

DEEP LEARNING FOR COMPUTER VISION

Week6

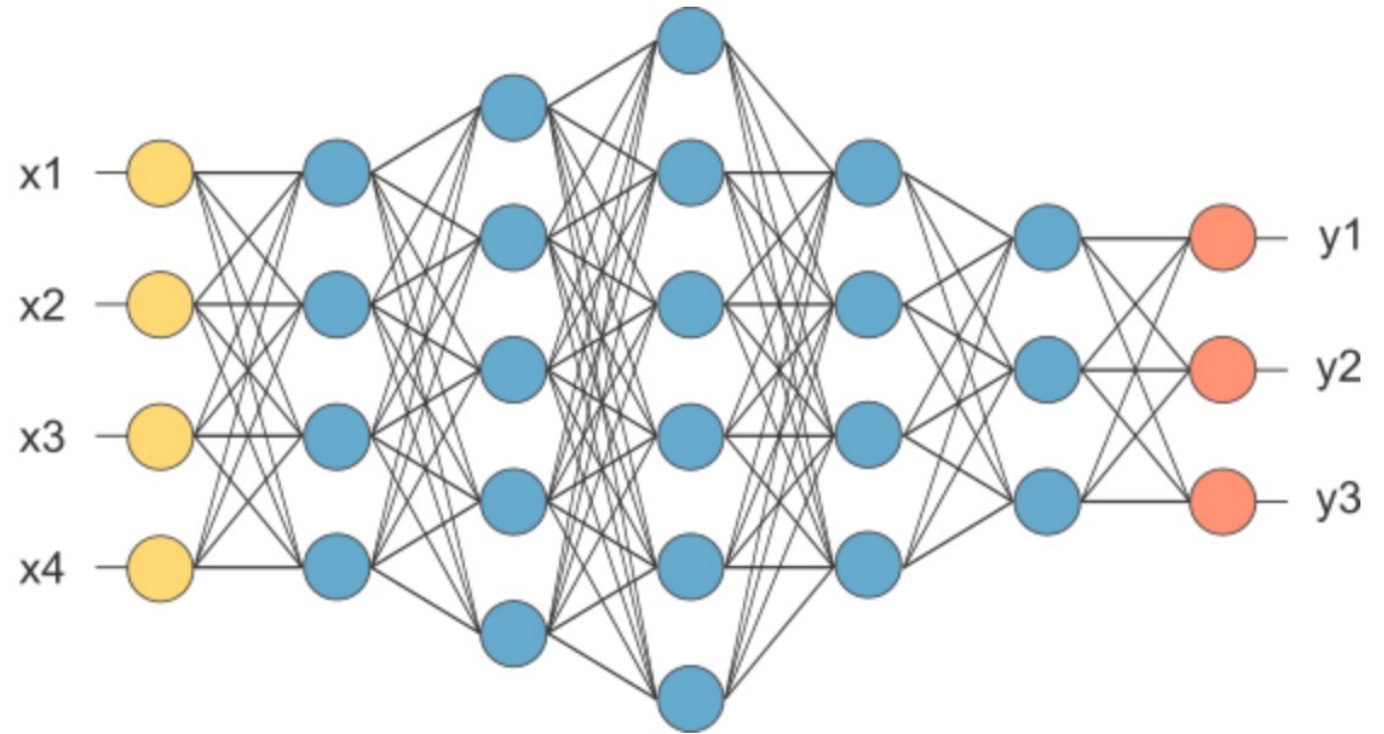


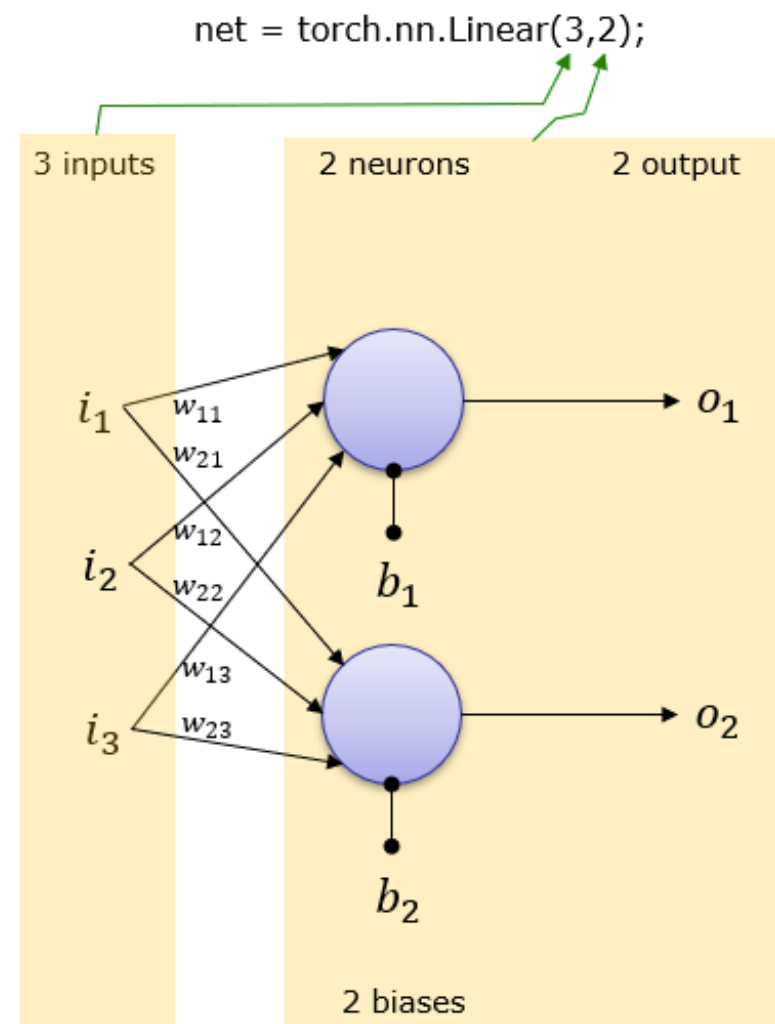
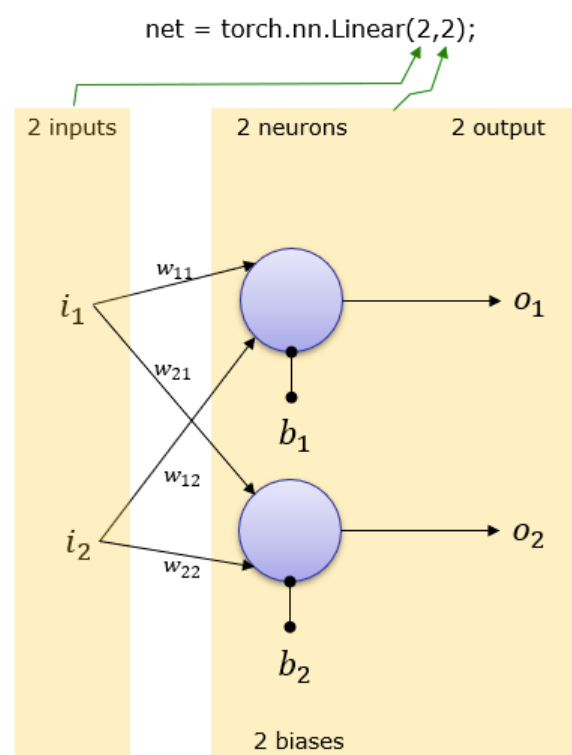
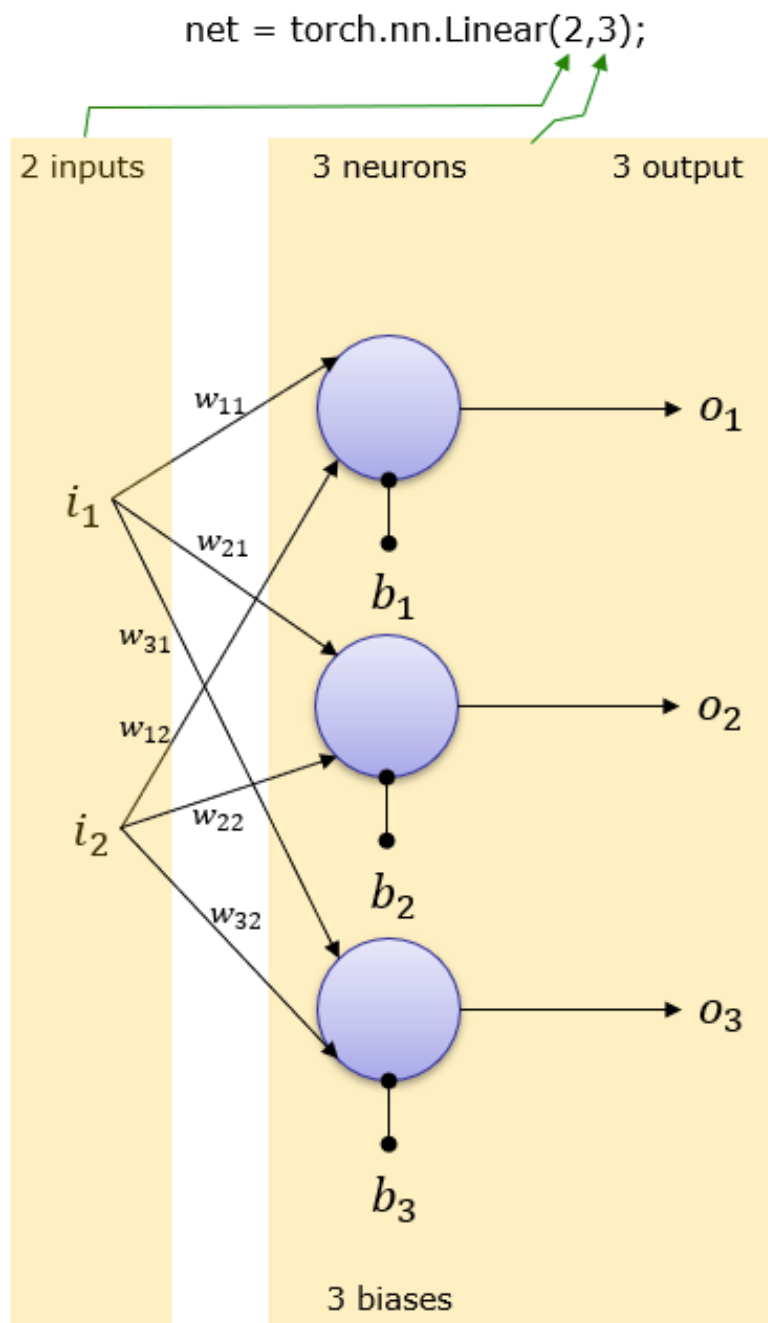
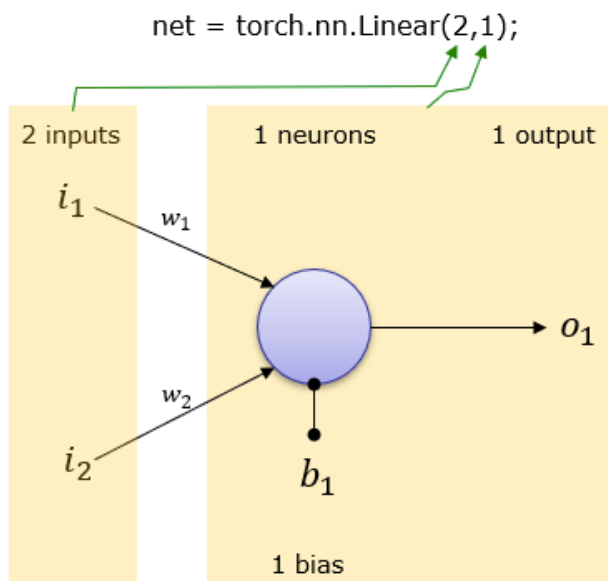
Dr. Tuchsanaï. PloySuwan

| สัปดาห์ที่ | หัวข้อ | วิธีการเรียน | ผลการเรียนรู้ | หมายเหตุ |
|------------|--|--------------|---------------------------|-----------|
| 1 | The basic concept of deep learning and their applications in computer vision | บรรยาย | CLO-1 | Chapter 1 |
| 2 | Introduction to Images and Videos Basics with Python and OpenCV | บรรยาย | CLO-1 | Chapter 1 |
| 3 | Advance in OpenCV | บรรยาย | CLO-1 | Chapter 1 |
| 4 | Introduction to <u>Pytorch</u> | บรรยาย | CLO-2, CLO-3, CLO-4 | Chapter 2 |
| 5 | Fundamental Neural network | บรรยาย | CLO-2 | Chapter 2 |
| 6 | Stochastic optimization methods | บรรยาย | CLO-4 | Chapter 2 |
| 7 | Convolutional neural network for image classification | บรรยาย | CLO-2, CLO-3 | Chapter 3 |
| 9 | Modern architectures such as VGG, <u>ResNet</u> , and <u>Efficientnet</u> | บรรยาย | CLO-3, | Chapter 3 |
| 10 | Transfer learning and fine tuning | บรรยาย | CLO-5 | Chapter 3 |
| 11 | Deep learning for Segmentation for images | บรรยาย | CLO-6 | Chapter 4 |
| 12 | Deep learning Object detection and Object Tracking | บรรยาย | CLO-7 | Chapter 4 |
| 13 | GAN architecture (Generative Adversarial Network) for photo-realistic images | บรรยาย | CLO-8 | Chapter 5 |
| 14 | Generative image from text CLIP model | บรรยาย | CLO-8 | Chapter 5 |
| 15 | Discussions | บรรยาย | | |

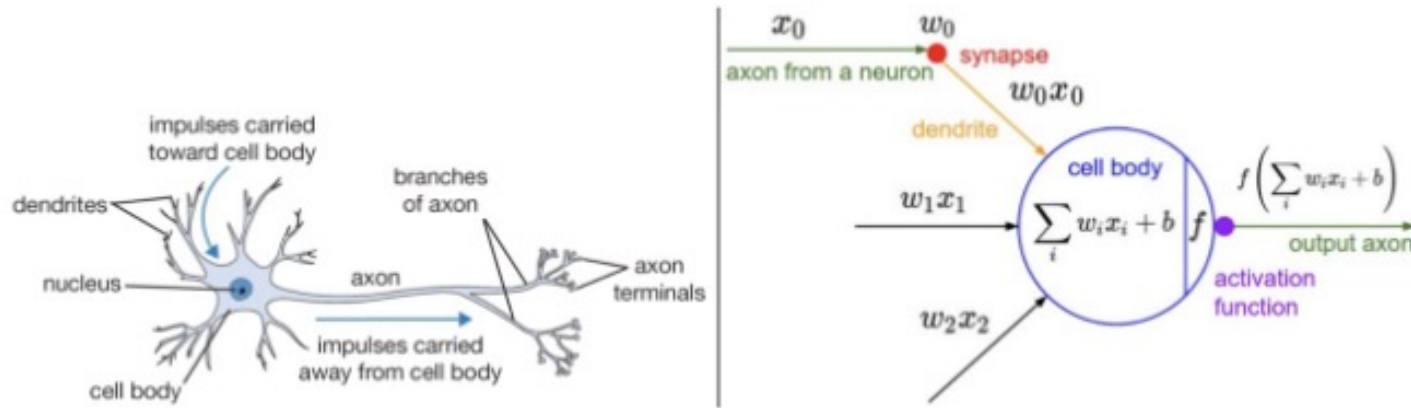
- **Fully Connected Layer**

The layer we call as FC layer, we flattened our matrix into vector and feed it into a fully connected layer like a neural network.



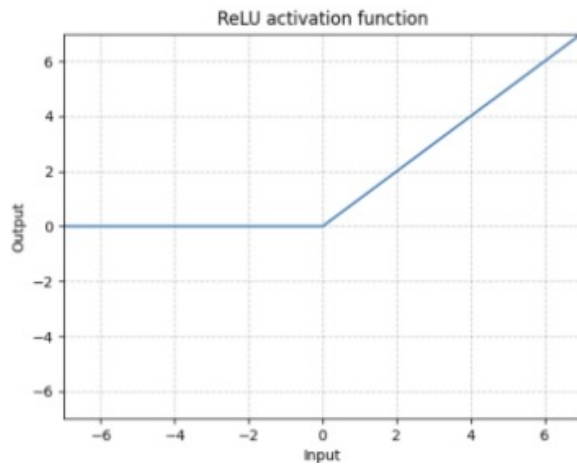


Activation functions in Neural Networks

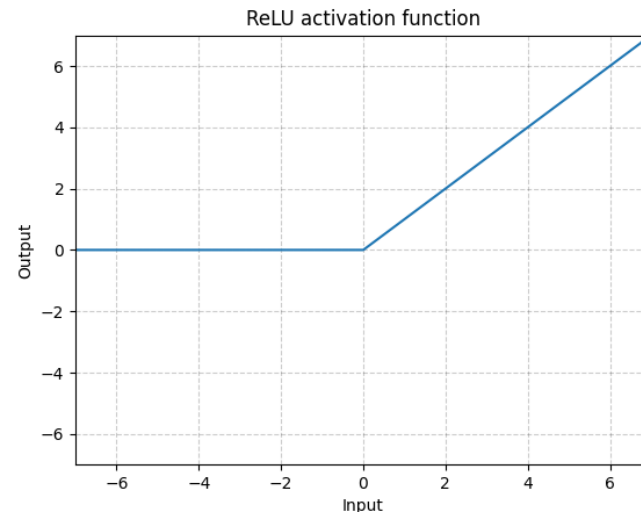


A cartoon drawing of a biological neuron (left) and its mathematical model (right).

$$\text{ReLU}(x) = (x)^+ = \max(0, x)$$



$$\text{LeakyReLU}(x) = \begin{cases} x, & \text{if } x \geq 0 \\ \text{negative_slope} \times x, & \text{otherwise} \end{cases}$$



$$\text{Sigmoid}(x) = \sigma(x) = \frac{1}{1 + \exp(-x)}$$

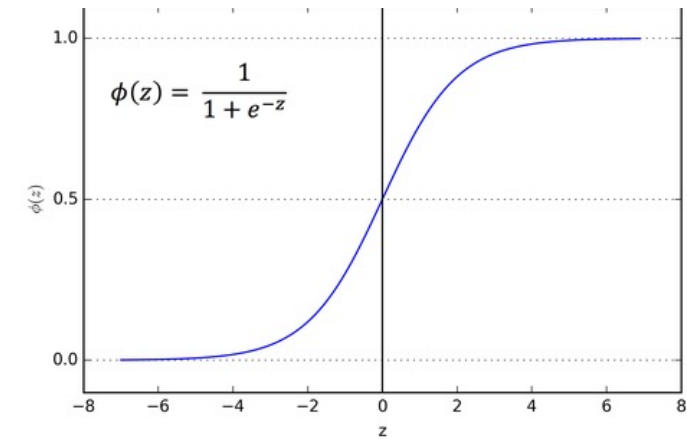
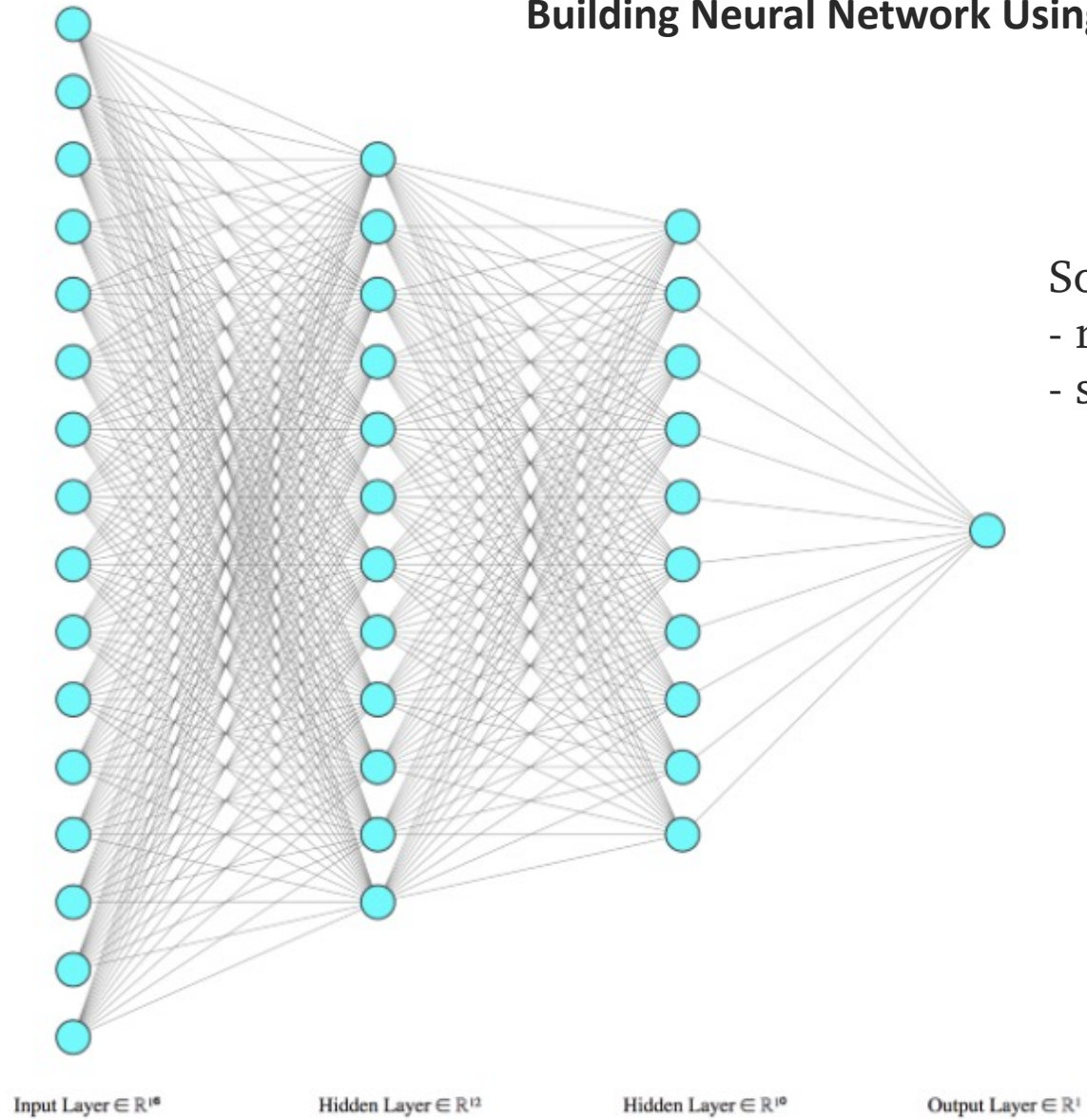


Fig: Sigmoid Function

`torch.nn.ReLU()` -

`torch.nn.Sigmoid`

Building Neural Network Using PyTorch



So this is a Fully Connected 16x12x10x1 Neural Network -
- relu activations in hidden layers,
- sigmoid activation in output layer.

Fully Connected (Feed Forward) Network

Three Ways to Build a Neural Network in PyTorch

```
import torch
import torch.nn.functional as F
from torch import nn

# define the network class
class MyNetwork(nn.Module):
    def __init__(self):
        # call constructor from superclass
        super().__init__()

        # define network layers
        self.fc1 = nn.Linear(16, 12)
        self.fc2 = nn.Linear(12, 10)
        self.fc3 = nn.Linear(10, 1)

    def forward(self, x):
        # define forward pass
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = torch.sigmoid(self.fc3(x))
        return x

# instantiate the model
model = MyNetwork()
```

```
1  from torch import nn
2
3  # define model architecture
4  model = nn.Sequential(
5      nn.Linear(16, 12),
6      nn.ReLU(),
7      nn.Linear(12, 10),
8      nn.ReLU(),
9      nn.Linear(10, 1),
10     nn.Sigmoid()
11 )
12
13 # print model architecture
14 print(model)
```

```
1  from torch import nn
2  from collections import OrderedDict
3
4  # define model architecture
5  model = nn.Sequential(OrderedDict([
6      ('fc1', nn.Linear(16, 12)),
7      ('relu1', nn.ReLU()),
8      ('fc2', nn.Linear(12, 10)),
9      ('relu2', nn.ReLU()),
10     ('fc3', nn.Linear(10, 1)),
11     ('sigmoid', nn.Sigmoid())
12 ]))
13
14 # print model architecture
15 print(model)
```

```
1  import torch
2  from torch import nn
3
4  class MyNetwork2(nn.Module):
5      def __init__(self):
6          super().__init__()
7
8          # define the layers
9          self.layers = nn.Sequential(
10              nn.Linear(16, 12),
11              nn.ReLU(),
12              nn.Linear(12, 10),
13              nn.ReLU(),
14              nn.Linear(10, 1)
15          )
16
17      def forward(self, x):
18          # forward pass
19          x = torch.sigmoid(self.layers(x))
20          return x
21
22  # instantiate the model
23  model = MyNetwork2()
24
25  # print model architecture
26  print(model)
```