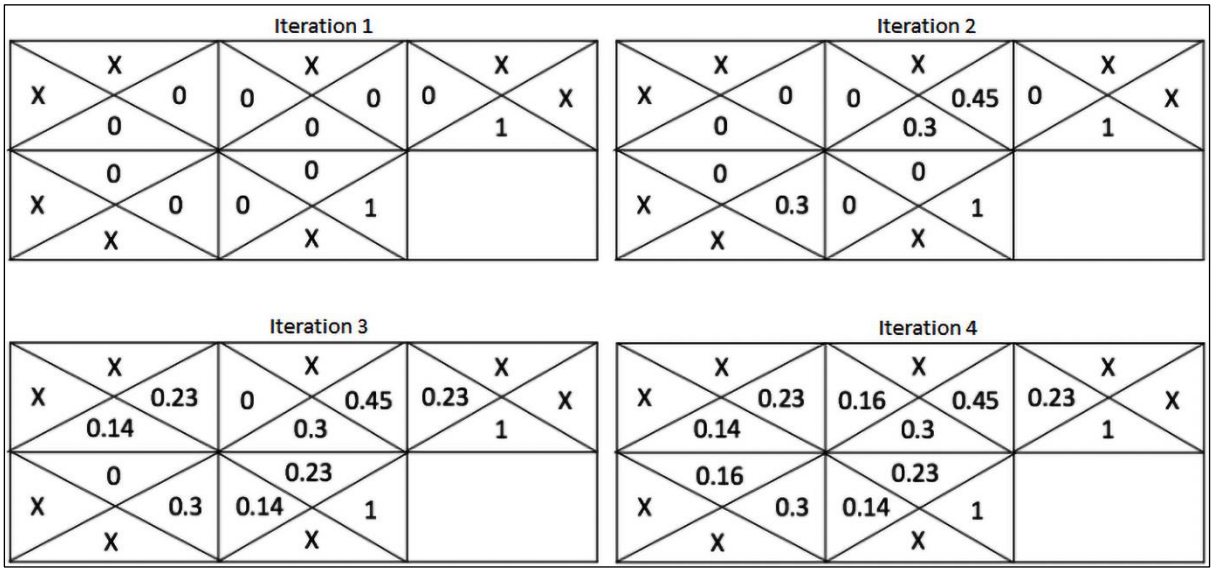

Original image
High resolution image

Chapter 14:

Start		
		+1

0.81	0.9	1
0.9	1	

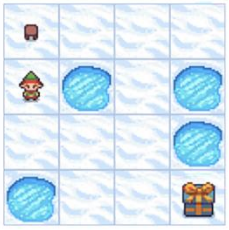




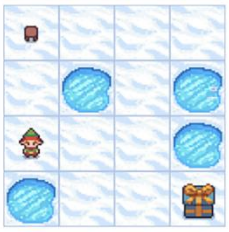
```
[[0.531441   0.59049    0.59049    0.531441  ]
 [0.531441   0.         0.6561     0.59049   ]
 [0.59049    0.729      0.59049    0.6561    ]
 [0.6561     0.         0.59049    0.59049   ]
 [0.59049    0.6561     0.         0.531441  ]
 [0.         0.         0.         0.        ]
 [0.         0.81       0.         0.6561    ]
 [0.         0.         0.         0.        ]
 [0.6561     0.         0.729      0.59049   ]
 [0.6561     0.81       0.81       0.        ]
 [0.729      0.9        0.         0.729     ]
 [0.         0.         0.         0.        ]
 [0.         0.         0.         0.        ]
 [0.         0.81       0.9        0.72899998]
 [0.80999997 0.9        1.         0.81      ]
 [0.         0.         0.         0.        ]]
```

```
----------------------
Episode 0
```

```
1
```



```
1
```

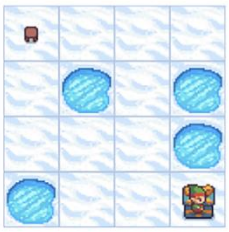


```
2
```

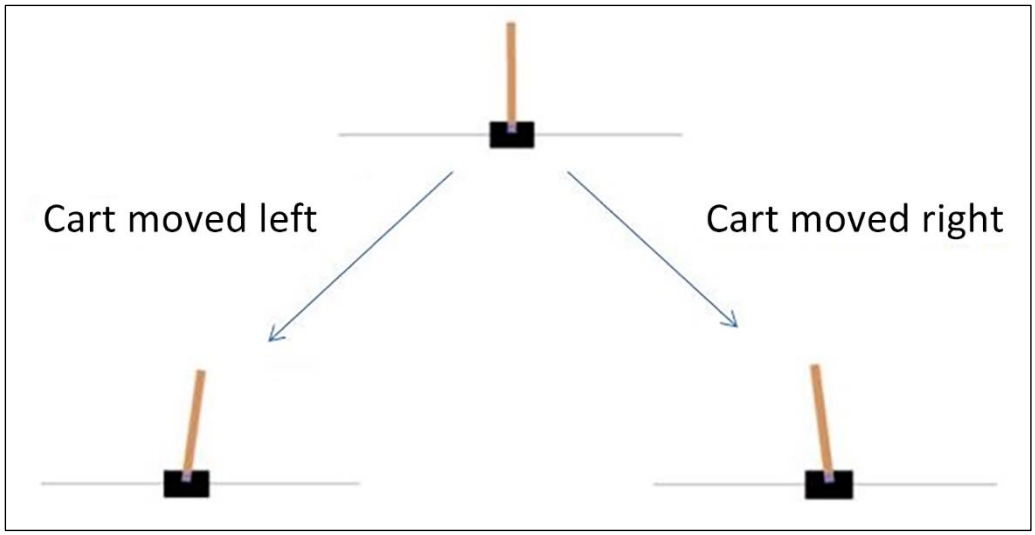
```
1
```

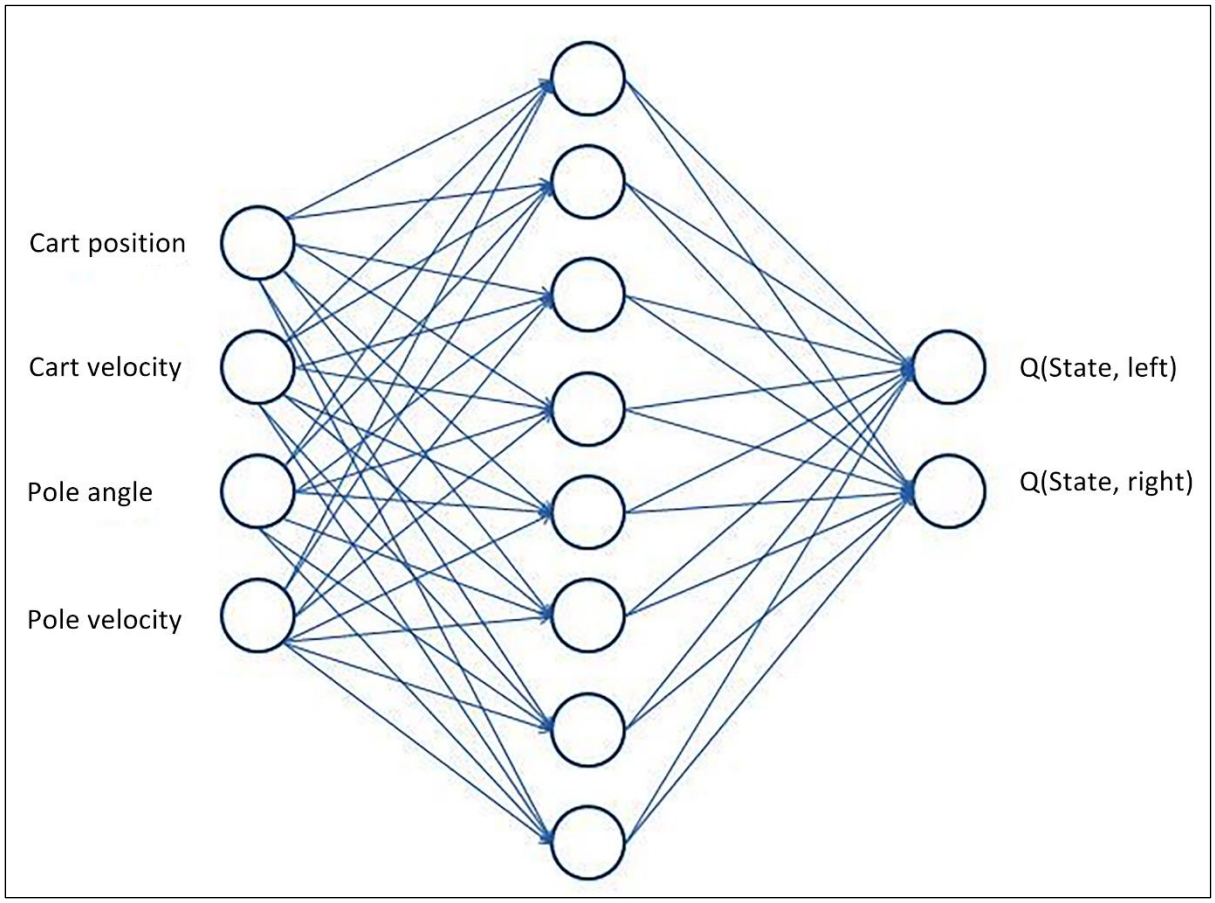
```
2
```

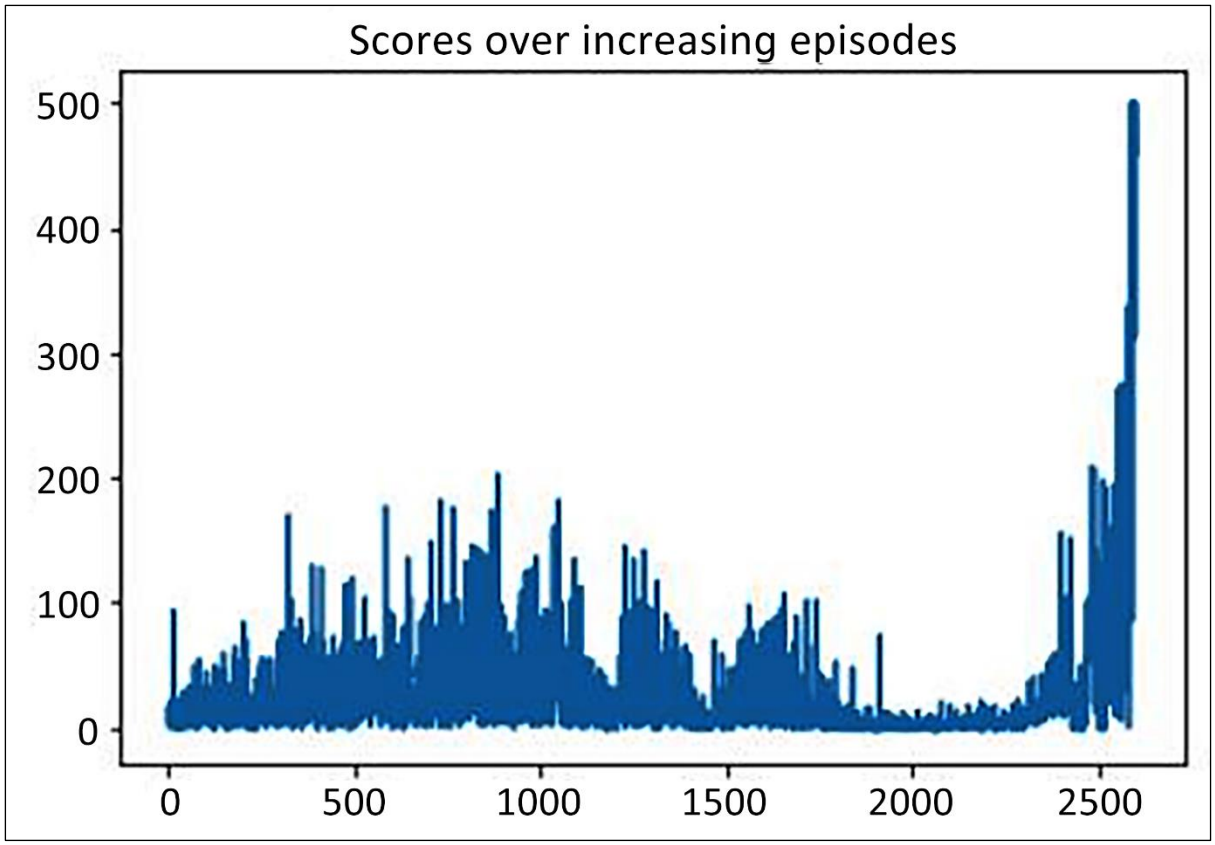
```
2
```

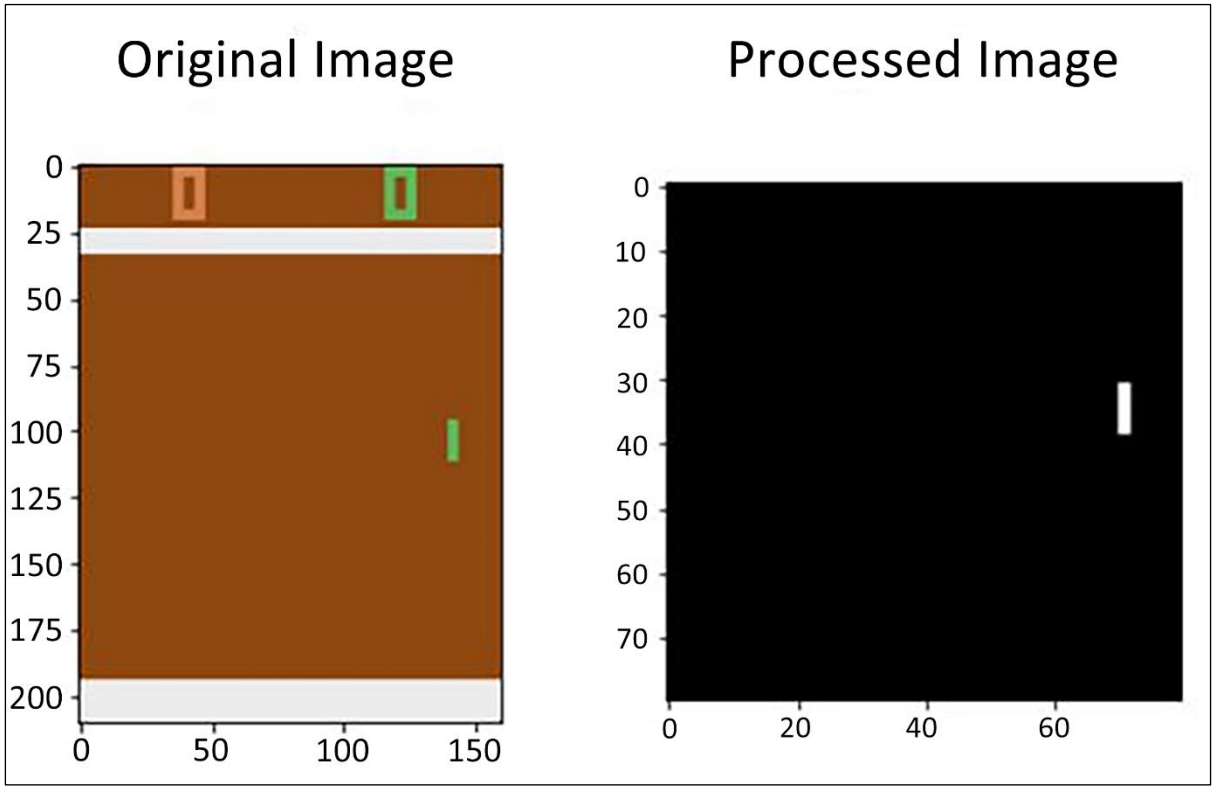


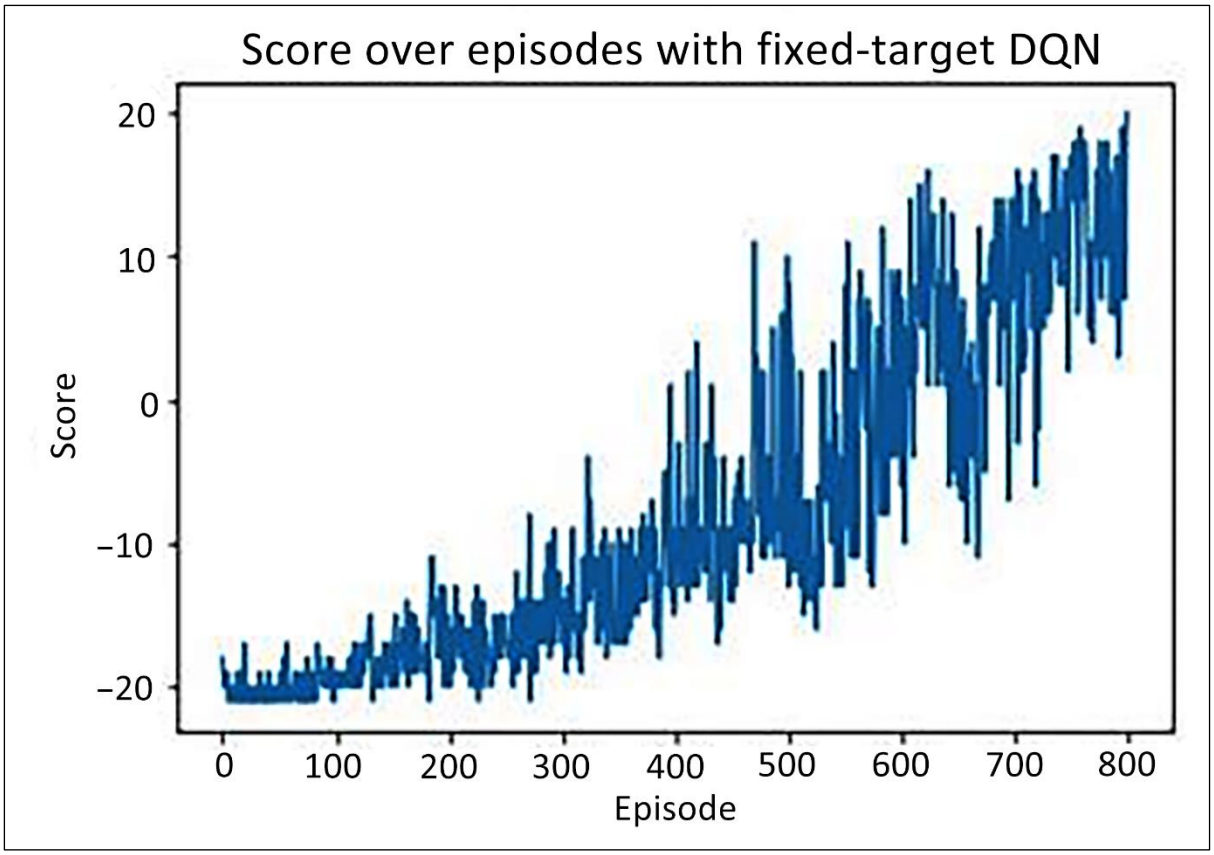
```
Number of steps 6
```


Cart moved left
Cart moved right

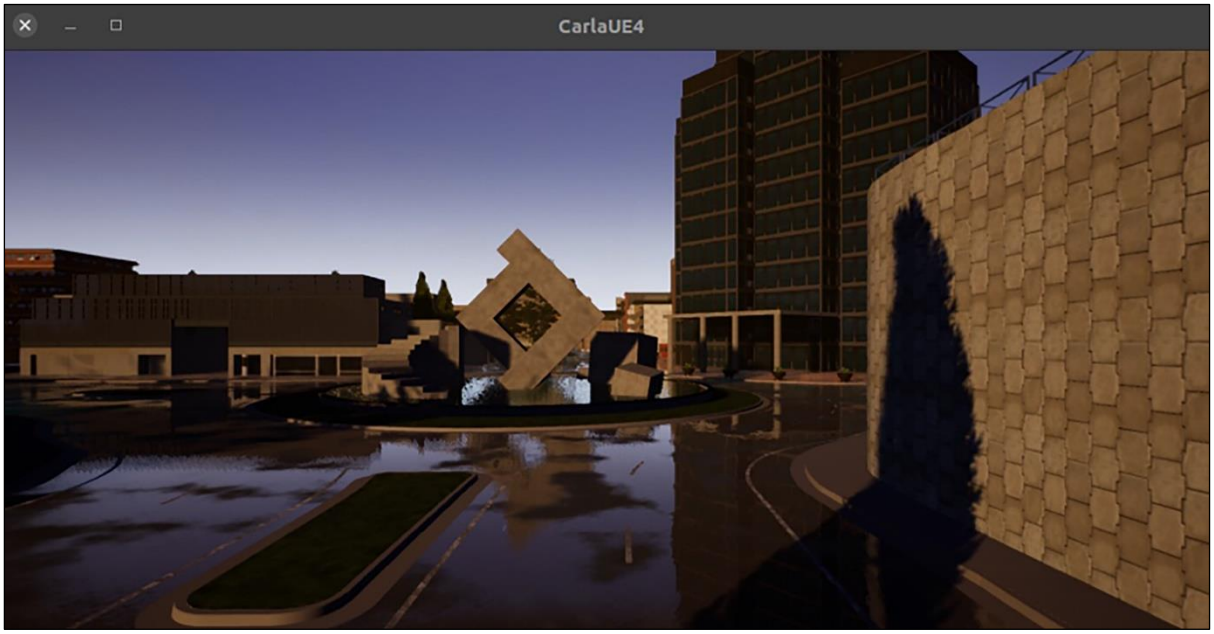

Cart position
Cart velocity
Pole angle
Pole velocity
Q(State, left)
Q(State, right)

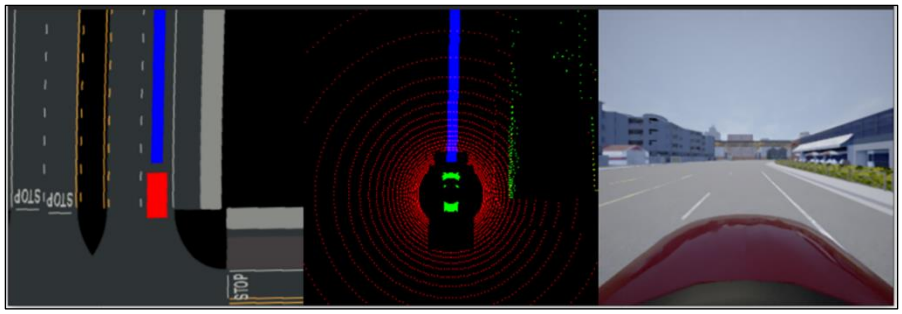

Scores over increasing episodes
500
400
300
200
100
0
0
500
1000
1500
2000
2500


Original Image
Processed Image
0
25
50
75
100
125
150
175
200
0
50
100
150
0
10
20
30
40
50
60
70
0
20
40
60

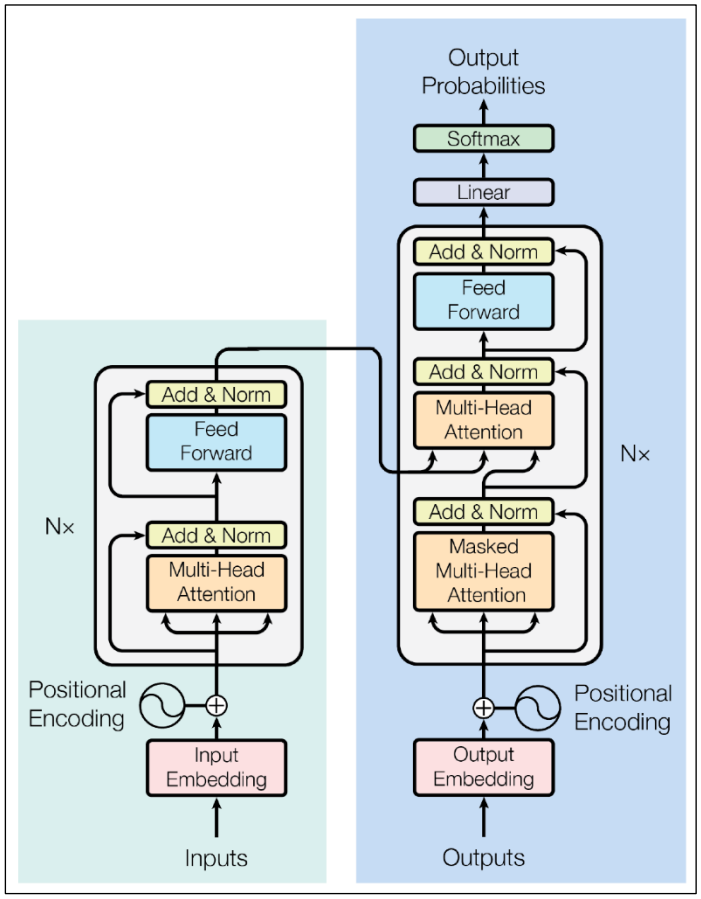


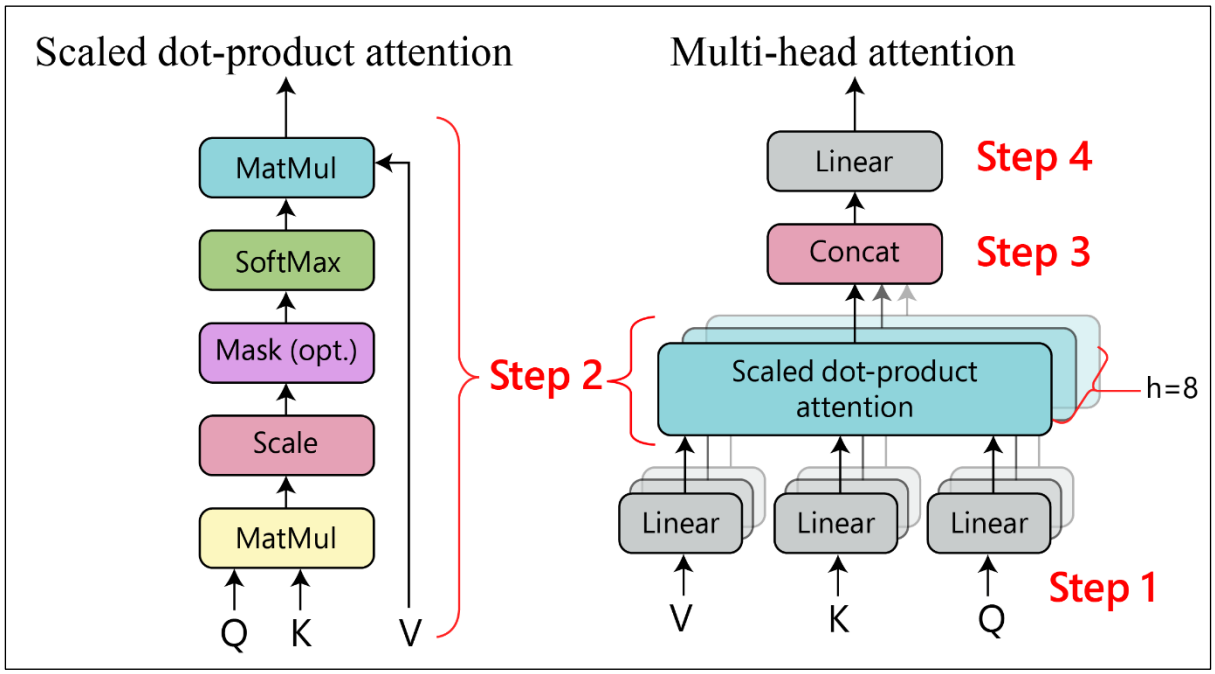

```
└─ ipython -c "import carla; carla.__spec__"                    ─┘
Out[1]: ModuleSpec(name='carla', loader=<_frozen_importlib_external.SourceFileLo
ader object at 0x7fbb31646590>, origin='/home/yyr/anaconda3/lib/python3.7/site-p
ackages/carla-0.9.6-py3.5-linux-x86_64.egg/carla/__init__.py', submodule_search_
locations=['/home/yyr/anaconda3/lib/python3.7/site-packages/carla-0.9.6-py3.5-li
nux-x86_64.egg/carla'])
```

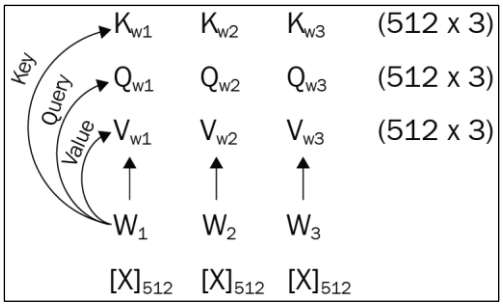



STOP
STOP
STOP

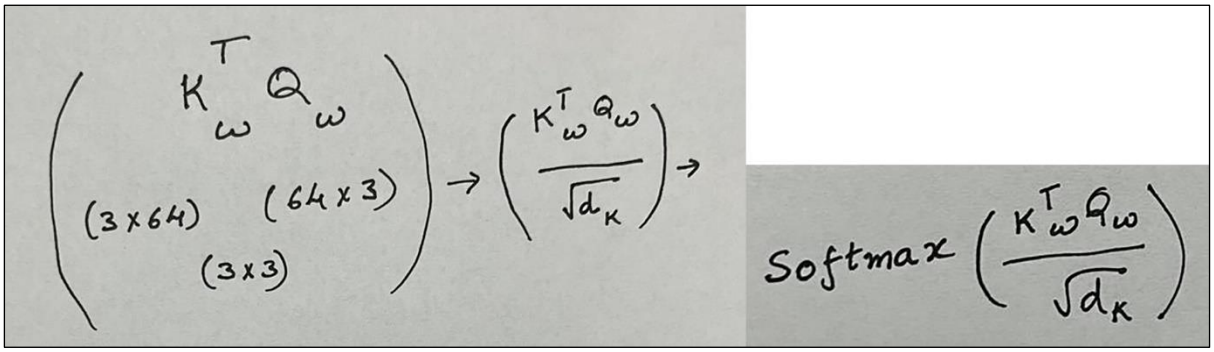
Chapter 15:

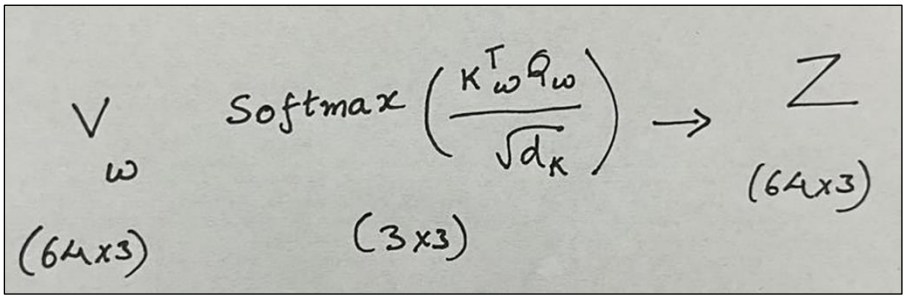

Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Positional Encoding
Output Embedding
Outputs
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Input Embedding
Inputs

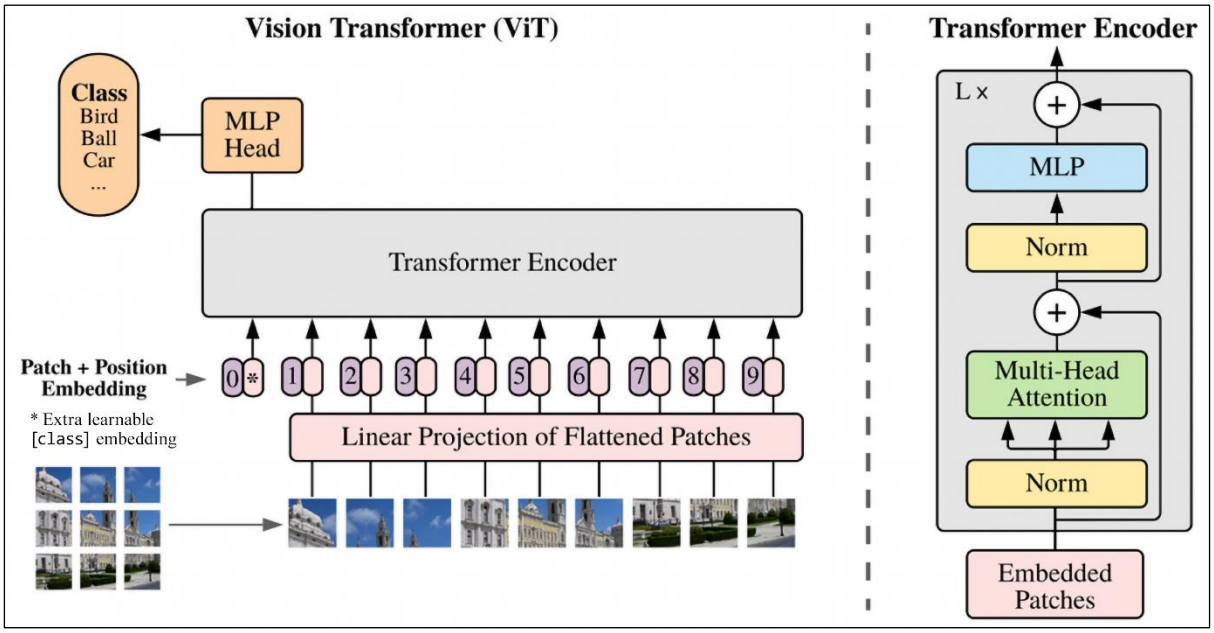

Scaled dot-product attention
MatMul
SoftMax
Mask (opt.)
Scale
MatMul
Q
K
V
Step 2
Multi-head attention
Linear
Step 4
Concat
Step 3
Scaled dot-product attention
h=8
Linear
Linear
Linear
V
K
Q
Step 1

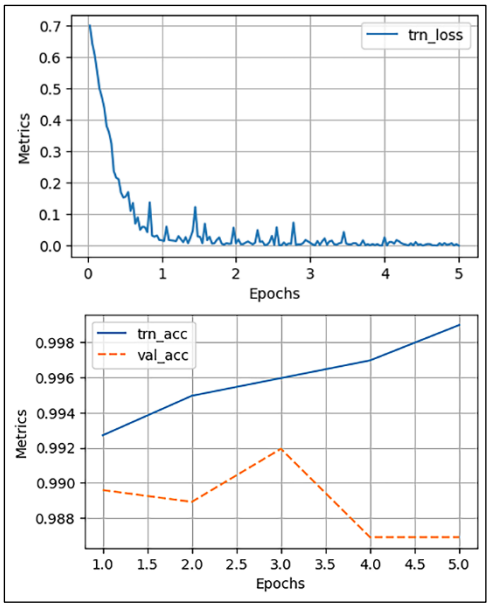


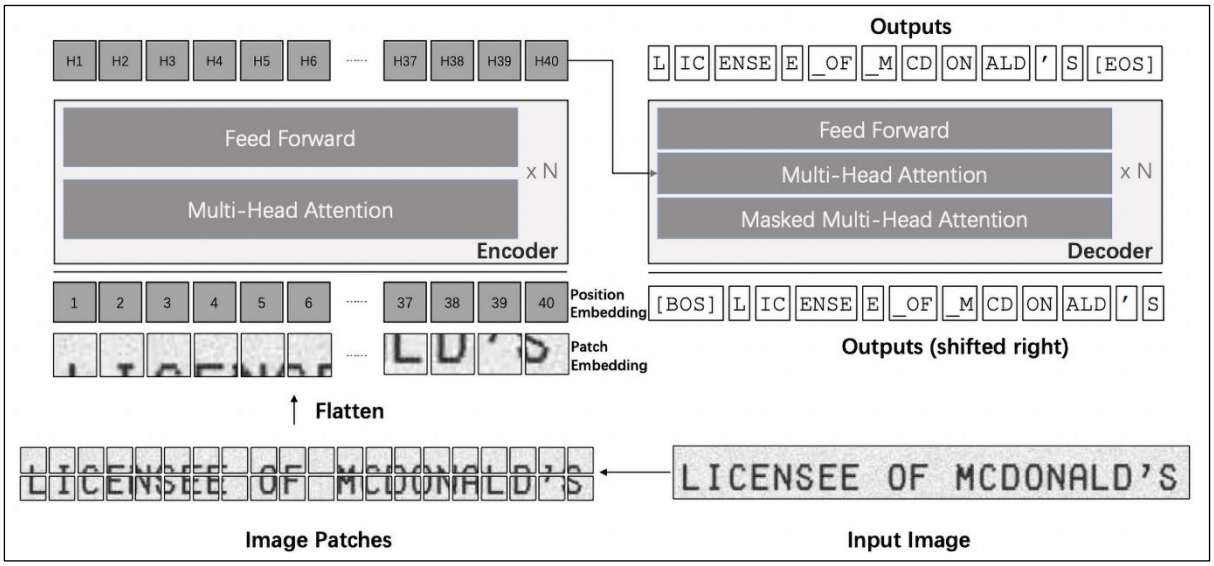
K_{w11}	K_{w21}	K_{w31}	(64 x 3) = K_w
Q_{w11}	Q_{w21}	Q_{w31}	(64 x 3) = Q_w
V_{w11}	V_{w21}	V_{w31}	(64 x 3) = V_w

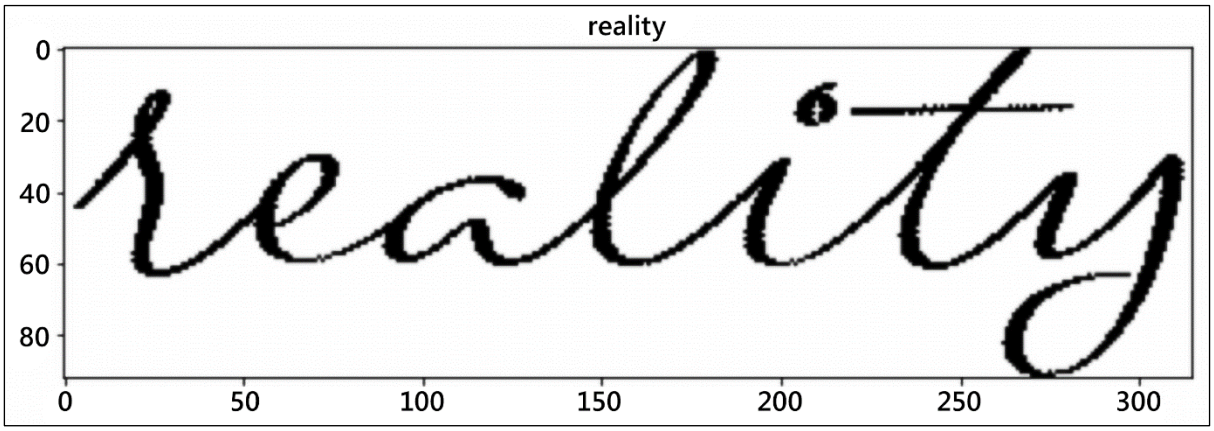







trn_loss
Metrics
Epochs
trn_acc
val_acc
Metrics
Epochs

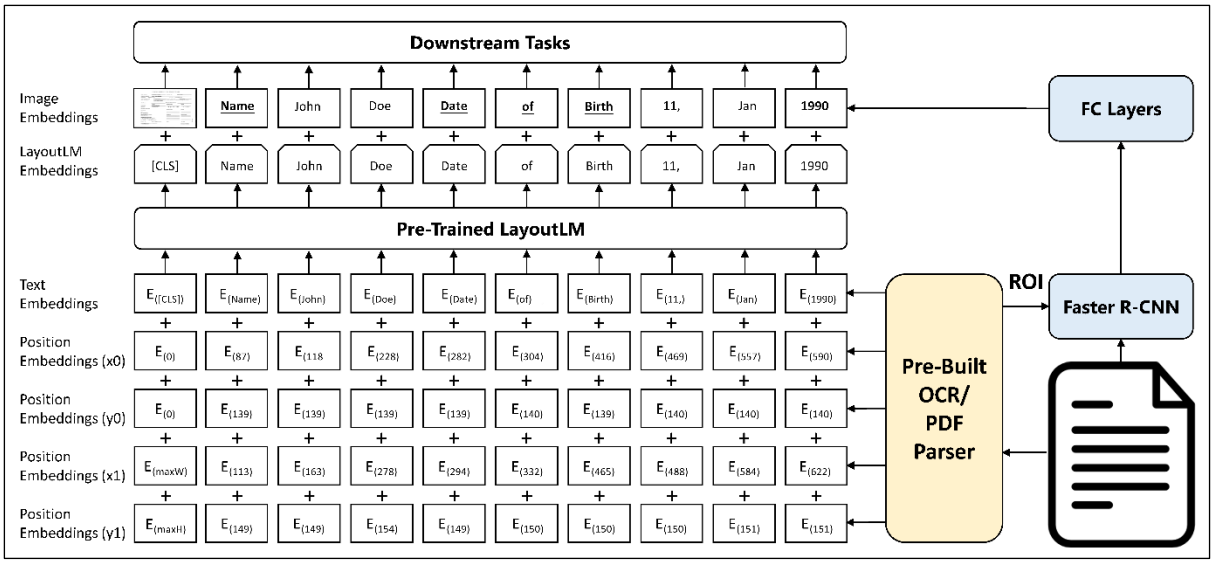

Outputs
H1 H2 H3 H4 H5 H6 ..... H37 H38 H39 H40
L IC ENSE E _OF _M CD ON ALD ' S [EOS]
Feed Forward
Multi-Head Attention
x N
Encoder
Feed Forward
Multi-Head Attention
Masked Multi-Head Attention
x N
Decoder
1 2 3 4 5 6 ..... 37 38 39 40
Position Embedding
[BOS] L IC ENSE E _OF _M CD ON ALD ' S
Patch Embedding
Outputs (shifted right)
Flatten
LICENSEE OF MCDONALD'S
Image Patches
Input Image

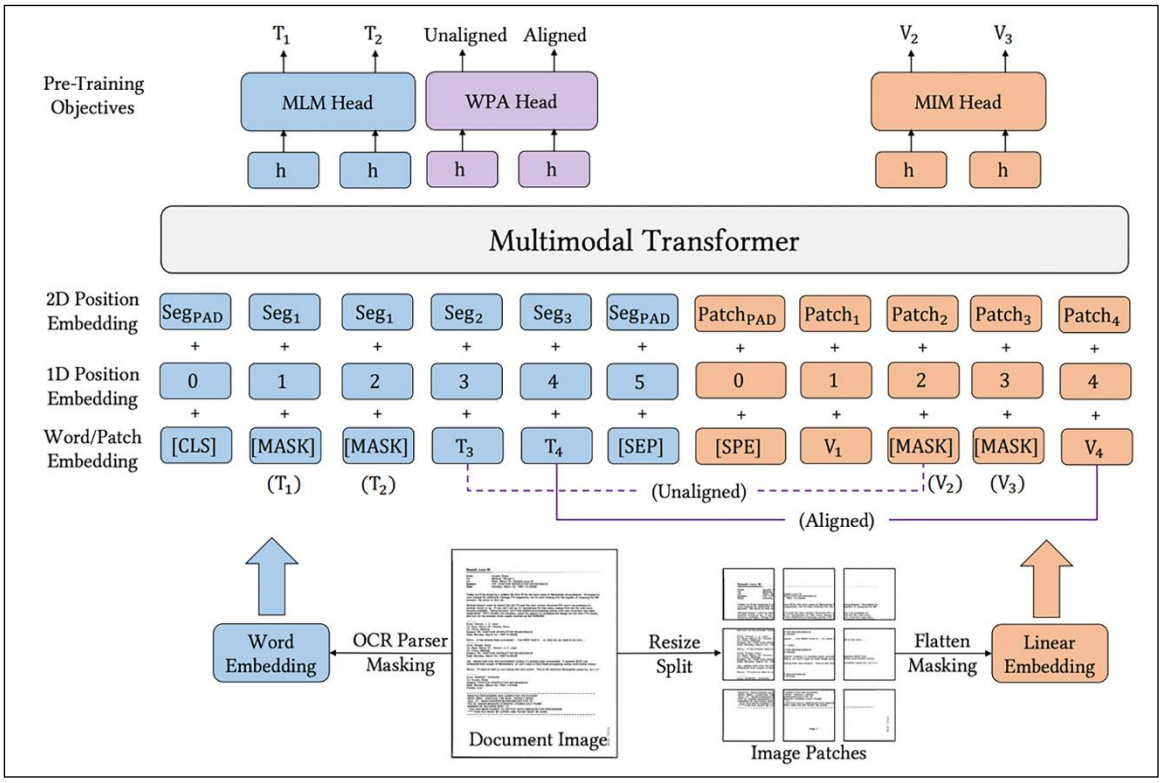

reality

```
['synthetic-data/already@dPkHBT.png',
 'synthetic-data/bring@OzNTFr.png',
 'synthetic-data/few@EhVOvU.png',
 'synthetic-data/research@cgo7rI.png',
 'synthetic-data/fast@bQ8Wkm.png']
```

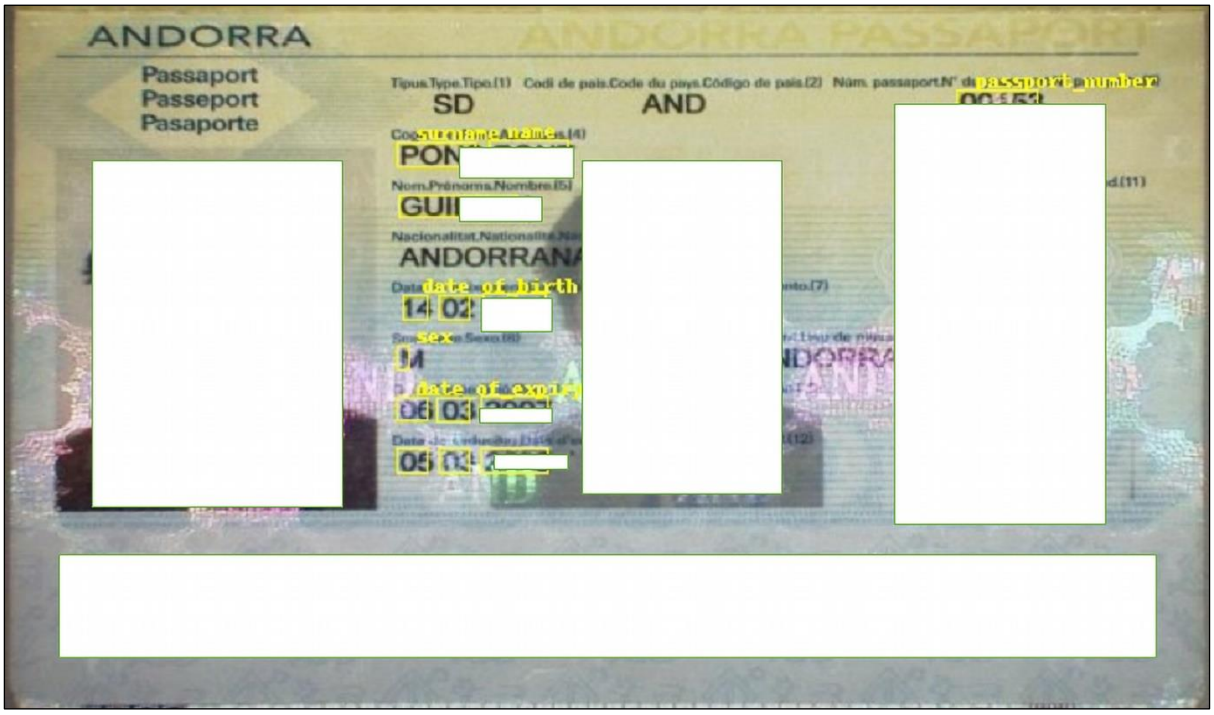
Step	Training Loss	Validation Loss	Cer
100	1.468400	1.485709	0.248651
200	1.201000	1.186254	0.245412
300	1.375000	1.082842	0.106513
400	1.034500	1.100040	0.167686
500	1.183800	1.077437	0.183879
600	0.777200	0.947086	0.169845
700	0.913500	1.018770	0.154372
800	0.759700	0.909072	0.077726
900	0.965700	0.894016	0.082044
1000	0.940400	0.830890	0.065491
1100	0.811900	0.762985	0.032746
1200	0.965600	0.873897	0.087801
1300	0.729500	0.851937	0.077726
1400	0.931800	0.811440	0.066931
1500	0.759000	0.789583	0.052177
1600	0.665700	0.761649	0.064772
1700	0.625800	0.721781	0.038143
1800	0.660300	0.713733	0.041742
1900	0.695700	0.720314	0.033105
2000	0.595400	0.661299	0.017992

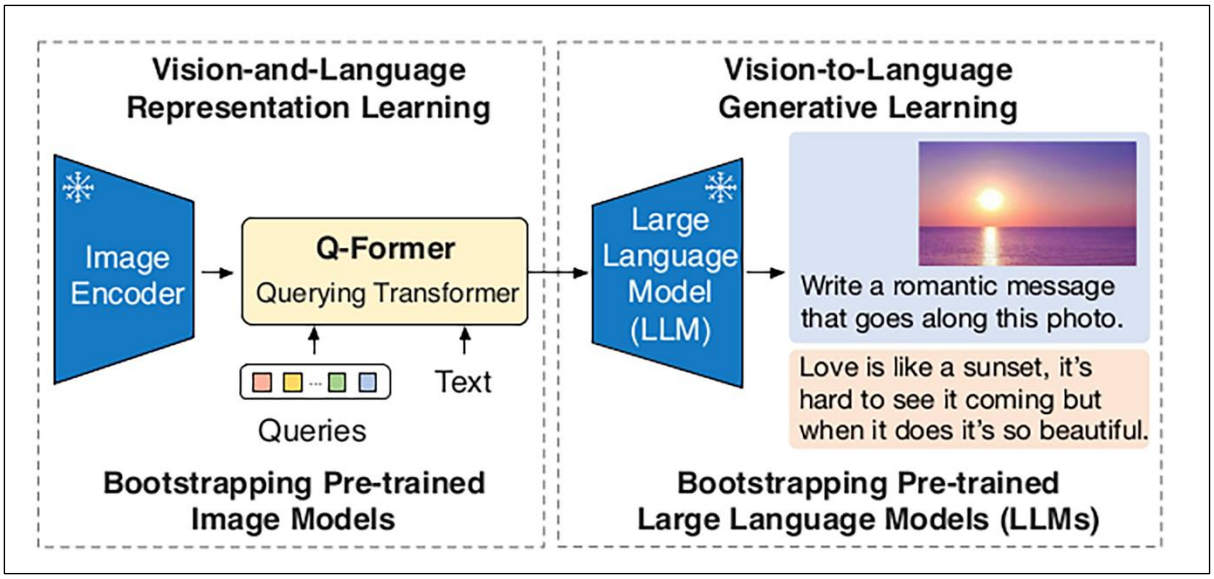

AMERICAN
Predicted Text: American

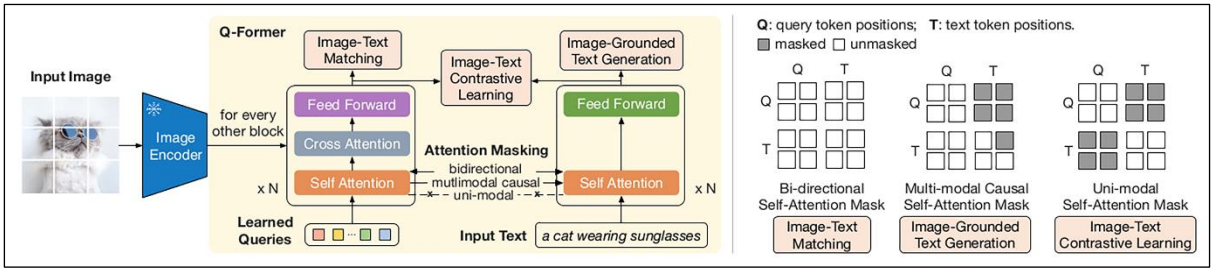

Downstream Tasks
Image Embeddings
LayoutLM Embeddings
Pre-Trained LayoutLM
Text Embeddings
Position Embeddings (x0)
Position Embeddings (y0)
Position Embeddings (x1)
Position Embeddings (y1)
FC Layers
ROI
Faster R-CNN
Pre-Built OCR/ PDF Parser

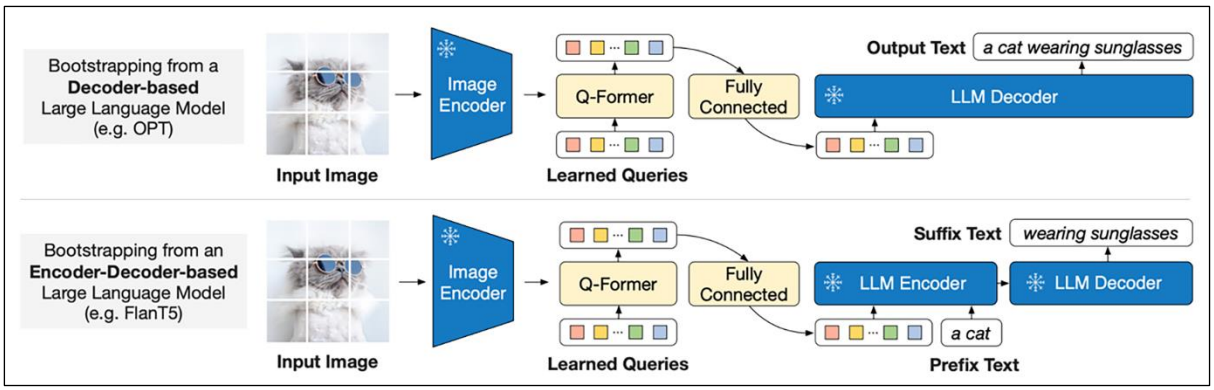

Pre-Training Objectives
MLM Head
WPA Head
MIM Head
Multimodal Transformer
2D Position Embedding
1D Position Embedding
Word/Patch Embedding
(Unaligned)
(Aligned)
Word Embedding
OCR Parser
Masking
Document Image
Resize
Split
Image Patches
Flatten
Masking
Linear Embedding

```
image <PIL.PngImagePlugin.PngImageFile image mode=RGB size=600x350 at 0x7C199E43E230>
label_string ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O'
words ['BELGIE', 'BELGIQUE', 'BELGIEN', 'BELGIUM', 'Type', '/', 'Type', 'Typ', '/', 'Ty
labels [5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5
boxes [[148, 37, 268, 80], [300, 35, 473, 80], [505, 35, 650, 80], [686, 35, 833, 77],
```

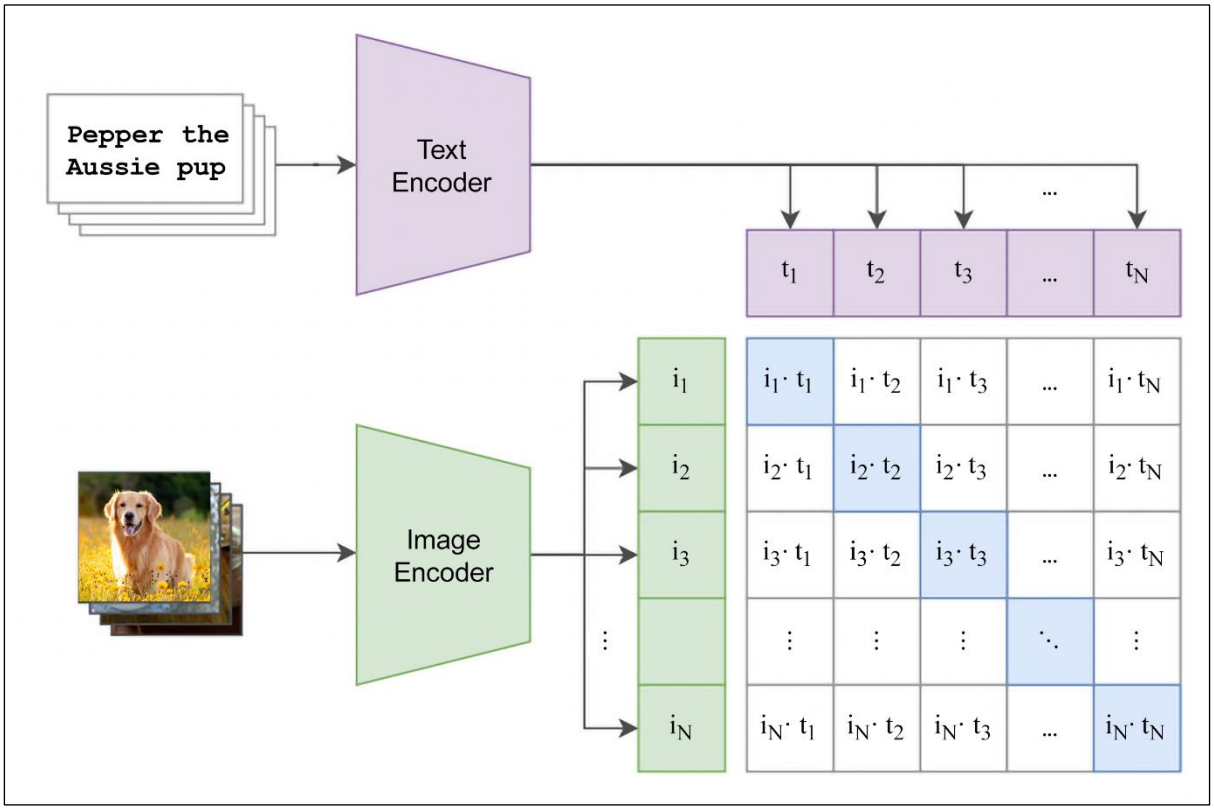



Vision-and-Language Representation Learning
Vision-to-Language Generative Learning
Image Encoder
Q-Former
Querying Transformer
Queries
Text
Large Language Model (LLM)
Write a romantic message that goes along this photo.
Love is like a sunset, it's hard to see it coming but when it does it's so beautiful.
Bootstrapping Pre-trained Image Models
Bootstrapping Pre-trained Large Language Models (LLMs)

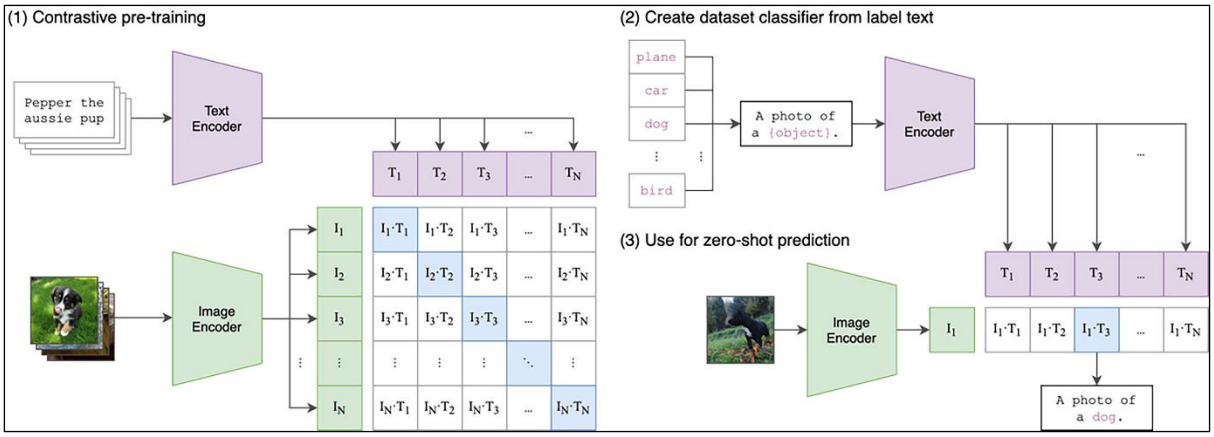

Input Image
Image Encoder
Q-Former
for every other block
Feed Forward
Cross Attention
Self Attention
x N
Learned Queries
Image-Text Matching
Image-Text Contrastive Learning
Image-Grounded Text Generation
Attention Masking
bidirectional
mutlimodal causal
uni-modal
Input Text
a cat wearing sunglasses
Q: query token positions; T: text token positions.
masked unmasked
Bi-directional Self-Attention Mask
Image-Text Matching
Multi-modal Causal Self-Attention Mask
Image-Grounded Text Generation
Uni-modal Self-Attention Mask
Image-Text Contrastive Learning

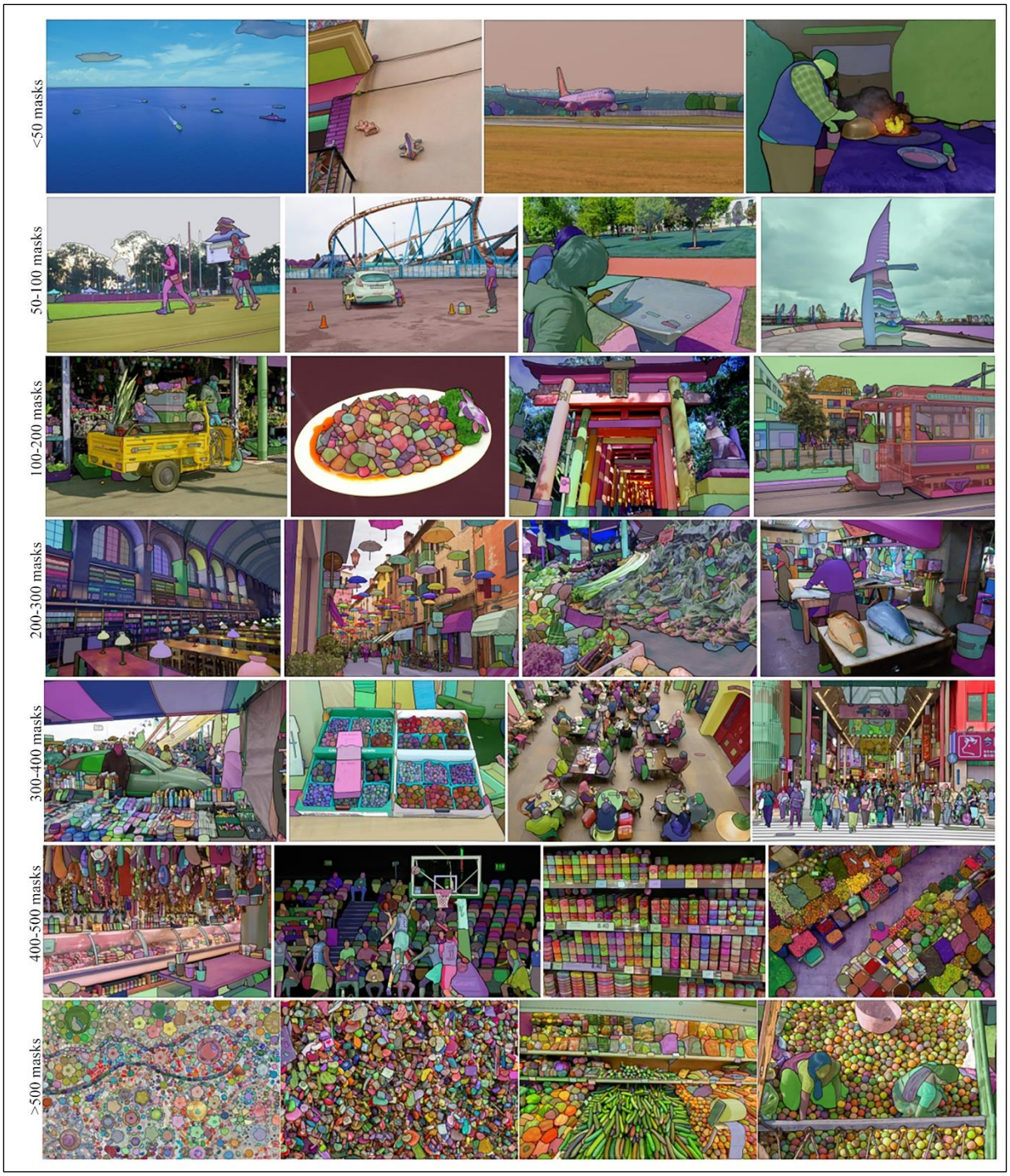

Bootstrapping from a Decoder-based Large Language Model (e.g. OPT)
Input Image
Image Encoder
Q-Former
Learned Queries
Fully Connected
LLM Decoder
Output Text
a cat wearing sunglasses
Bootstrapping from an Encoder-Decoder-based Large Language Model (e.g. FlanT5)
Input Image
Image Encoder
Q-Former
Learned Queries
Fully Connected
LLM Encoder
LLM Decoder
Suffix Text
wearing sunglasses
a cat
Prefix Text

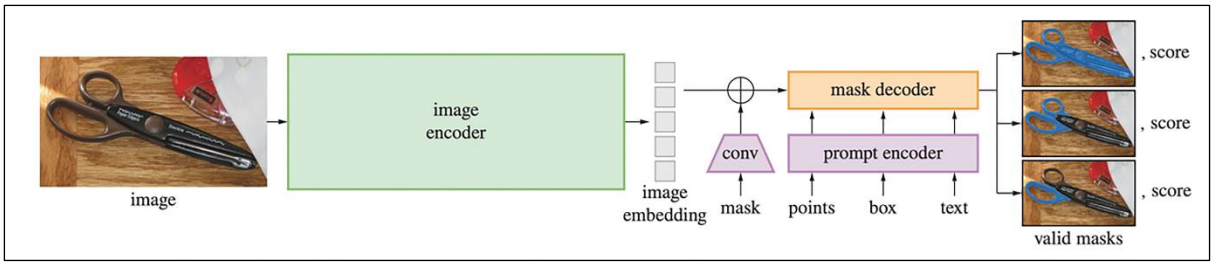
Chapter 16:


Pepper the Aussie pup
Text Encoder
Image Encoder
t1 t2 t3 ... tN
i1 i2 i3 iN
i1· t1 i1· t2 i1· t3 ... i1· tN
i2· t1 i2· t2 i2· t3 ... i2· tN
i3· t1 i3· t2 i3· t3 ... i3· tN
iN· t1 iN· t2 iN· t3 ... iN· tN


305
New Notebook
Download (1 GB)

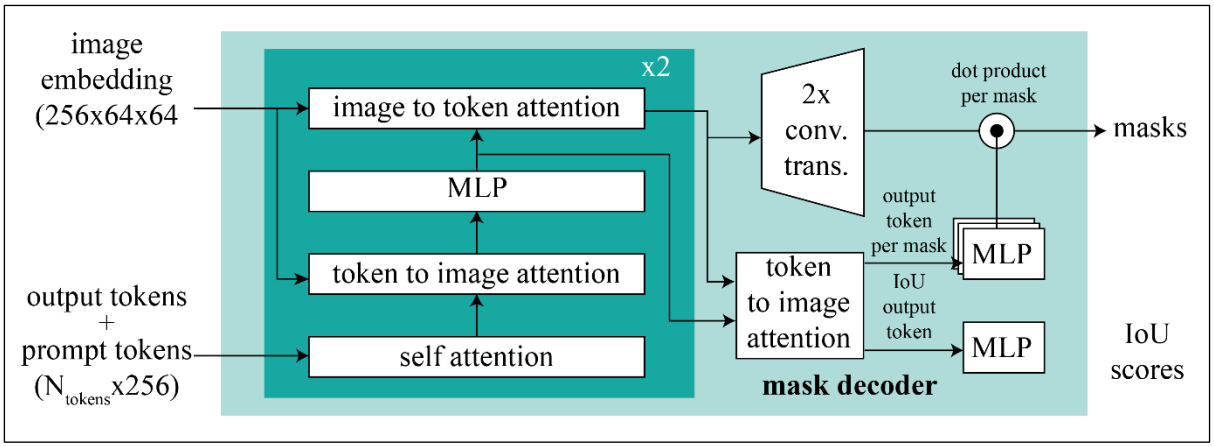

(1) Contrastive pre-training
Pepper the aussie pup
Text Encoder
Image Encoder
(2) Create dataset classifier from label text
plane
car
dog
bird
A photo of a {object}.
Text Encoder
(3) Use for zero-shot prediction
Image Encoder
A photo of a dog.


<50 masks
50-100 masks
100-200 masks
200-300 masks
300-400 masks
400-500 masks
>500 masks

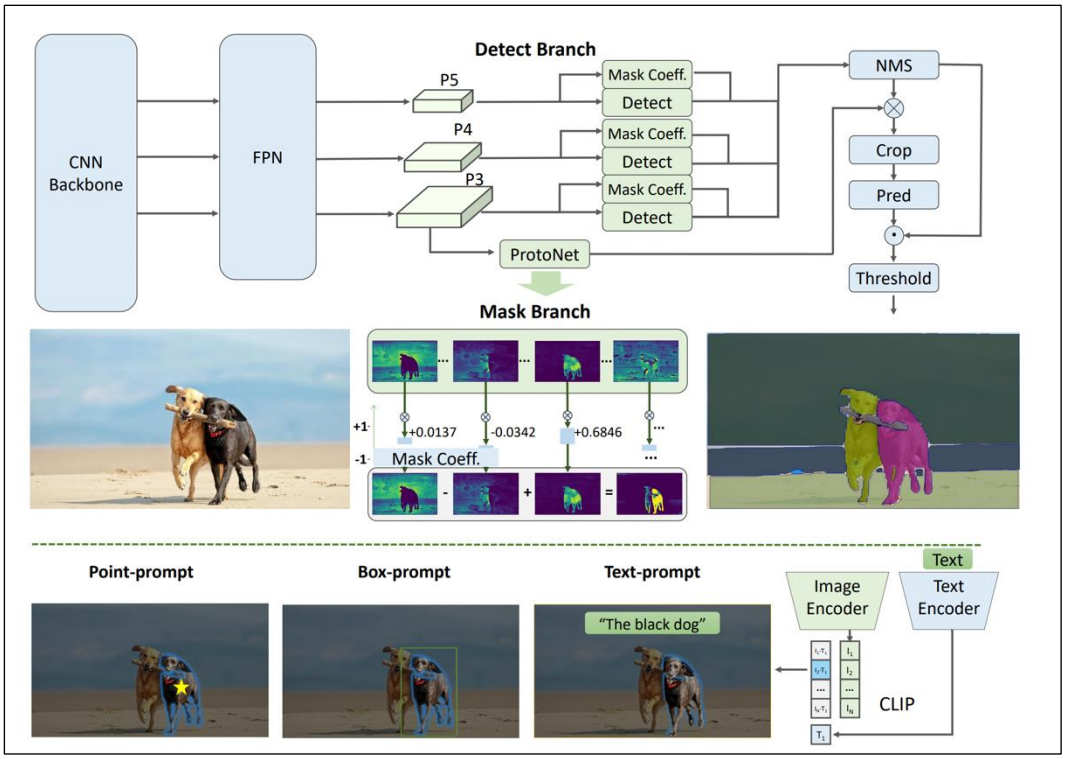


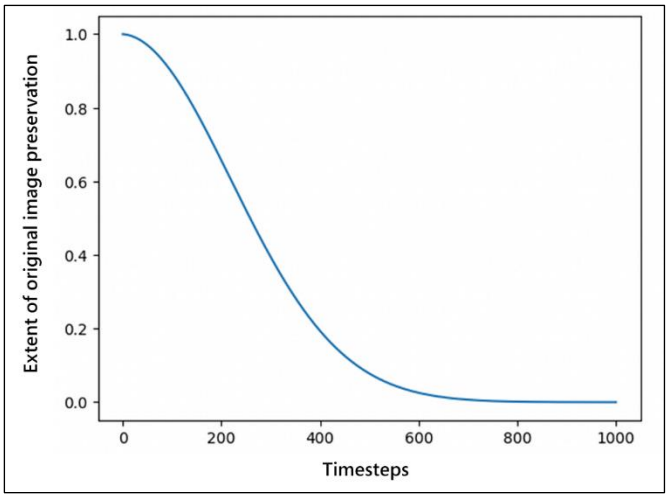

```python
def forward(
    self,
    points: Optional[Tuple[torch.Tensor, torch.Tensor]],
    boxes: Optional[torch.Tensor],
    masks: Optional[torch.Tensor],
):
    bs = self._get_batch_size(points, boxes, masks)
    sparse_embeddings = torch.empty((bs, 0, self.embed_dim), device=self._get_device())
    if points is not None:
        coords, labels = points
        point_embeddings = self._embed_points(coords, labels, pad=(boxes is None))
        sparse_embeddings = torch.cat([sparse_embeddings, point_embeddings], dim=1)
    if boxes is not None:
        box_embeddings = self._embed_boxes(boxes)
        sparse_embeddings = torch.cat([sparse_embeddings, box_embeddings], dim=1)

    if masks is not None:
        dense_embeddings = self._embed_masks(masks)
    else:
        dense_embeddings = self.no_mask_embed.weight.reshape(1, -1, 1, 1).expand(
            bs, -1, self.image_embedding_size[0], self.image_embedding_size[1]
        )

    return sparse_embeddings, dense_embeddings
```

```
Module Name: PromptEncoder
Input Kwargs:
    points
        0 - tensor[64, 1, 2] f64 n=128 (1Kb) x∈[10.672, 672.328] μ=186.750 σ=209.371 cuda:0 - ID:#5b3b2d
        1 - tensor[64, 1] i32 x∈[1, 1] μ=1.000 σ=0. cuda:0 - ID:#08f4ad
    boxes - NoneType
    masks - NoneType

Input Args:

Outputs:
    sparse_embeddings - tensor[64, 2, 256] n=32768 (0.1Mb) x∈[-1.352, 1.358] μ=-0.006 σ=0.530 cuda:0 - ID:#ca373f
    dense_embeddings  - tensor[64, 256, 64, 64] n=67108864 (0.2Gb) x∈[-0.218, 0.138] μ=-0.001 σ=0.031 grad cuda:0 - ID:#3c6848
```

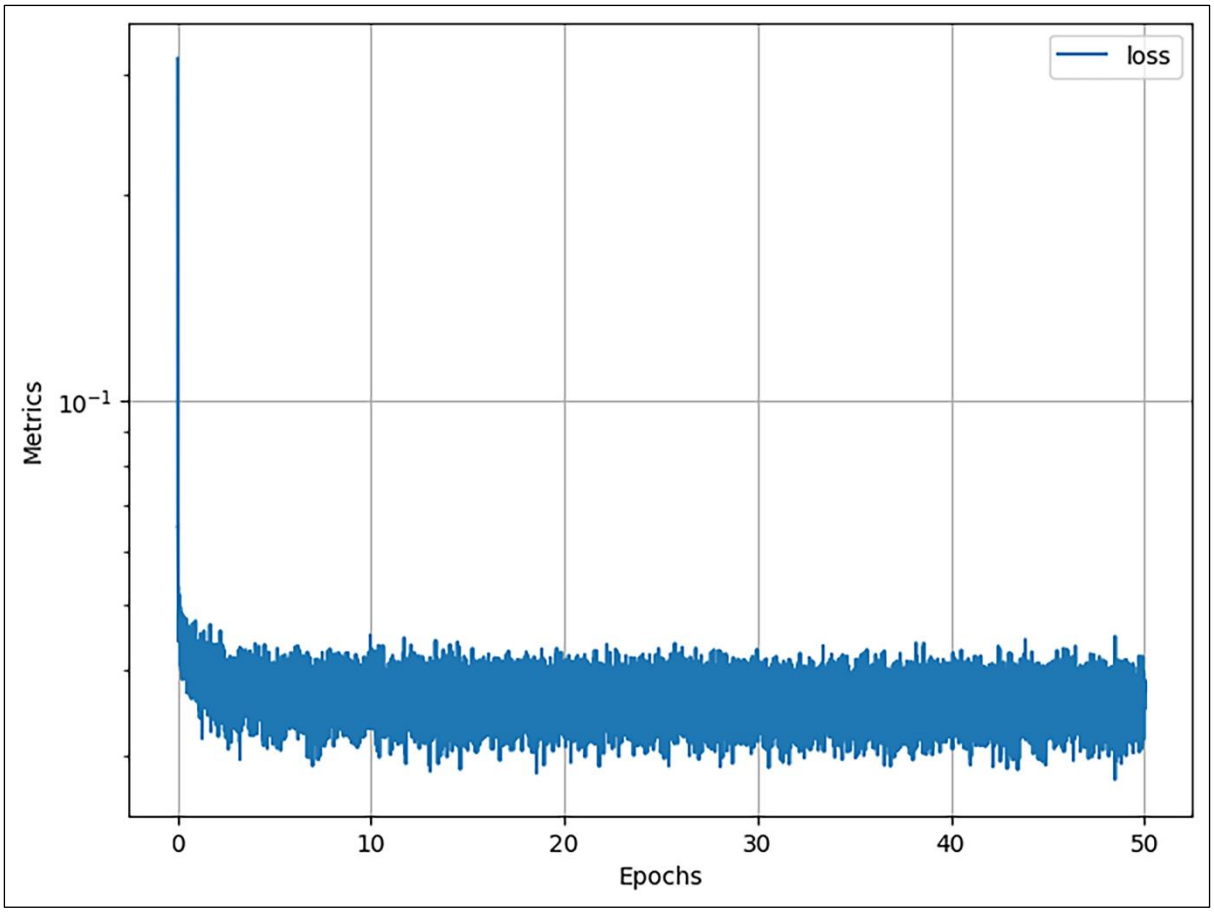


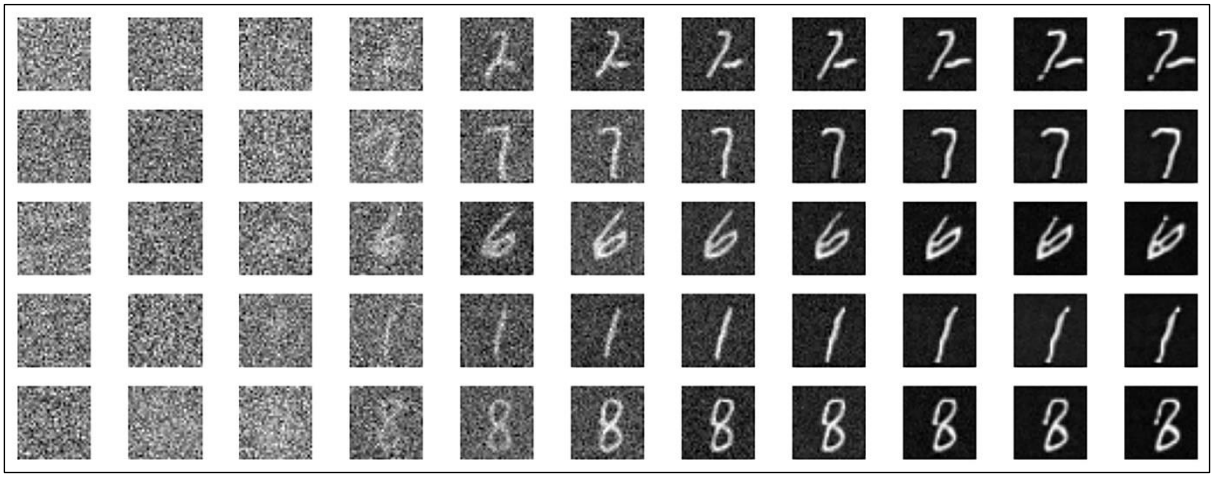

Mask 1, Score: 0.994
Mask 2, Score: 1.002
Mask 3, Score: 0.978

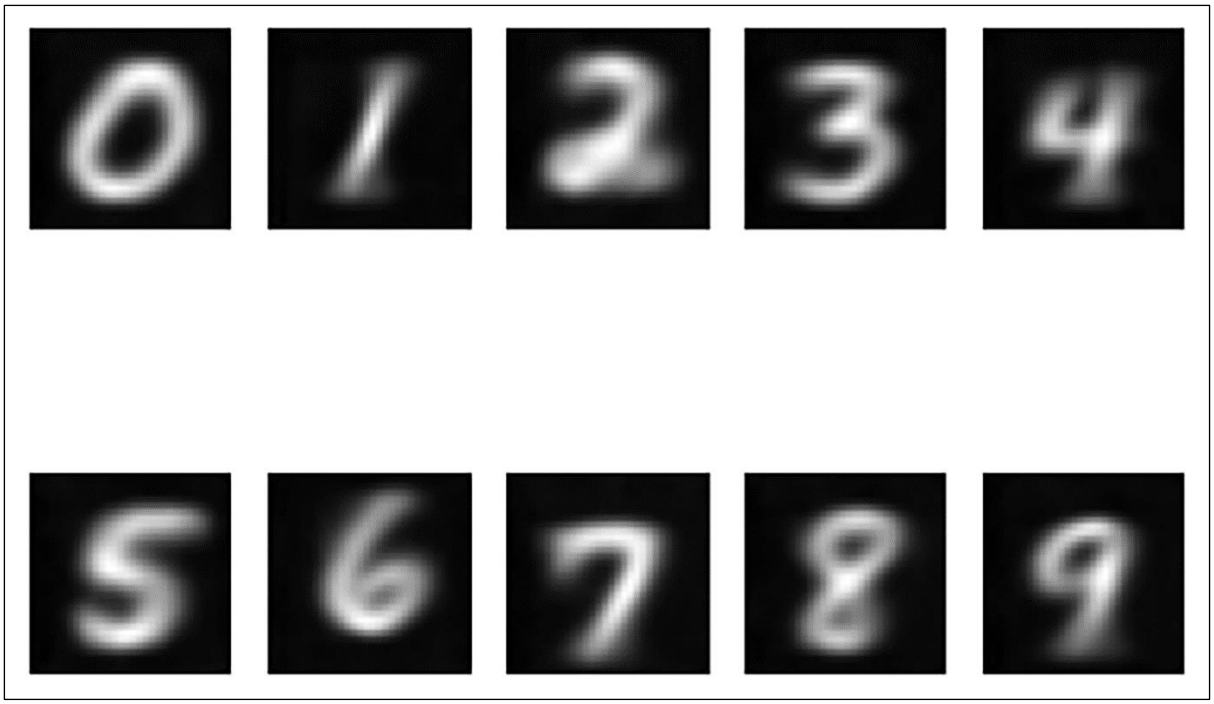

CNN Backbone
FPN
Detect Branch
P5
P4
P3
Mask Coeff.
Detect
Mask Coeff.
Detect
Mask Coeff.
Detect
ProtoNet
NMS
Crop
Pred
Threshold
Mask Branch
+1
-1
+0.0137
-0.0342
+0.6846
Mask Coeff.
Point-prompt
Box-prompt
Text-prompt
"The black dog"
Text
Image Encoder
Text Encoder
CLIP


Extent of original image preservation
1.0
0.8
0.6
0.4
0.2
0.0
0
200
400
600
800
1000
Timesteps

```
==========================================================================
Layer (type:depth-idx)                                  Param #
==========================================================================
UNet2DModel                                             --
├─Conv2d: 1-1                                           320
├─Timesteps: 1-2                                        --
├─TimestepEmbedding: 1-3                                --
│    └─Linear: 2-1                                      4,224
│    └─SiLU: 2-2                                        --
│    └─Linear: 2-3                                      16,512
├─ModuleList: 1-4                                       --
│    └─DownBlock2D: 2-4                                 32,000
│    └─AttnDownBlock2D: 2-5                             119,680
│    └─AttnDownBlock2D: 2-6                             460,544
│    └─AttnDownBlock2D: 2-7                             1,215,744
├─ModuleList: 1-5                                       --
│    └─AttnUpBlock2D: 2-8                               4,661,248
│    └─AttnUpBlock2D: 2-9                               1,347,840
│    └─AttnUpBlock2D: 2-10                              346,240
│    └─UpBlock2D: 2-11                                  78,528
├─UNetMidBlock2D: 1-6                                   --
│    └─ModuleList: 2-12                                 263,680
│    └─ModuleList: 2-13                                 2,428,416
├─GroupNorm: 1-7                                        64
├─SiLU: 1-8                                             --
├─Conv2d: 1-9                                           289
==========================================================================
Total params: 10,975,329
Trainable params: 10,975,329
Non-trainable params: 0
==========================================================================
```


loss
Metrics
10^{-1}
0
10
20
30
40
50
Epochs





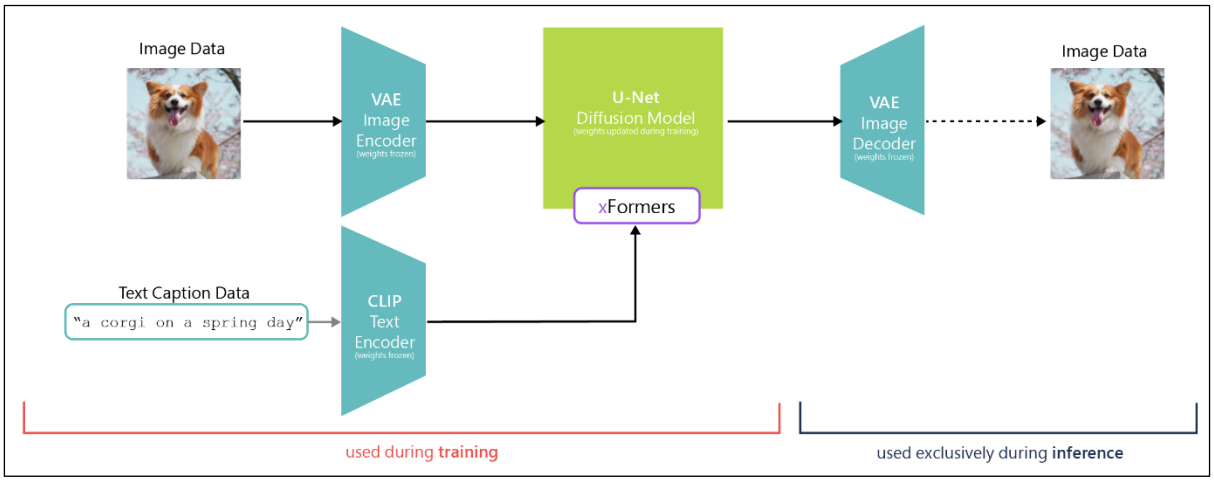
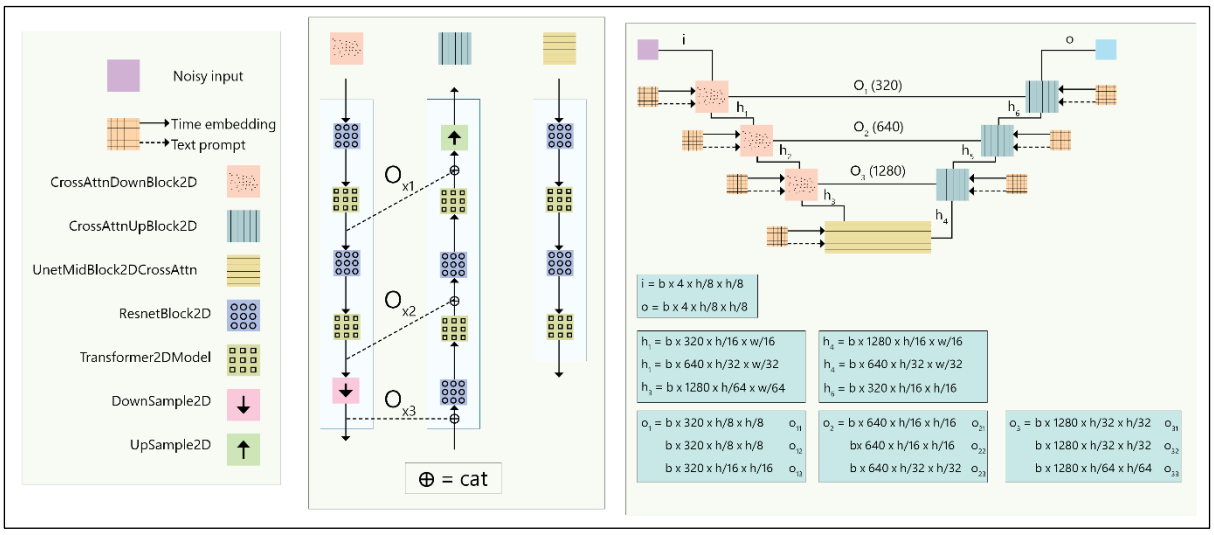


Image Data
VAE
Image
Encoder
(weights frozen)
U-Net
Diffusion Model
(weights updated during training)
xFormers
VAE
Image
Decoder
(weights frozen)
Image Data
Text Caption Data
"a corgi on a spring day"
CLIP
Text
Encoder
(weights frozen)
used during training
used exclusively during inference


Noisy input
Time embedding
Text prompt
CrossAttnDownBlock2D
CrossAttnUpBlock2D
UnetMidBlock2DCrossAttn
ResnetBlock2D
Transformer2DModel
DownSample2D
UpSample2D
⊕ = cat
i
o
O1 (320)
O2 (640)
O3 (1280)
i = b x 4 x h/8 x h/8
o = b x 4 x h/8 x h/8

```
==========================================================================================
Layer (type:depth-idx)                                  Param #
==========
UNet2DConditionModel                                    --
├─Conv2d: 1-1                                           11,840
├─Timesteps: 1-2                                        --
├─TimestepEmbedding: 1-3                                --
│    └─LoRACompatibleLinear: 2-1                        410,880
│    └─SiLU: 2-2                                        --
│    └─LoRACompatibleLinear: 2-3                        1,639,680
├─ModuleList: 1-4                                       --
│    └─CrossAttnDownBlock2D: 2-4                        10,852,160
│    └─CrossAttnDownBlock2D: 2-5                        37,473,920
│    └─CrossAttnDownBlock2D: 2-6                        141,303,040
│    └─DownBlock2D: 2-7                                 62,277,120
├─ModuleList: 1-5                                       --
│    └─UpBlock2D: 2-8                                   162,241,280
│    └─CrossAttnUpBlock2D: 2-9                          260,296,960
│    └─CrossAttnUpBlock2D: 2-10                         72,396,160
│    └─CrossAttnUpBlock2D: 2-11                         19,302,080
├─UNetMidBlock2DCrossAttn: 1-6                          --
│    └─ModuleList: 2-12                                 35,416,320
│    └─ModuleList: 2-13                                 62,277,120
├─GroupNorm: 1-7                                        640
├─SiLU: 1-8                                             --
├─Conv2d: 1-9                                           11,524
==========
Total params: 865,910,724
Trainable params: 865,910,724
Non-trainable params: 0
==========================================================================================
```

```
==========================================================================================
Layer (type:depth-idx)                                  Param #
==========
CrossAttnDownBlock2D                                    --
├─ModuleList: 1-1                                       --
│    └─Transformer2DModel: 2-1                          2,710,080
│    └─Transformer2DModel: 2-2                          2,710,080
├─ModuleList: 1-2                                       --
│    └─ResnetBlock2D: 2-3                               2,255,040
│    └─ResnetBlock2D: 2-4                               2,255,040
├─ModuleList: 1-3                                       --
│    └─Downsample2D: 2-5                                921,920
==========
Total params: 10,852,160
Trainable params: 10,852,160
Non-trainable params: 0
==========================================================================================
```

```
hidden_states - tensor[2, 320, 96, 96] n=5898240 (22Mb) x∈[-3.921, 3.216] μ=0.004 σ=0.351 cuda:0
temb - tensor[2, 1280] n=2560 (10Kb) x∈[-2.014, 5.339] μ=-0.003 σ=0.256 cuda:0
encoder_hidden_states - tensor[2, 77, 1024] n=157696 (0.6Mb) x∈[-6.571, 13.023] μ=-0.169 σ=1.032 cuda:0
```

```python
    def forward(
        self,
        hidden_states: torch.FloatTensor,
        temb: Optional[torch.FloatTensor] = None,
        encoder_hidden_states: Optional[torch.FloatTensor] = None,
    ):
        output_states = ()
        for i, (resnet, attn) in enumerate(blocks):
            hidden_states = resnet(hidden_states, temb)
            hidden_states = attn(
                hidden_states,
                encoder_hidden_states=encoder_hidden_states,
            )[0]
            output_states = output_states + (hidden_states,)
        if self.downsamplers is not None:
            for downsampler in self.downsamplers:
                hidden_states = downsampler(hidden_states, scale=lora_scale)
            output_states = output_states + (hidden_states,)
        return hidden_states, output_states
```

```
hidden_states -> tensor[2, 320, 48, 48] n=1474560 (5.6Mb) x∈[-13.873, 16.032] μ=0.017 σ=2.290 cuda:0
output_states ->
        1 - tensor[2, 320, 96, 96] n=5898240 (22Mb) x∈[-8.291, 6.062] μ=-0.084 σ=0.641 cuda:0
        2 - tensor[2, 320, 96, 96] n=5898240 (22Mb) x∈[-9.585, 7.211] μ=-0.082 σ=0.963 cuda:0
        3 - tensor[2, 320, 48, 48] n=1474560 (5.6Mb) x∈[-13.873, 16.032] μ=0.017 σ=2.290 cuda:0
```

```
hidden_states - tensor[2, 1280, 12, 12] n=368640 (1.4Mb) x∈[-43.398, 32.103] μ=-0.329 σ=4.600 cuda:0
temb - tensor[2, 1280] n=2560 (10Kb) x∈[-2.014, 5.339] μ=-0.003 σ=0.256 cuda:0
encoder_hidden_states - tensor[2, 77, 1024] n=157696 (0.6Mb) x∈[-6.571, 13.023] μ=-0.169 σ=1.032 cuda:0
```

```python
    def forward(
        self,
        hidden_states: torch.FloatTensor,
        temb: Optional[torch.FloatTensor] = None,
        encoder_hidden_states: Optional[torch.FloatTensor] = None,
    ):
        hidden_states = self.resnets[0](hidden_states, temb)
        for attn, resnet in zip(self.attentions, self.resnets[1:]):
            hidden_states = attn(
                hidden_states,
                encoder_hidden_states=encoder_hidden_states,
            )[0]
            hidden_states = resnet(hidden_states, temb)
        return hidden_states
```

```
tensor[2, 1280, 12, 12] n=368640 (1.4Mb) x∈[-43.307, 34.261] μ=-0.404 σ=4.882 cuda:0
```

```
1. hidden_states - tensor[2, 640, 96, 96] n=11796480 (45Mb) x∈[-486.840, 68.294] μ=-0.102 σ=3.369 cuda:0
2. temb - tensor[2, 1280] n=2560 (10Kb) x∈[-2.014, 5.339] μ=-0.003 σ=0.256 cuda:0
3. res_hidden_states_tuple
        1 - tensor[2, 320, 96, 96] n=5898240 (22Mb) x∈[-3.706, 3.410] μ=0.003 σ=0.351 cuda:0
        2 - tensor[2, 320, 96, 96] n=5898240 (22Mb) x∈[-8.291, 6.062] μ=-0.084 σ=0.641 cuda:0
        3 - tensor[2, 320, 96, 96] n=5898240 (22Mb) x∈[-9.585, 7.211] μ=-0.082 σ=0.963 cuda:0
4. encoder_hidden_states - tensor[2, 77, 1024] n=157696 (0.6Mb) x∈[-6.571, 13.023] μ=-0.169 σ=1.032 cuda:0
```

```python
    def forward(
        self,
        hidden_states: torch.FloatTensor,
        res_hidden_states_tuple: Tuple[torch.FloatTensor, ...],
        temb: Optional[torch.FloatTensor] = None,
        encoder_hidden_states: Optional[torch.FloatTensor] = None,
    ):
        for resnet, attn in zip(self.resnets, self.attentions):
            # pop res hidden states
            res_hidden_states = res_hidden_states_tuple[-1]
            res_hidden_states_tuple = res_hidden_states_tuple[:-1]
            hidden_states = torch.cat([hidden_states, res_hidden_states], dim=1)
            hidden_states = resnet(hidden_states, temb, scale=lora_scale)
            hidden_states = attn(
                hidden_states,
                encoder_hidden_states=encoder_hidden_states,
                return_dict=False,
            )[0]

        if self.upsamplers is not None:
            for upsampler in self.upsamplers:
                hidden_states = upsampler(hidden_states, upsample_size, scale=lora_scale)

        return hidden_states
```

```
tensor[2, 320, 96, 96] n=5898240 (22Mb) x∈[-29.935, 52.374] μ=-0.834 σ=4.356 cuda:0
```

```
hidden_states - tensor[2, 320, 96, 96] n=5898240 (22Mb) x∈[-10.410, 6.776] μ=-0.085 σ=0.782 cuda:0
encoder_hidden_states - tensor[2, 77, 1024] n=157696 (0.6Mb) x∈[-6.571, 13.023] μ=-0.169 σ=1.032 cuda:0
```

```python
    def forward(
        self,
        hidden_states: torch.Tensor,
        encoder_hidden_states: Optional[torch.Tensor] = None,
    ):
        batch, _, height, width = hidden_states.shape
        residual = hidden_states
        hidden_states = self.norm(hidden_states)
        hidden_states = self.proj_in(hidden_states) # nn.Conv2D
        inner_dim = hidden_states.shape[1]
        hidden_states = hidden_states.permute(0, 2, 3, 1).reshape(batch, height * width, inner_dim)
        for block in self.transformer_blocks:
            hidden_states = block(
                hidden_states,
                attention_mask=attention_mask,
                encoder_hidden_states=encoder_hidden_states,
            )
        hidden_states = self.proj_out(hidden_states) # nn.Conv2D
        hidden_states = hidden_states.reshape(batch, height, width, inner_dim).permute(0, 3, 1, 2).contiguous()
        output = hidden_states + residual
        return Transformer2DModelOutput(sample=output)
```

```
tensor[2, 320, 96, 96] n=5898240 (22Mb) x∈[-30.745, 52.179] μ=-0.837 σ=4.354 cuda:0
```

```
hidden_states - tensor[2, 320, 96, 96] n=5898240 (22Mb) x∈[-3.619, 3.284] μ=0.004 σ=0.348 cuda:0
temb - tensor[2, 1280] n=2560 (10Kb) x∈[-2.014, 5.339] μ=-0.003 σ=0.256 cuda:0
```

```python
def forward(
        self,
        input_tensor: torch.FloatTensor,
        temb: torch.FloatTensor,
    ):
        hidden_states = input_tensor
        hidden_states = self.norm1(hidden_states)
        hidden_states = self.nonlinearity(hidden_states)
        hidden_states = self.conv1(hidden_states)
        if self.time_emb_proj is not None:
            if not self.skip_time_act:
                temb = self.nonlinearity(temb)
            temb = self.time_emb_proj(temb)[:, :, None, None]
        hidden_states = hidden_states + temb
        hidden_states = self.nonlinearity(hidden_states)
        hidden_states = self.dropout(hidden_states)
        hidden_states = self.conv2(hidden_states, scale) if not USE_PEFT_BACKEND else self.conv2(hidden_states)
        output_tensor = (input_tensor + hidden_states)
        return output_tensor
```

```
tensor[2, 320, 96, 96] n=5898240 (22Mb) x∈[-7.879, 5.240] μ=-0.017 σ=0.642 cuda:0
```

Chapter 17:

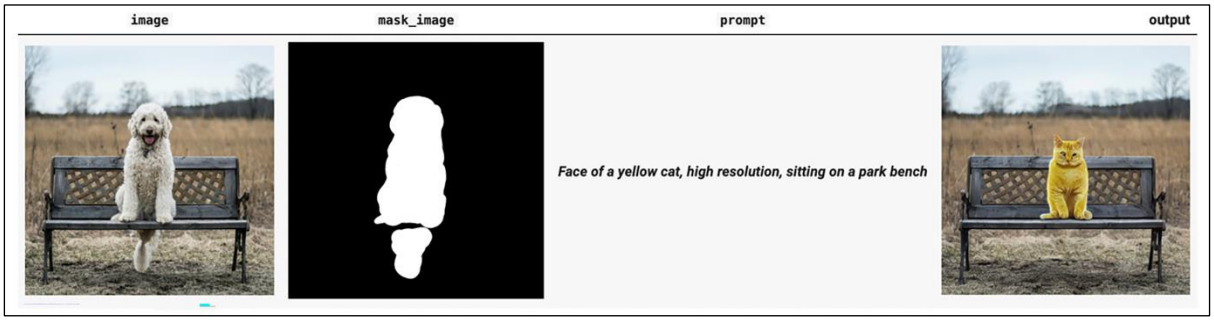


image
mask_image
prompt
output
Face of a yellow cat, high resolution, sitting on a park bench

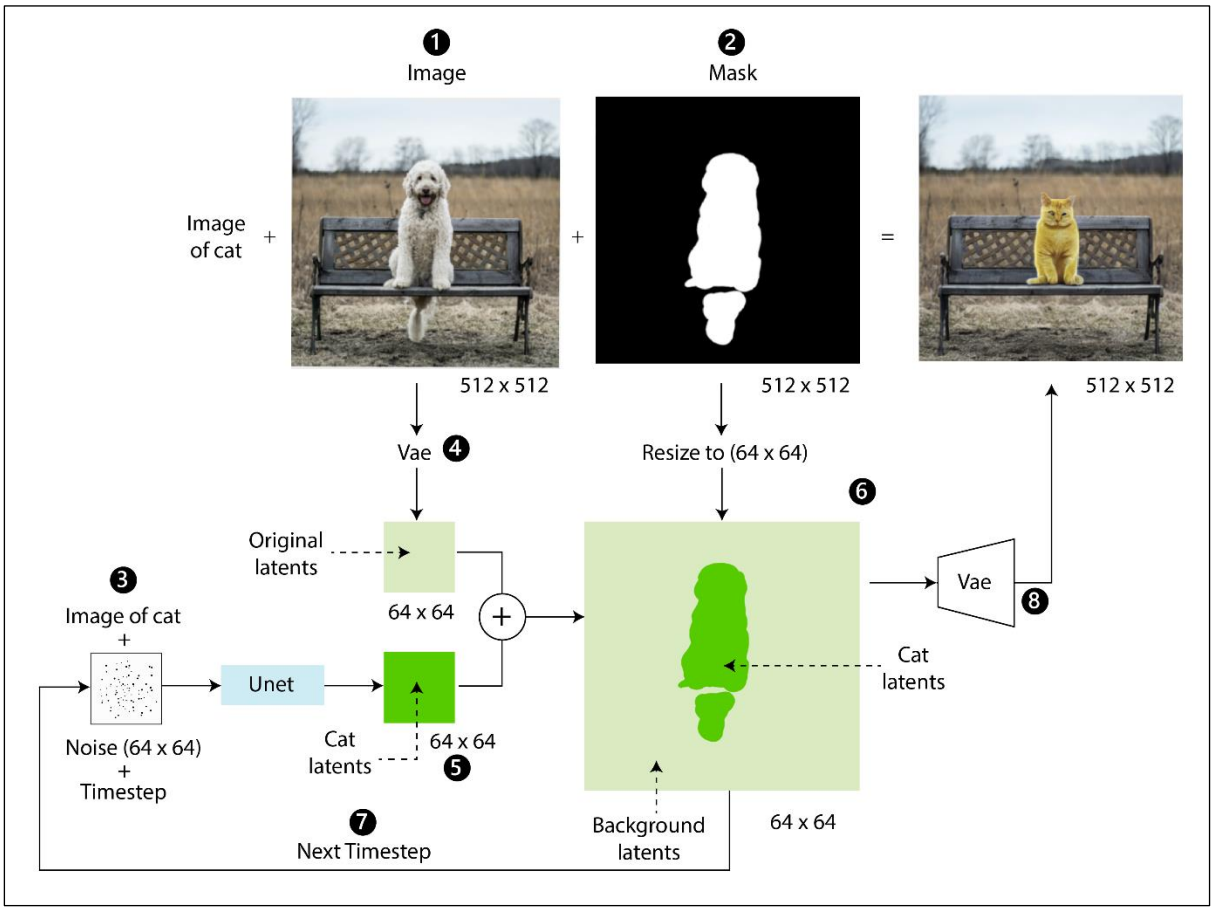
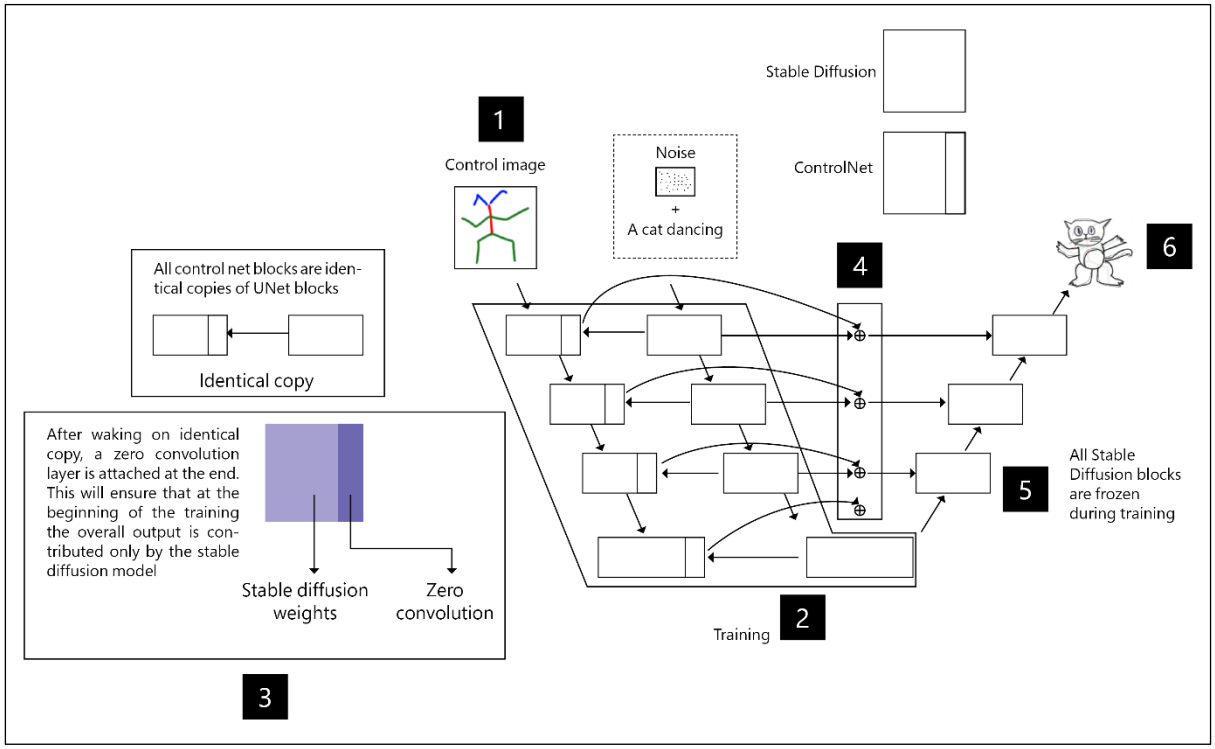
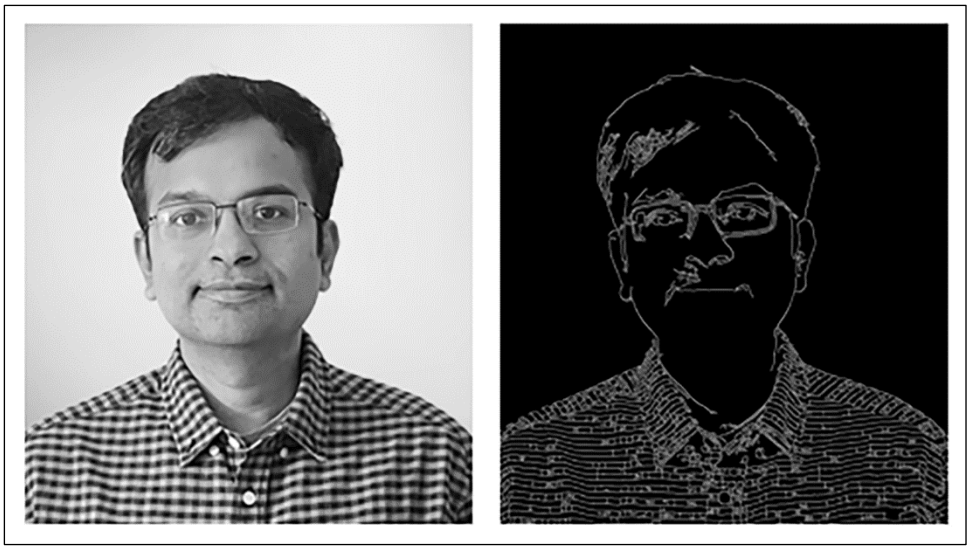
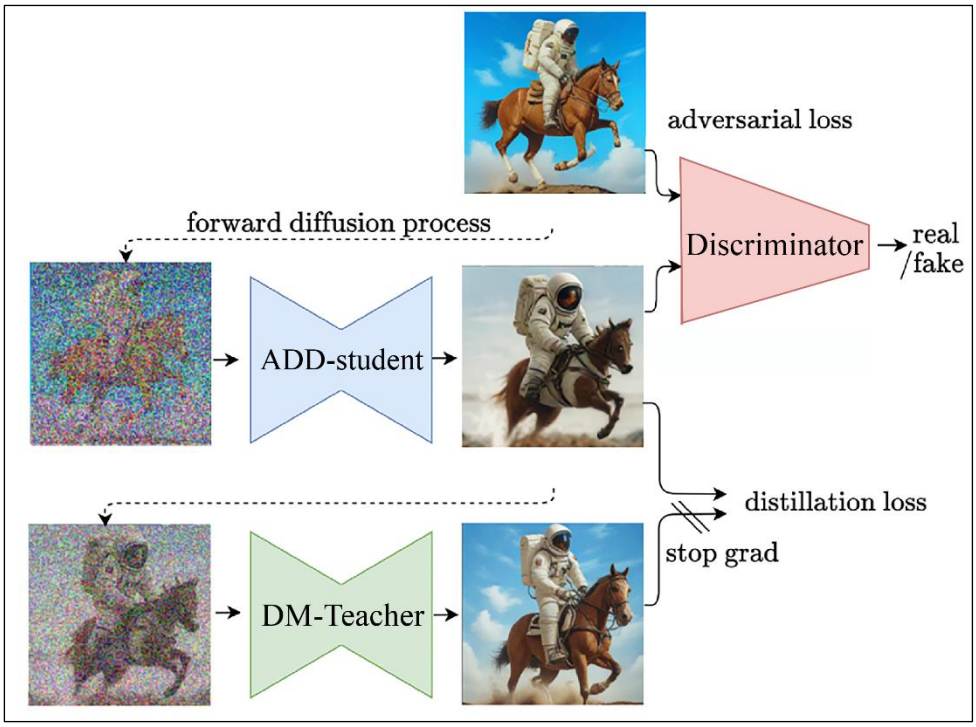
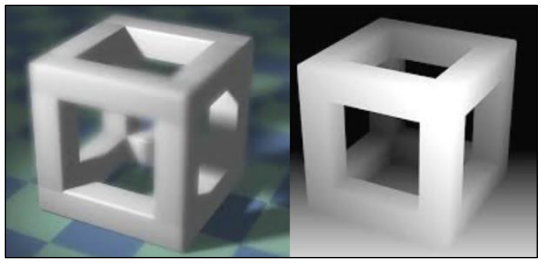

1
Image
2
Mask
Image of cat
+
+
=
512 x 512
512 x 512
512 x 512
Vae
4
Resize to (64 x 64)
6
Original latents
64 x 64
3
Image of cat
+
Noise (64 x 64)
+
Timestep
Unet
Cat latents
64 x 64
5
Vae
8
Cat latents
7
Next Timestep
Background latents
64 x 64


image
mask_image


Stable Diffusion
1
Control image
Noise
+
A cat dancing
ControlNet
All control net blocks are identical copies of UNet blocks
Identical copy
4
6
After waking on identical copy, a zero convolution layer is attached at the end. This will ensure that at the beginning of the training the overall output is contributed only by the stable diffusion model
Stable diffusion weights
Zero convolution
5
All Stable Diffusion blocks are frozen during training
Training
2
3

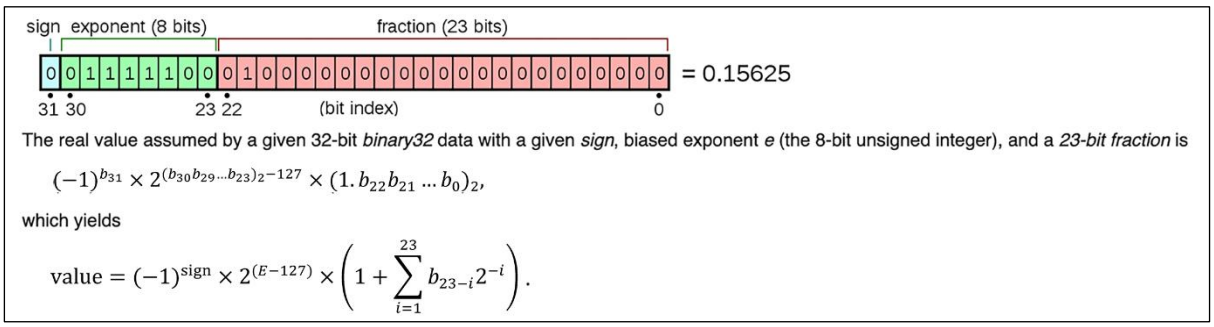



adversarial loss
forward diffusion process
Discriminator
real
/fake
ADD-student
distillation loss
stop grad
DM-Teacher



Chapter 18:


```
INFO:     Started server process [71575]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     127.0.0.1:62268 - "GET / HTTP/1.1" 404 Not Found
INFO:     127.0.0.1:62268 - "GET /favicon.ico HTTP/1.1" 404 Not Found
INFO:     127.0.0.1:62274 - "GET /docs HTTP/1.1" 200 OK
INFO:     127.0.0.1:62274 - "GET /openapi.json HTTP/1.1" 200 OK
```

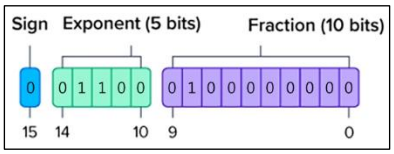


The real value assumed by a given 32-bit *binary32* data with a given *sign*, biased exponent *e* (the 8-bit unsigned integer), and a *23-bit fraction* is

$$(-1)^{b_{31}} \times 2^{(b_{30}b_{29}\ldots b_{23})_2-127} \times (1.b_{22}b_{21}\ldots b_0)_2,$$

which yields

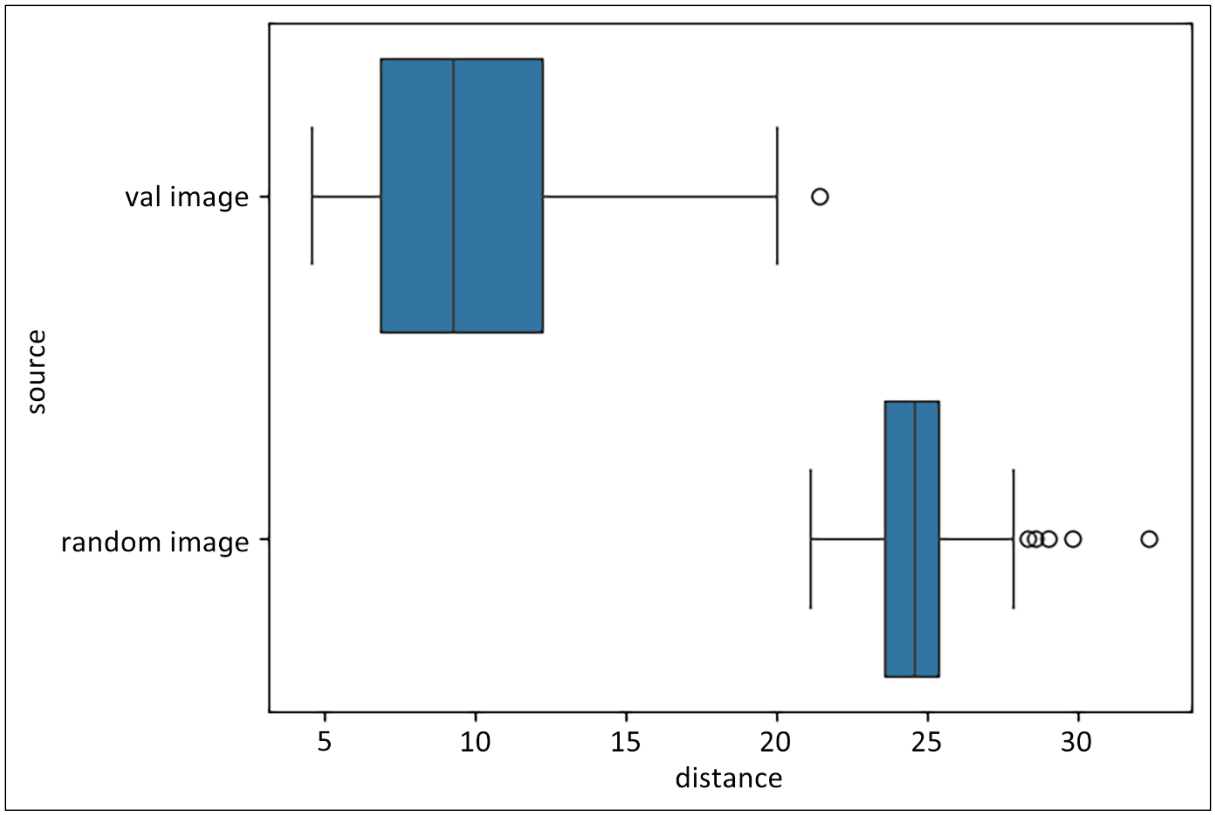
$$\text{value} = (-1)^{\text{sign}} \times 2^{(E-127)} \times \left(1 + \sum_{i=1}^{23} b_{23-i}2^{-i}\right).$$

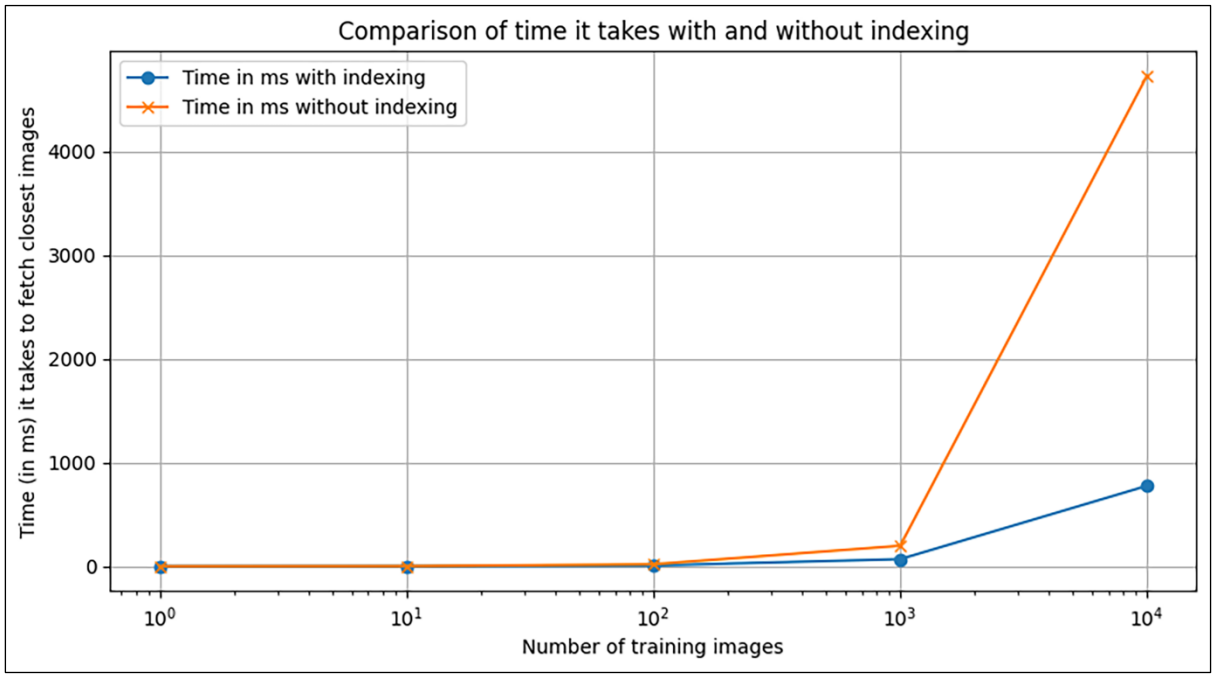



```
⇒ [8/9] COPY . /app                                                    0.8s
⇒ [9/9] RUN ls /app                                                    0.6s
⇒ exporting to image                                                   0.7s
⇒ ⇒ exporting layers                                                   0.6s
⇒ ⇒ writing image sha256:1b921c0b5f03a6eab430e518e341987e54b3f1462ec55f5e8  0.0s
⇒ ⇒ naming to docker.io/library/sdd:latest                             0.0s
```

```
❯ make docker-run
docker run -p 5000:5000 sdd:latest
INFO:     Started server process [1]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://0.0.0.0:5000 (Press CTRL+C to quit)
```

```
curl -X POST "http://127.0.0.1:5000/predict" -H "accept: application/json" -H "Conte
nt-Type: multipart/form-data" -F "file=@/tmp/non-defect.png;type=image/png"
```


val image
random image
source
5
10
15
20
25
30
distance


Comparison of time it takes with and without indexing
Time in ms with indexing
Time in ms without indexing
Time (in ms) it takes to fetch closest images
0
1000
2000
3000
4000
10^0
10^1
10^2
10^3
10^4
Number of training images