

Computer Vision for Embedded Systems

Yung-Hsiang Lu
Purdue University
yunglu@purdue.edu



Yung-Hsiang Lu, Purdue University



Machine Learning: A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T , as measured by P , improves with experience E .

What is machine learning (ML)?

data \Rightarrow program \Rightarrow results

(1) no learning

data \Rightarrow program \Rightarrow results



(2) learning

pattern

data \Rightarrow program

(3) training



pattern

data \Rightarrow program \Rightarrow results



(4) inference

pattern

Different types of learning

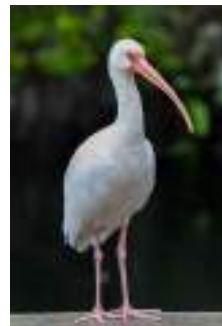
(1) Supervised learning: A teacher provides correct answers.

teacher
↓
Panda



Different types of learning

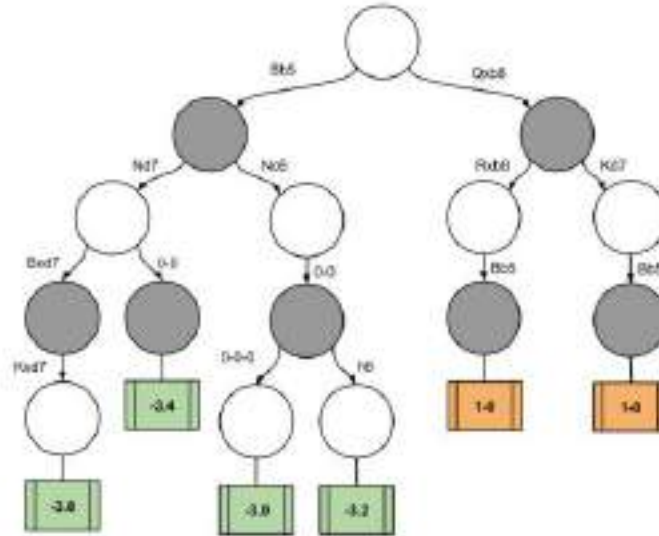
(2) Unsupervised learning: No teacher tells correct answers.



ImageNet

Different types of learning

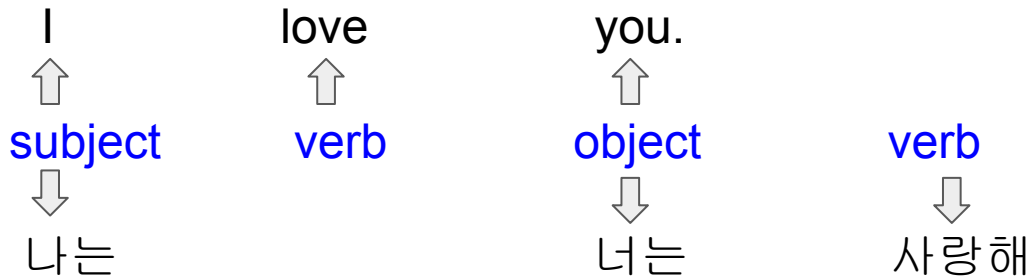
(3) Reinforcement learning: Evaluate the effects of sequences of decisions



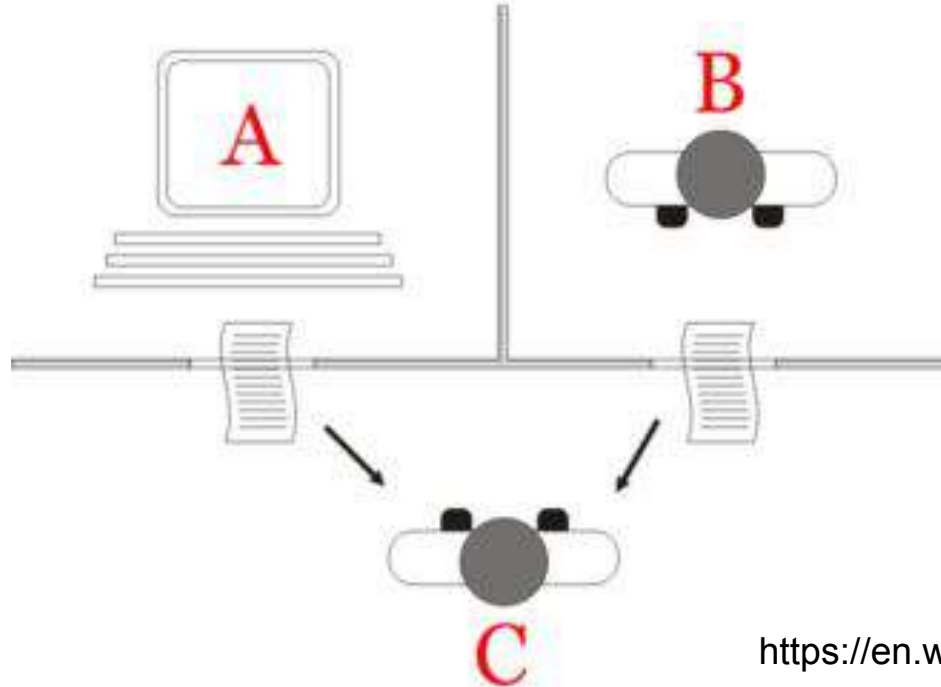
<https://kevinbinz.com/2015/02/28/the-chess-supertree/>

Different types of learning

(4) Transfer learning: Use knowledge in one (source) domain for another (target) domain



Turing Test of Intelligence



https://en.wikipedia.org/wiki/Turing_test

IBM's Deep Blue



<https://spectrum.ieee.org/how-ibms-deep-blue-beat-world-champion-chess-player-garry-kasparov>

Yung-Hsiang Lu, Purdue University

DeepMind's AlphaGo



<https://deepmind.com/alphago-korea>

https://www.youtube.com/watch?v=WXuK6gekU1Y&ab_channel=DeepMind

Sentient AI?



 SIGN IN

 NPR SHOP

 NEWS

 CULTURE

 MUSIC

 PODCASTS & SHOWS

 SEARCH

TECHNOLOGY



The Google engineer who sees company's AI as 'sentient' thinks a chatbot has a soul

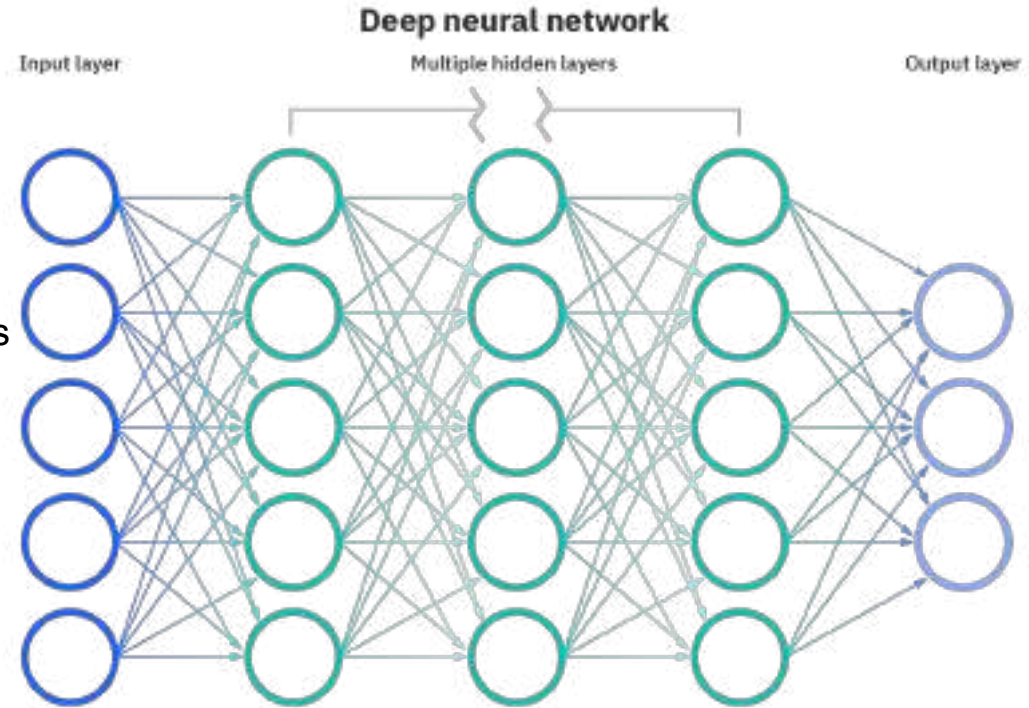
June 16, 2022 - 4:31 PM ET

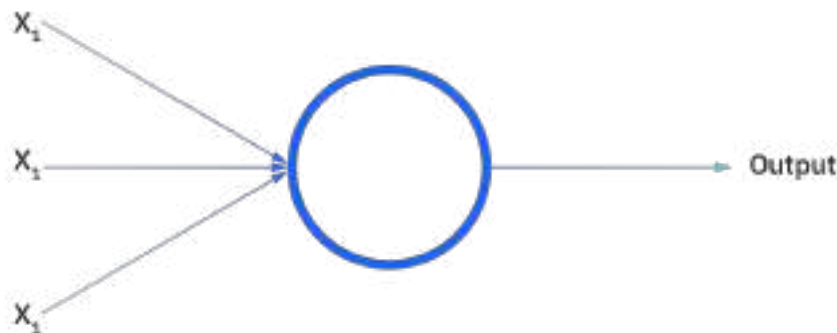
Heard on [All Things Considered](#)

Neural Networks

Hidden: not input or output
Deep: many hidden layers

<https://www.ibm.com/cloud/learn/neural-networks>





$$\sum_{i=1}^m w_i x_i + \text{bias} = w_1 x_1 + w_2 x_2 + w_3 x_3 + \text{bias}$$

$$\text{output} = f(x) = \begin{cases} 1 & \text{if } \sum w_i x_i + b \geq 0 \\ 0 & \text{if } \sum w_i x_i + b < 0 \end{cases}$$

Step Function

Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



tanh

$$\tanh(x)$$



ReLU

$$\max(0, x)$$



Rectified Linear Unit

Leaky ReLU

$$\max(0.1x, x)$$



Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

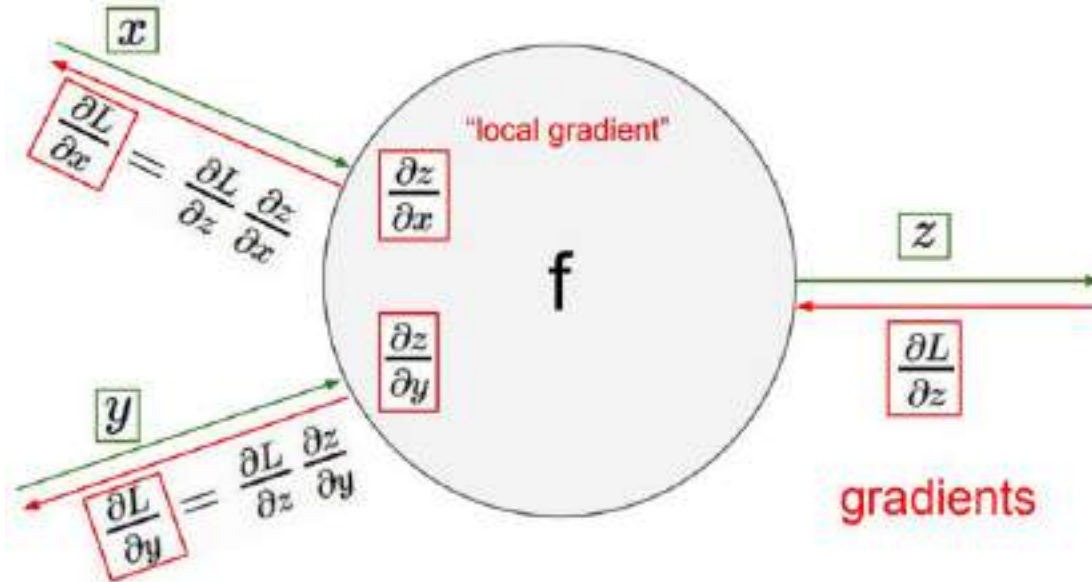
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



<https://www.ibm.com/cloud/learn/neural-networks>

<https://hsf-training.github.io/hsf-training-ml-webpage/03-nn/index.html>

Back propagation



<https://becominghuman.ai/back-propagation-in-convolutional-neural-networks-intuition-and-code-714ef1c38199>

Why complex machine models?



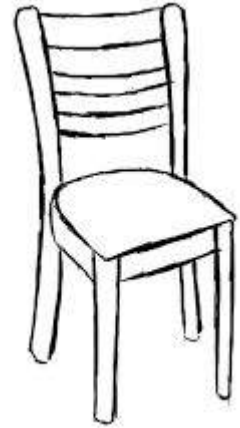
Car



Dog



Pineapple



Chair

<https://coloringhome.com/coloring-page/17144>

<https://www.easydrawingtips.com/dog-head-front-view-drawing-step-by-step/>

<https://www.pinterest.com/pin/479211216585933414/>

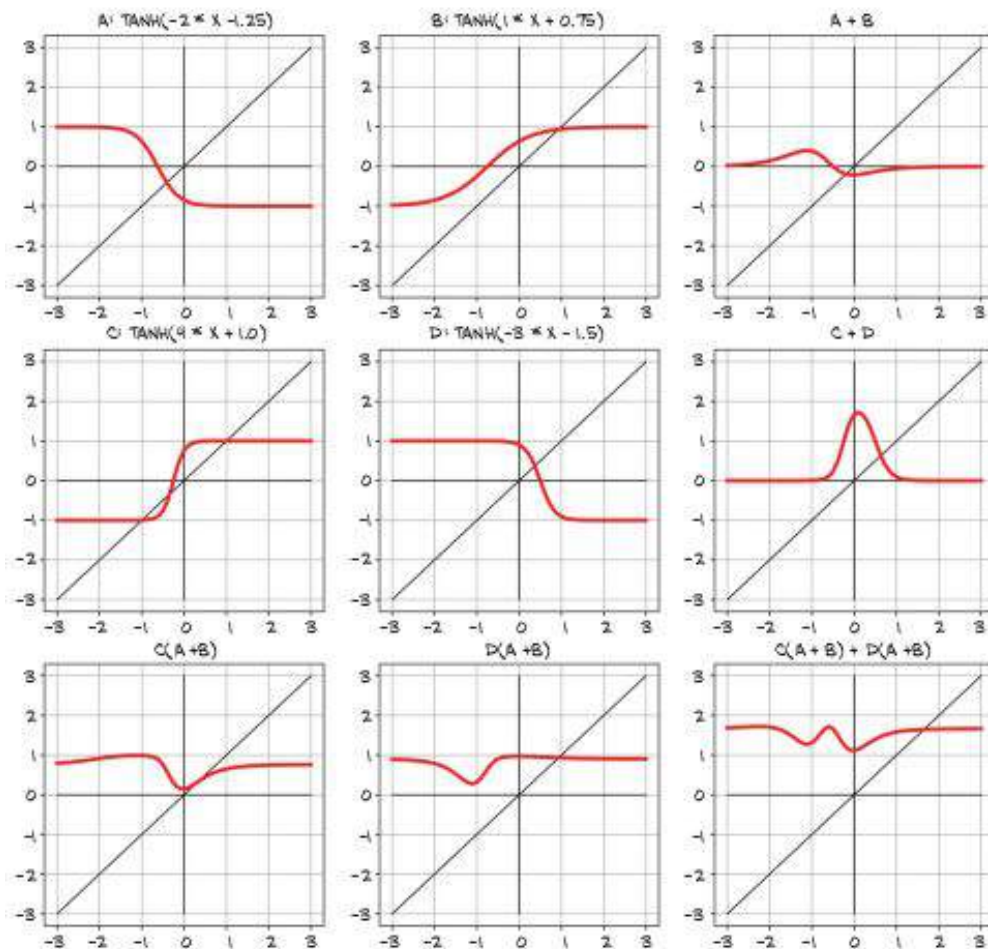
<https://feltmagnet.com/drawing/How-to-draw-a-chair>

Complex Functions

Deep Learning with PyTorch

ELI STEVENS, LUCA ANTIGA,

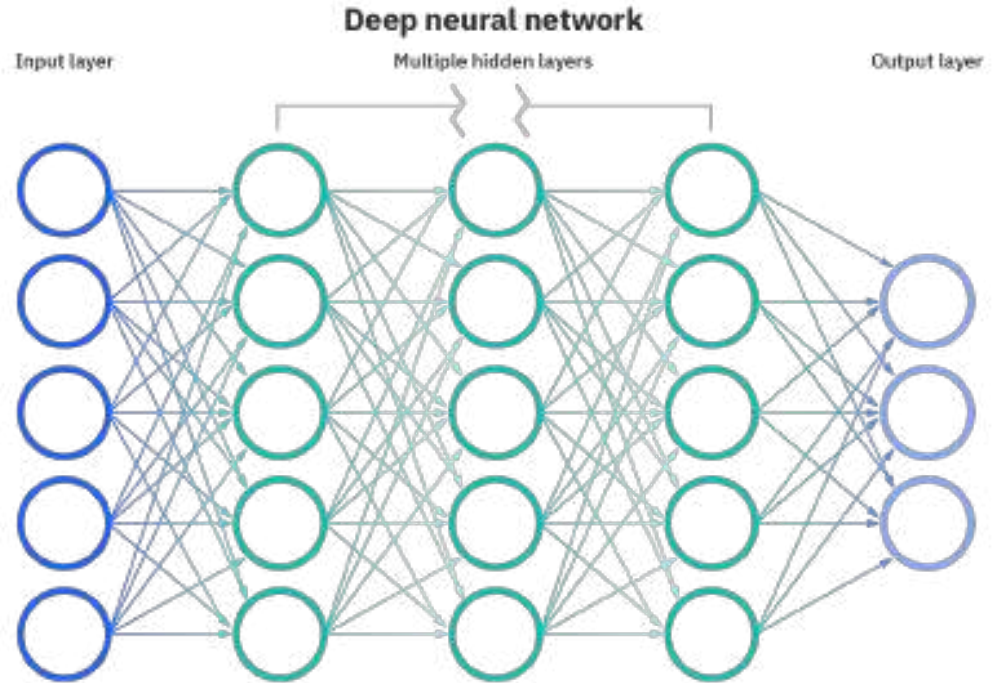
AND THOMAS VIEHMANN



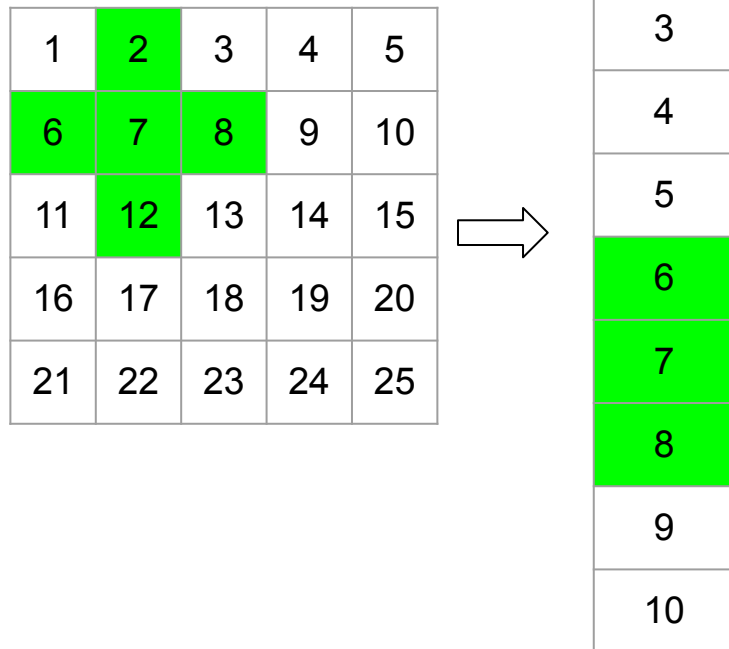
CNN: Convolutional Neural Network

Hidden: not input or output

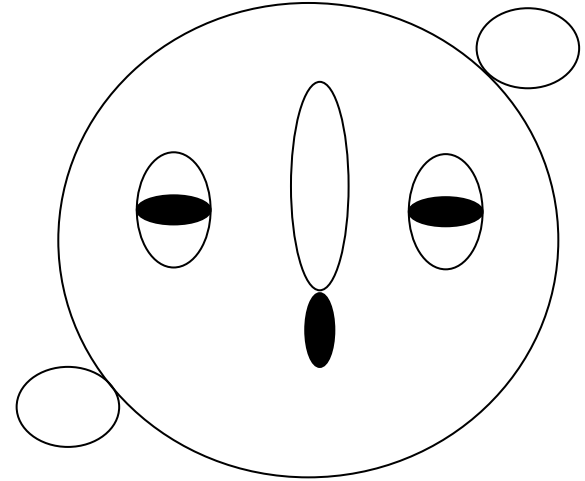
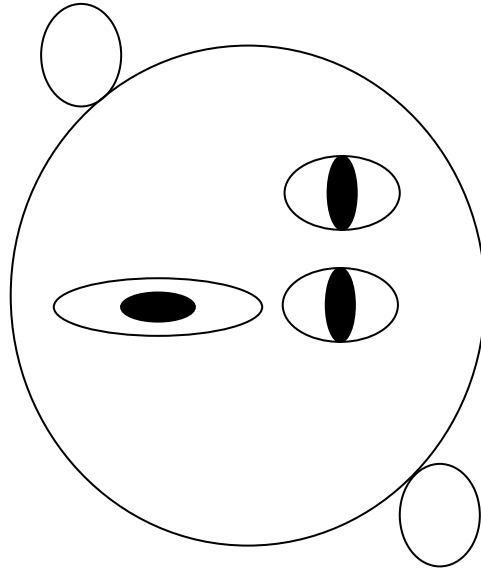
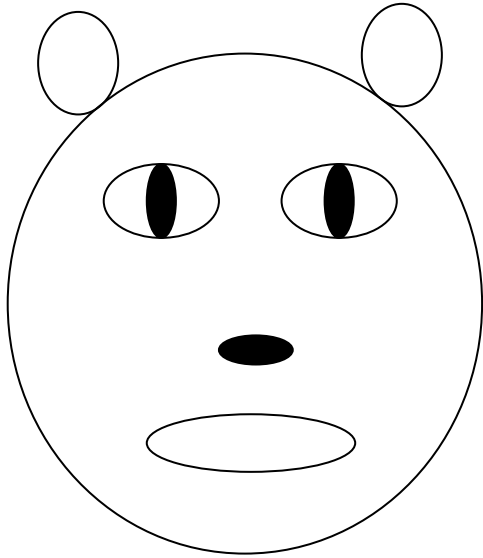
Deep: many hidden layers



Neural Networks are 1D structures



How to recognize the spatial relationship?



How to distinguish these patterns?

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

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

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

0	1	0
1	1	1
0	1	0

1	1	1
1	0	1
1	1	1

1	-1
1	-1

What is the right filter from x to y ?

```
x = tensor([[1., 1., 0., 0., 0., 0., 1., 1.],
            [1., 1., 0., 0., 0., 0., 1., 1.],
            [1., 1., 0., 0., 0., 0., 1., 1.],
            [1., 1., 0., 0., 0., 0., 1., 1.],
            [1., 1., 0., 0., 0., 0., 1., 1.],
            [1., 1., 0., 0., 0., 0., 1., 1.]])
y = tensor([[ 0.,  1.,  0.,  0.,  0., -1.,  0.],
            [ 0.,  1.,  0.,  0.,  0., -1.,  0.],
            [ 0.,  1.,  0.,  0.,  0., -1.,  0.],
            [ 0.,  1.,  0.,  0.,  0., -1.,  0.],
            [ 0.,  1.,  0.,  0.,  0., -1.,  0.],
            [ 0.,  1.,  0.,  0.,  0., -1.,  0.]])
```

```

# Construct a two-dimensional convolutional layer with 1 output channel and a
# kernel of shape (1, 2). For the sake of simplicity, we ignore the bias here
conv2d = nn.Conv2d(1, 1, kernel_size=(1, 2), bias=False)

# The two-dimensional convolutional layer uses four-dimensional input and
# output in the format of (example, channel, height, width), where the batch
# size (number of examples in the batch) and the number of channels are both 1
X = X.reshape((1, 1, 6, 8))
Y = Y.reshape((1, 1, 6, 7))
lr = 3e-2 # Learning rate

for i in range(10):
    Y_hat = conv2d(X)
    l = (Y_hat - Y)**2
    conv2d.zero_grad()
    l.sum().backward()
    # Update the kernel
    conv2d.weight.data[:] -= lr * conv2d.weight.grad
    if (i + 1) % 2 == 0:
        print(f'batch {i + 1}, loss {l.sum():.3f}')

```

https://d2l.ai/chapter_convolutional-neural-networks/conv-layer.html


```
# Construct a two-dimensional convolutional layer with 1 output channel and a
# kernel of shape (1, 2). For the sake of simplicity, we ignore the bias here
conv2d = nn.Conv2d(1, 1, kernel_size=(1, 2), bias=False)
```

```
# The two-dimensional convolutional layer uses four-dimensional input and
# output in the format of (example, channel, height, width), where the batch
# size (number of examples in the batch) and the number of channels are both 1
```

```
X = X.reshape((1, 1, 6, 8))
```

```
Y = Y.reshape((1, 1, 6, 7))
```

```
lr = 3e-2 # Learning rate
```

```
for i in range(10):
```

```
    Y_hat = conv2d(X)
```

```
    l = (Y_hat - Y)**2
```

```
    conv2d.zero_grad()
```

```
    l.sum().backward()
```

```
    # Update the kernel
```

```
    conv2d.weight.data[:] -= lr * conv2d.w
```

```
    if (i + 1) % 2 == 0:
```

```
        print(f'batch {i + 1}, loss {l.sum()}'
```

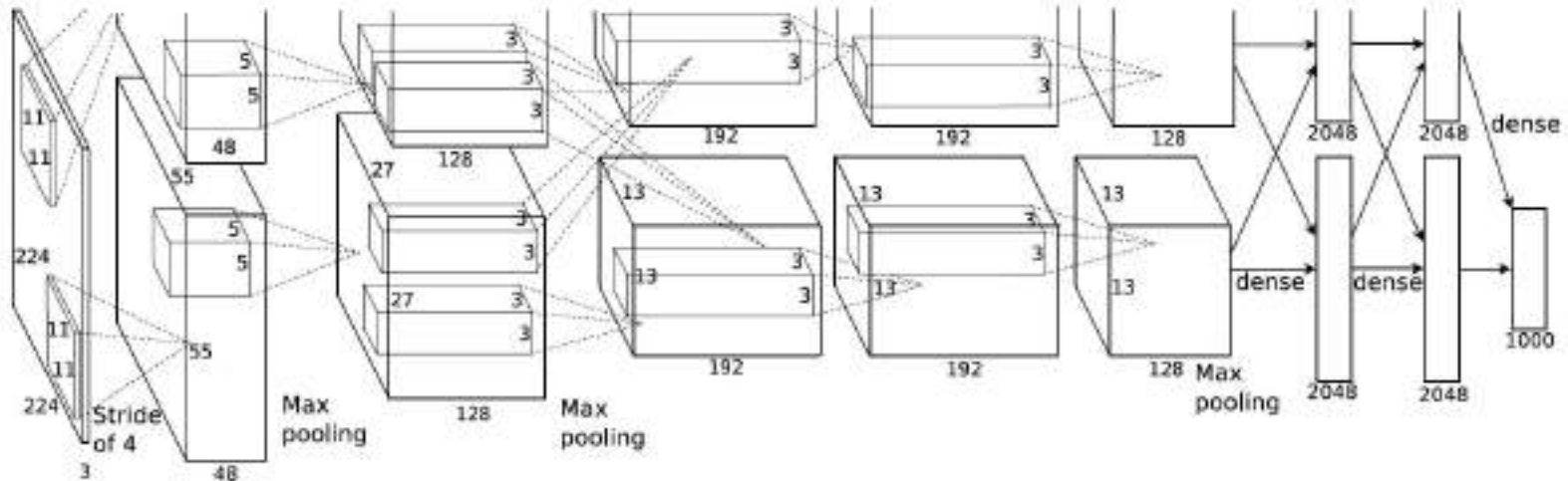
```
batch 2, loss 8.819
OrderedDict([('weight', tensor([[[[ 0.6129, -0.3754]]]]))])
batch 4, loss 1.799
OrderedDict([('weight', tensor([[[[ 0.8688, -0.7168]]]]))])
batch 6, loss 0.433
OrderedDict([('weight', tensor([[[[ 0.9638, -0.8665]]]]))])
batch 8, loss 0.126
OrderedDict([('weight', tensor([[[[ 0.9964, -0.9341]]]]))])
batch 10, loss 0.043
OrderedDict([('weight', tensor([[[[ 1.0057, -0.9658]]]]))])
batch 12, loss 0.016
OrderedDict([('weight', tensor([[[[ 1.0069, -0.9814]]]]))])
batch 14, loss 0.006
OrderedDict([('weight', tensor([[[[ 1.0058, -0.9895]]]]))])
batch 16, loss 0.003
OrderedDict([('weight', tensor([[[[ 1.0042, -0.9938]]]]))])
batch 18, loss 0.001
OrderedDict([('weight', tensor([[[[ 1.0029, -0.9963]]]]))])
batch 20, loss 0.000
OrderedDict([('weight', tensor([[[[ 1.0020, -0.9977]]]]))])
```

Deep Neural Networks (DNN)

AlexNet <https://en.wikipedia.org/wiki/AlexNet>

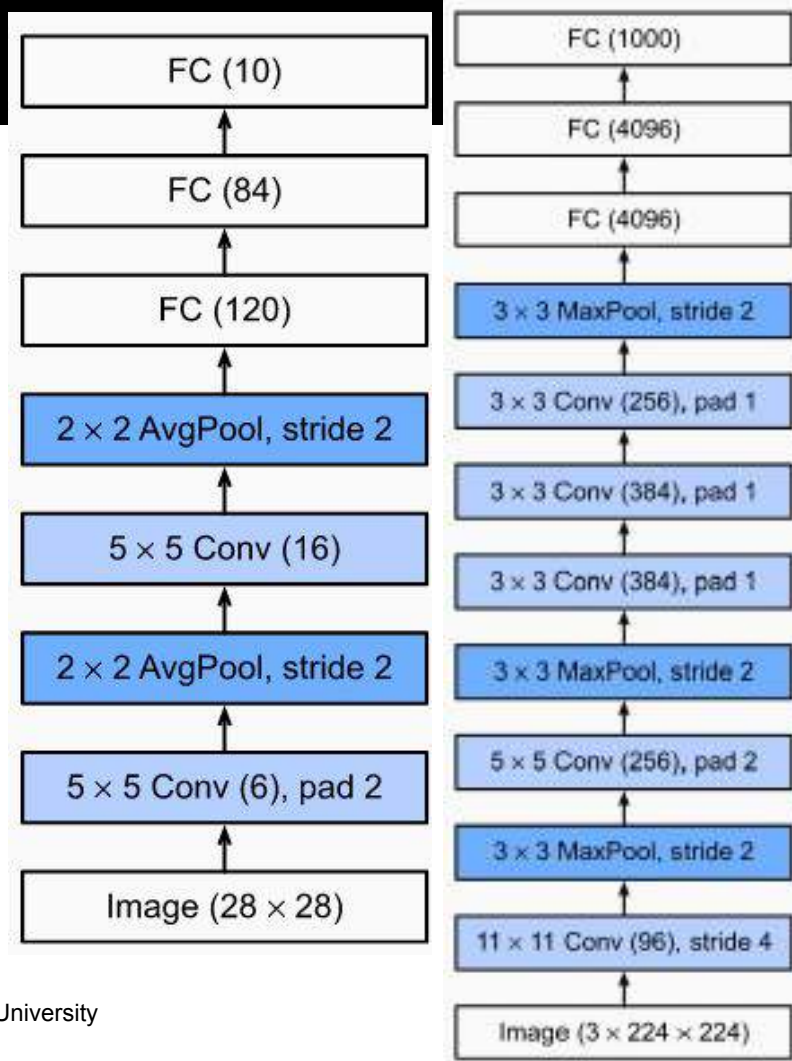
Deeper neural networks allow increasingly complex patterns to be learned, but increases number of computations.

Tradeoff between performance, storage, and runtime as # hidden layers increase



AlexNet

- opened the era of deep learning
- won 2012 ImageNet challenge
- inspired by the LeNet in 1998
- FC = full connected
- 2018 Turing Awards




```

net = nn.Sequential(
    # Here, we use a larger 11 x 11 window to capture objects. At the same
    # time, we use a stride of 4 to greatly reduce the height and width of the
    # output. Here, the number of output channels is much larger than that in
    # LeNet
    nn.Conv2d(1, 96, kernel_size=11, stride=4, padding=1), nn.ReLU(),
    nn.MaxPool2d(kernel_size=3, stride=2),
    # Make the convolution window smaller, set padding to 2 for consistent
    # height and width across the input and output, and increase the number of
    # output channels
    nn.Conv2d(96, 256, kernel_size=5, padding=2), nn.ReLU(),
    nn.MaxPool2d(kernel_size=3, stride=2),
    # Use three successive convolutional layers and a smaller convolution
    # window. Except for the final convolutional layer, the number of output
    # channels is further increased. Pooling layers are not used to reduce the
    # height and width of input after the first two convolutional layers
    nn.Conv2d(256, 384, kernel_size=3, padding=1), nn.ReLU(),
    nn.Conv2d(384, 384, kernel_size=3, padding=1), nn.ReLU(),
    nn.Conv2d(384, 256, kernel_size=3, padding=1), nn.ReLU(),
    nn.MaxPool2d(kernel_size=3, stride=2), nn.Flatten(),
    # Here, the number of outputs of the fully-connected layer is several
    # times larger than that in LeNet. Use the dropout layer to mitigate
    # overfitting
    nn.Linear(6400, 4096), nn.ReLU(), nn.Dropout(p=0.5),
    nn.Linear(4096, 4096), nn.ReLU(), nn.Dropout(p=0.5),
    # Output layer. Since we are using Fashion-MNIST, the number of classes is
    # 10, instead of 1000 as in the paper
    nn.Linear(4096, 10))

```

AlexNet

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) of 1000 classes of objects, 14M images.

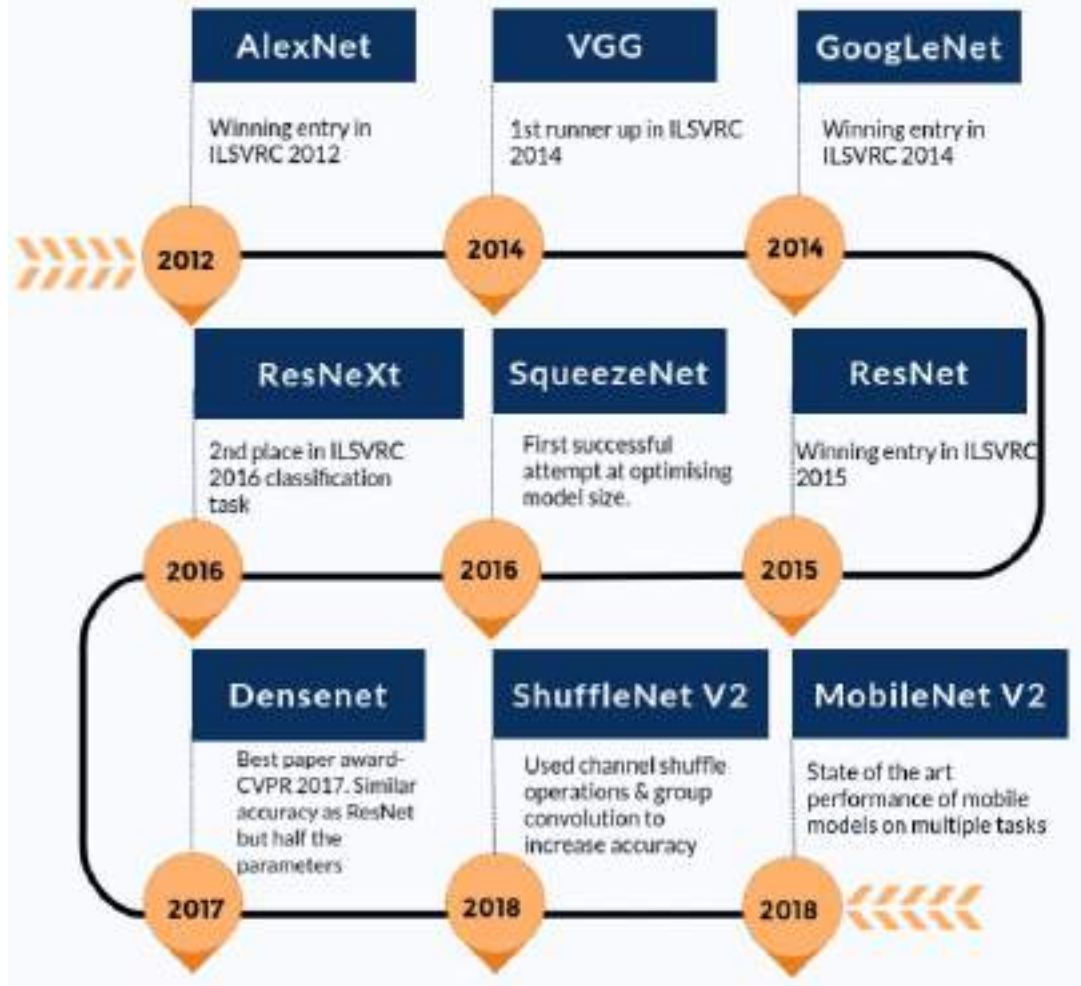
AlexNet: Top-5 error rate (rate of not finding the true label of a given image among its top 5 predictions) of 15.3%.

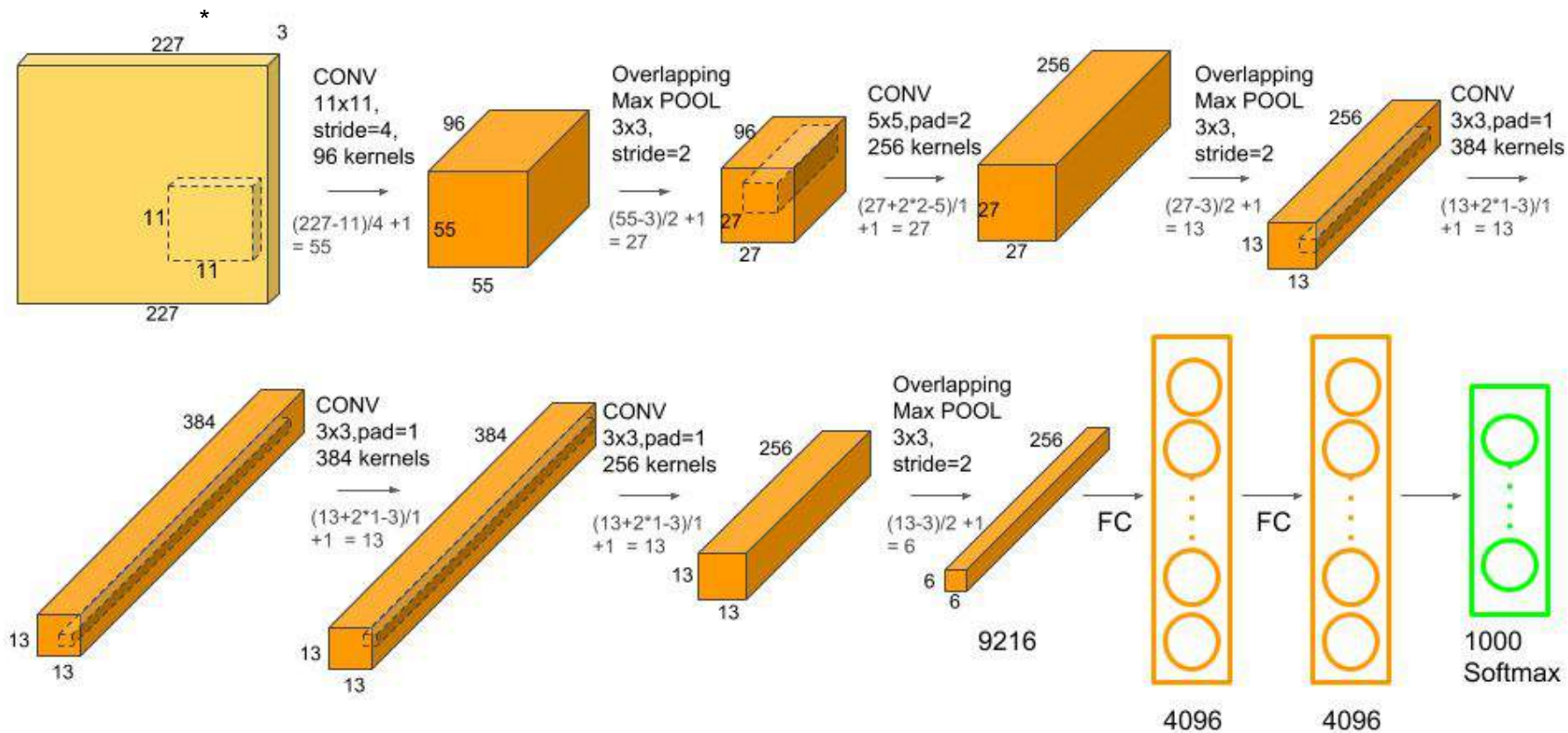
The next best result was 26.2%.

(smaller is better)

VGG: Visual Geometry Group

<https://learnopencv.com/pytorch-for-beginners-image-classification-using-pre-trained-models/>





<https://learnopencv.com/understanding-alexnet/>

*This website says 224 x 224 was an error.

Padding

Input Kernel Output

0	1	2
3	4	5
6	7	8

*

0	1
2	3

=

19	25
37	43

Number of pixels reduced after convolution

$$0 \times 0 + 1 \times 1 + 3 \times 2 + 4 \times 3 = 19$$

$$4 \times 0 + 5 \times 1 + 7 \times 2 + 8 \times 3 = 43$$

Input Kernel Output

0	0	0	0	0
0	0	1	2	0
0	3	4	5	0
0	6	7	8	0
0	0	0	0	0

*

0	1
2	3

=

0	3	8	4
9	19	25	10
21	37	43	16
6	7	8	0

Pad 0 at boundaries

https://d2l.ai/chapter_convolutional-neural-networks/padding-and-strides.html

Max Pooling and Stride

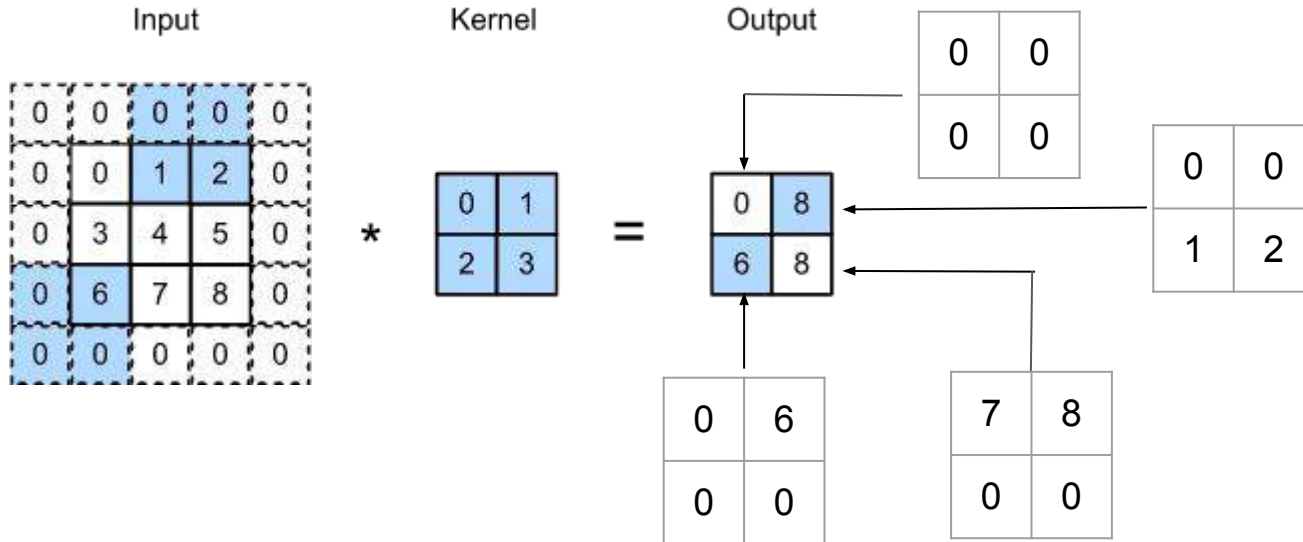
0	1	2
3	4	5
6	7	8

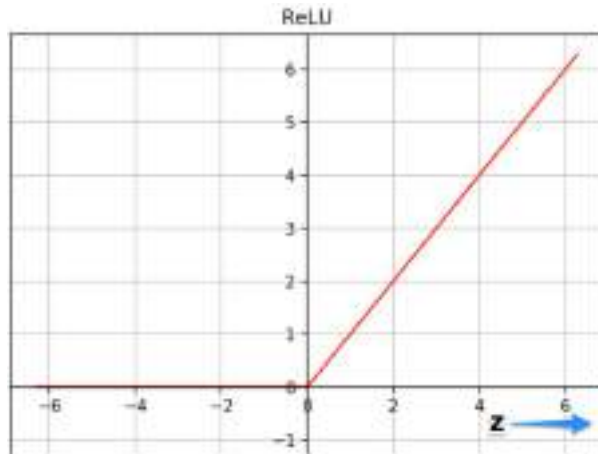
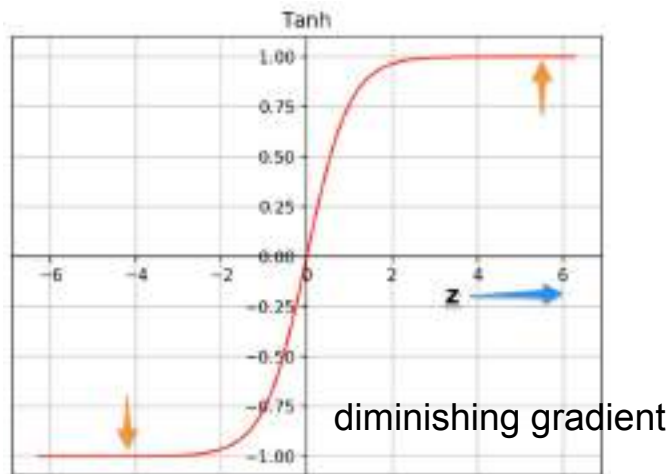
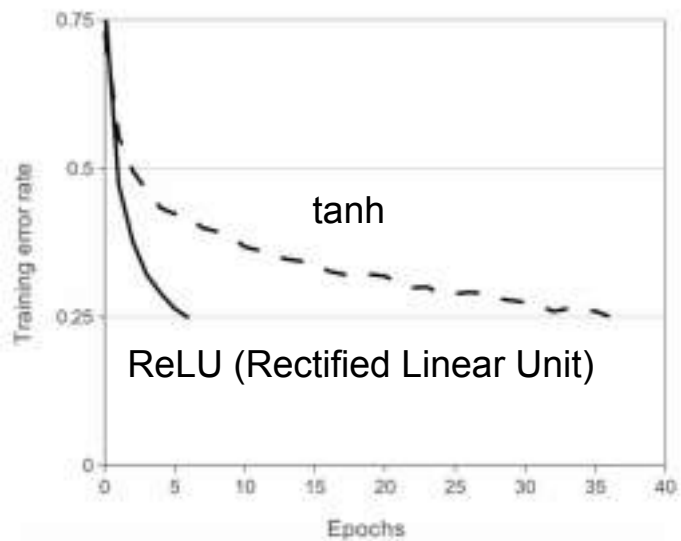
2 x 2 Max
Pooling

4	5
7	8

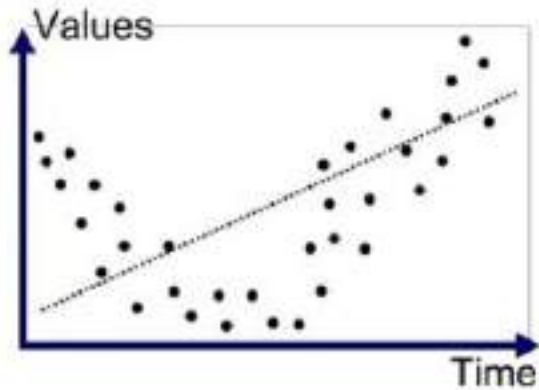
**Max pooling: select the
largest in the window**

stride = 2 horizontally, 3 vertically

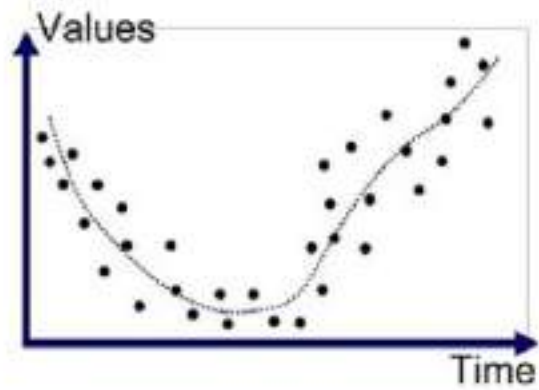




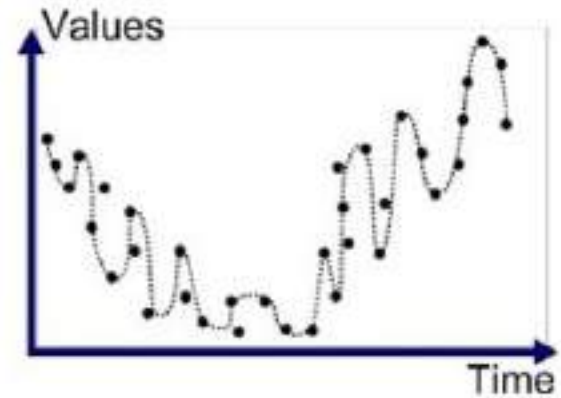
**If the dimension of the machine model is too low: underfitting
too high: overfitting**



Underfitted

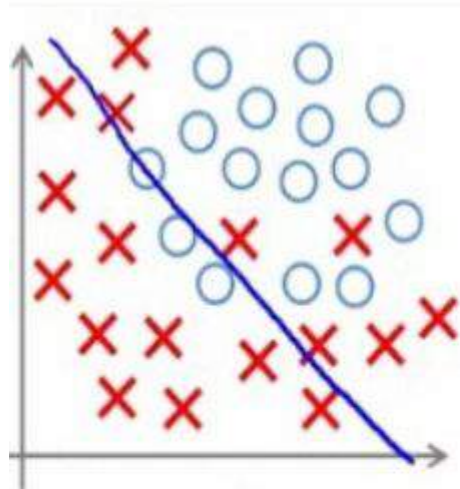


Good Fit/Robust



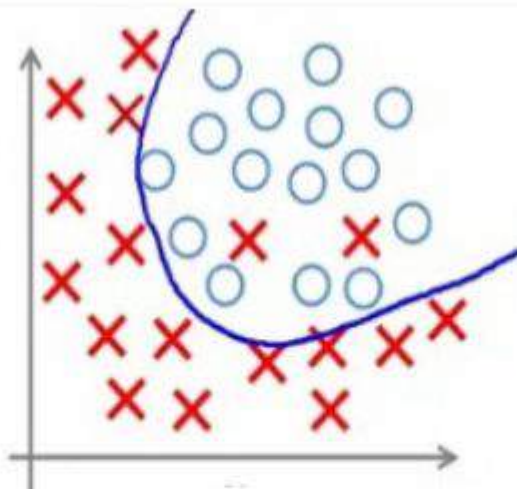
Overfitted

<https://medium.com/greyatom/what-is-underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6803a989c76>

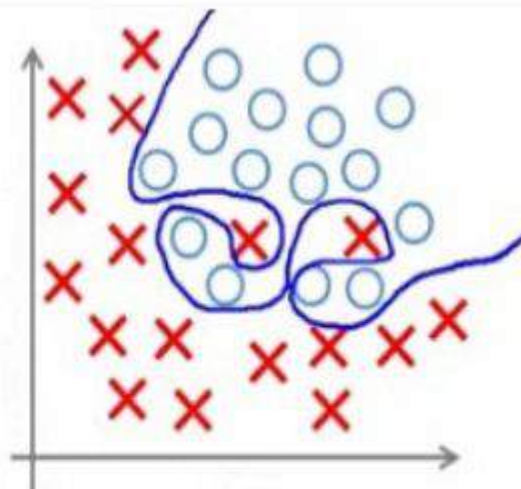


Under-fitting

(too simple to
explain the
variance)



Appropriate-fitting



Over-fitting

(forcefitting -- too
good to be true)



Mirror Image

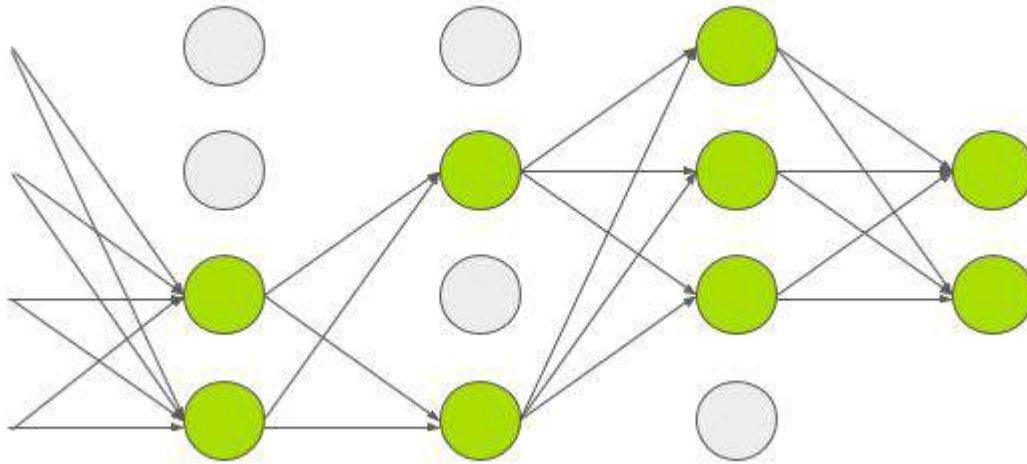


Data Augmentation
(create more training data)
to reduce overfitting



Random Crops





**Dropout: each connection has 50% not used in training
⇒ strengthen the weights among the remaining connections**

```
In [284]: # based on https://learnopencv.com/pytorch-for-beginners-image-classification-using-pre-trained-models/
from torchvision import models
# dir(models)
```

```
In [285]: AlexNet = models.alexnet(pretrained=True)
```

```
In [286]: print(AlexNet)
```

```
AlexNet(
  (features): Sequential(
    (0): Conv2d(3, 64, kernel_size=(11, 11), stride=(4, 4), padding=(2, 2))
    (1): ReLU(inplace=True)
    (2): MaxPool2d(kernel_size=3, stride=2, padding=0, dilation=1, ceil_mode=False)
    (3): Conv2d(64, 192, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
    (4): ReLU(inplace=True)
    (5): MaxPool2d(kernel_size=3, stride=2, padding=0, dilation=1, ceil_mode=False)
    (6): Conv2d(192, 384, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (7): ReLU(inplace=True)
    (8): Conv2d(384, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (9): ReLU(inplace=True)
    (10): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (11): ReLU(inplace=True)
    (12): MaxPool2d(kernel_size=3, stride=2, padding=0, dilation=1, ceil_mode=False)
  )
)
```


List of ImageNet class numbers and names as used in Keras' pre-trained models.

Extracted from https://s3.amazonaws.com/deep-learning-models/image-models/imagenet_class_index.json

```
|  
0, tench  
1, goldfish  
2, great_white_shark  
3, tiger_shark  
4, hammerhead  
5, electric_ray  
6, stingray  
7, cock  
8, hen  
9, ostrich  
10, brambling  
11, goldfinch  
12, house_finch  
13, junco  
14, indigo_bunting  
15, robin  
16, bulbul  
17, jay  
18, magpie
```



```
In [287]: from torchvision import transforms
transform = transforms.Compose([
    transforms.Resize(256),
    transforms.CenterCrop(224),
    transforms.ToTensor(),
    transforms.Normalize(
        mean=[0.485, 0.456, 0.406],
        std=[0.229, 0.224, 0.225]
    )])
```



```
In [289]: from google.colab import drive # correction from https://le
re-trained-models/
drive.mount('/content/drive', force_remount = True)
```

Mounted at /content/drive

```
In [290]: !ls # in Colab, can run Linux command after !
drive    sample_data
```

```
In [291]: from PIL import Image
img = Image.open('drive/MyDrive/Colab Notebooks/dog.jpg')
img
```

```
import torch
img_t = transform(img)
batch_t = torch.unsqueeze(img_t, 0)
AlexNet.eval()
output = AlexNet(batch_t)
print(output.shape)
```

```
torch.Size([1, 1000])
```

```
with open('drive/MyDrive/Colab Notebooks/imagenet_classes.txt') as f:
    # skip first four lines, the txt file has four additional lines at the top
    for ind in range(4):
        lines = f.readline()
    classes = [line.strip() for line in f.readlines()]

print(classes[0:10])
```

```
['0, tench', '1, goldfish', '2, great_white_shark', '3, tiger_shark', '4, ham
cock', '8, hen', '9, ostrich']
```

```
_, index = torch.max(output, 1)
percentage = torch.nn.functional.softmax(output, dim=1)[0] * 100
print(classes[index[0]], percentage[index[0]].item()) # correction :
-classification-using-pre-trained-models/
```

208, Labrador_retriever 41.58515930175781

```
_, indices = torch.sort(output, descending=True)
[(classes[idx], percentage[idx].item()) for idx in indices[0][:5]]

[('208, Labrador_retriever', 41.58515930175781),
 ('207, golden_retriever', 16.591659545898438),
 ('176, Saluki', 16.286876678466797),
 ('172, whippet', 2.853910207748413),
 ('173, Ibizan_hound', 2.3924787044525146)]
```



```
In [296]: img2 = Image.open('drive/MyDrive/Colab Notebooks/strawberries.jpg')
img2
```



```
img_t2 = transform(img2)
batch_t2 = torch.unsqueeze(img_t2, 0)
AlexNet.eval()
output2 = AlexNet(batch_t2)
_, index = torch.max(output2, 1)
percentage = torch.nn.functional.softmax(output2, dim=1)[0] * 100
_, indices = torch.sort(output2, descending=True)
[(classes[idx], percentage[idx].item()) for idx in indices[0][:5]]
```

```
[('949, strawberry', 99.99370574951172),
 ('956, custard_apple', 0.000983853824436646),
 ('954, banana', 0.0008134939707815647),
 ('950, orange', 0.0007900753989815712),
 ('509, confectionery', 0.000582081382162869)]
```

PyTorch (by Facebook)

PyTorch is a software library for machine learning



C

more control



Python

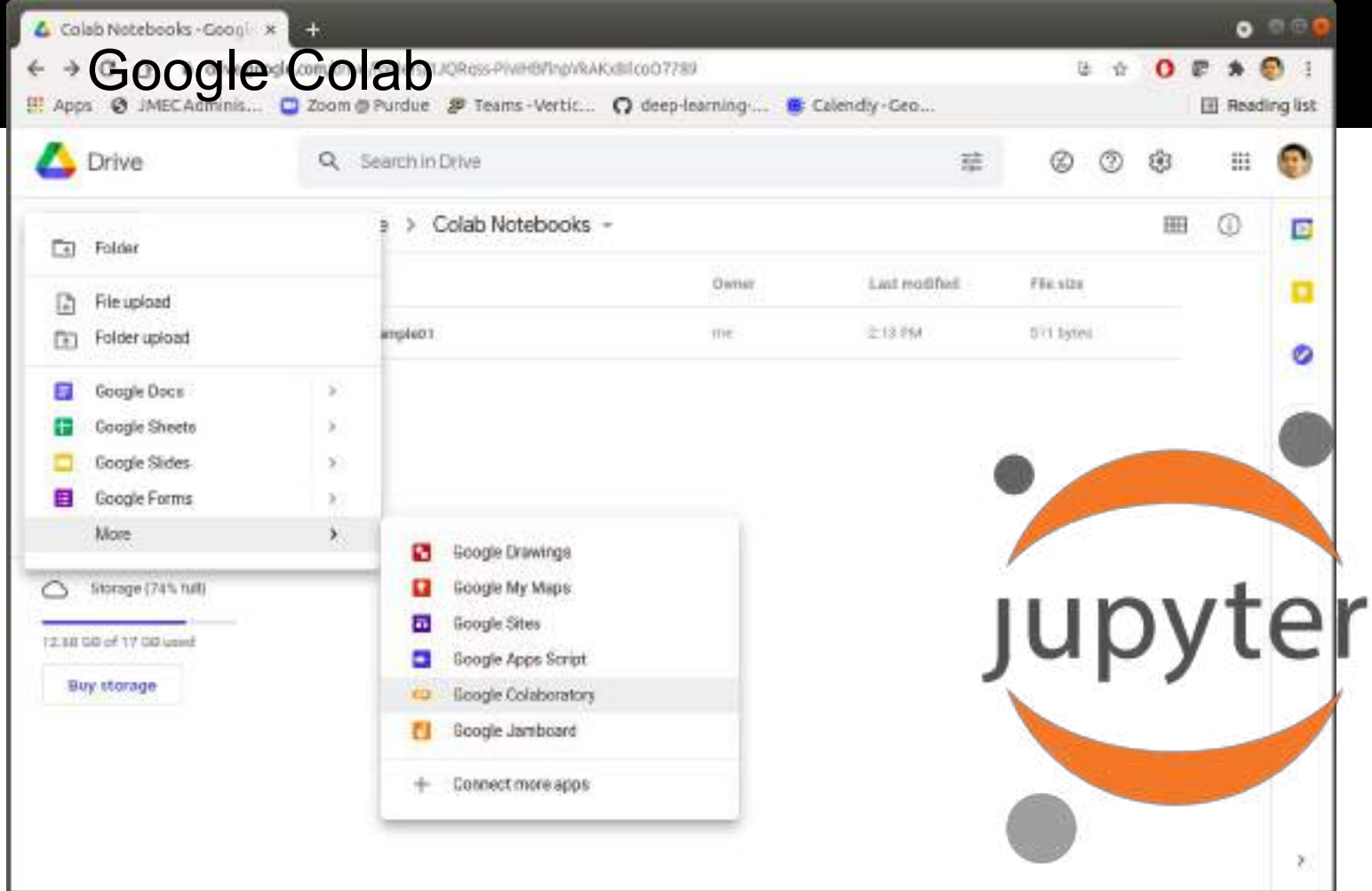


PyTorch

more automation

<https://firstaerosquadron.com/2015/09/23/cockpit-evolution-from-the-beginning-to-present/>

Google Colab




Enable GPU



runtime - change runtime type

Notebook settings

Hardware accelerator

GPU 

To get the most out of Colab, avoid using a GPU unless you need one. [Learn more](#)

☐ Omit code cell output when saving this notebook

Cancel

Save

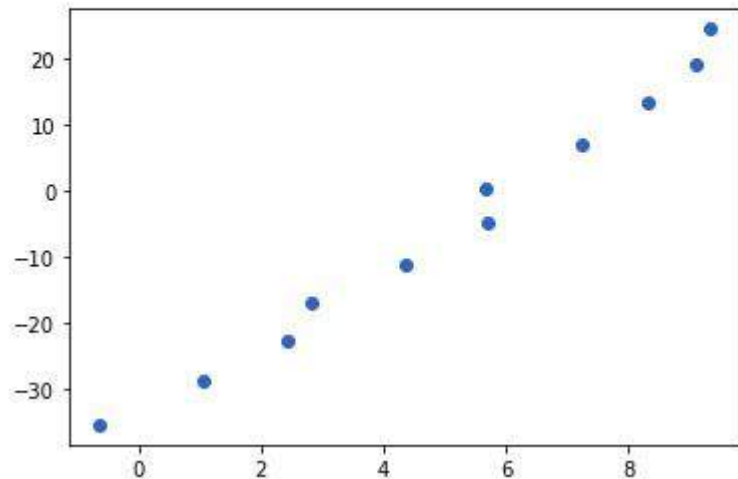
```
import torch
print (torch.cuda.is_available()) # Is GPU available?
```

True

PyTorch Example

```
import random
x_values = [i + (random.random() - random.random()) for i in range(11)]
y_values = [(6 * i + (random.random() - random.random()) - 35) for i in range(11)]

import matplotlib.pyplot as plt
plt.plot(x_values, y_values, 'o')
plt.show()
```



```

import numpy as np
x_train = np.array(x_values, dtype=np.float32)
x_train = x_train.reshape(-1, 1) # convert to a column vector
y_train = np.array(y_values, dtype=np.float32)
y_train = y_train.reshape(-1, 1)

```

```

inputDim = 1          # takes variable 'x'
outputDim = 1         # takes variable 'y'
learningRate = 0.01
epochs = 1000
model = linearRegression(inputDim, outputDim)
model.cuda()

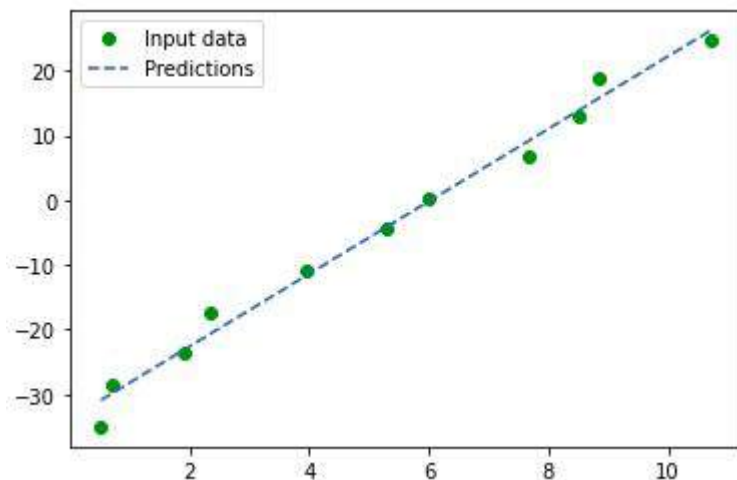
```

```

criterion = torch.nn.MSELoss()
optimizer = torch.optim.SGD(model.parameters(), lr=learningRate)
for epoch in range(epochs):
    inputs = Variable(torch.from_numpy(x_train).cuda())
    labels = Variable(torch.from_numpy(y_train).cuda())
    optimizer.zero_grad()
    outputs = model(inputs)
    loss = criterion(outputs, labels)
    print(loss)
    loss.backward()
    optimizer.step()
    print('epoch {}, loss {}'.format(epoch, loss.item()))

```

```
with torch.no_grad(): # we don't need gradients in the testing phase
    predicted = model(torch.from_numpy(x_train).cuda()).cpu().data.numpy()
    print(predicted)
plt.clf()
plt.plot(x_train, y_train, 'go', label='Input data')
plt.plot(x_train, predicted, '--', label='Predictions')
plt.legend(loc='best')
plt.show()
```



Set up Your Own Raspberry PI

Operating System X

**Raspberry Pi OS (32-bit)**

A port of Debian with the Raspberry Pi Desktop (Recommended)

Released: 2020-08-20

Online - 1.1 GB download

**Raspberry Pi OS (other)**

Other Raspberry Pi OS based images

**LibreELEC**

A Kodi Entertainment Center distribution

**Ubuntu**

Choose from Ubuntu Core and Server images

**RetroPie**

Turn your Raspberry Pi into a retro gaming machine



<https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up/2>



14:37

Raspberry Pi Configuration



System

Display

Interfaces

Performance

Localisation

Camera:	<input type="radio"/> Enable	<input checked="" type="radio"/> Disable
SSH:	<input checked="" type="radio"/> Enable	<input type="radio"/> Disable
VNC:	<input type="radio"/> Enable	<input checked="" type="radio"/> Disable
SPI:	<input type="radio"/> Enable	<input checked="" type="radio"/> Disable
I2C:	<input type="radio"/> Enable	<input checked="" type="radio"/> Disable
Serial Port:	<input type="radio"/> Enable	<input checked="" type="radio"/> Disable
Serial Console:	<input checked="" type="radio"/> Enable	<input type="radio"/> Disable
1-Wire:	<input type="radio"/> Enable	<input checked="" type="radio"/> Disable
Remote GPIO:	<input type="radio"/> Enable	<input checked="" type="radio"/> Disable

Cancel

OK

Raspberry Pi Configuration



System

Display

Interfaces

Performance

Localisation

Locale:	Set Locale...
Timezone:	Set Timezone...
Keyboard:	Set Keyboard...
WiFi Country:	Set WiFi Country...

Cancel

OK

```
pi@raspberrypi:~ $ ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether dc:a6:32:d9:b1:1c txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.4.34 netmask 255.255.252.0 broadcast 192.168.7.255
    inet6 fd84:9c47:c41c:1:3dd6:a366:b5b1:bfac prefixlen 64 scopeid 0x0<gl
obal>
    inet6 fe80::8813:86cf:6ba2:f56c prefixlen 64 scopeid 0x20<link>
    ether dc:a6:32:d9:b1:1d txqueuelen 1000 (Ethernet)
    RX packets 29491 bytes 37135414 (35.4 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 9957 bytes 1726630 (1.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```



```
[(Dell) ~/] ssh pi@192.168.4.34
Linux raspberrypi 5.10.17-v7l+ #1414 SMP Fri Apr 30 13:20:47 BST 2021 armv7l
```

The programs included with the Debian GNU/Linux system are free software; the exact distribution terms for each program are described in the individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent permitted by applicable law.

Last login: Wed Aug 18 15:03:44 2021 from 192.168.4.24

```
pi@raspberrypi:~$
```

```
pi@raspberrypi:~$ more /proc/cpuinfo
processor       : 0
model name     : ARMv7 Processor rev 3 (v7l)
BogoMIPS      : 108.00
Features      : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idivt
vfpd32 lpae evtstrm crc32
CPU implementer : 0x41
CPU architecture: 7
CPU variant    : 0x0
CPU part       : 0xd08
CPU revision   : 3

processor       : 1
model name     : ARMv7 Processor rev 3 (v7l)
BogoMIPS      : 108.00
Features      : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idivt
vfpd32 lpae evtstrm crc32
CPU implementer : 0x41
CPU architecture: 7
CPU variant    : 0x0
CPU part       : 0xd08
CPU revision   : 3
```

```
pi@raspberrypi:~ $ more /proc/meminfo
```

```
MemTotal:      8064284 kB
MemFree:       7437852 kB
MemAvailable:  7759488 kB
Buffers:       23248 kB
Cached:        525000 kB
SwapCached:    0 kB
Active:        205484 kB
Inactive:      367308 kB
Active(anon):   368 kB
Inactive(anon): 32816 kB
Active(file):   205116 kB
Inactive(file): 334492 kB
Unevictable:    16 kB
Mlocked:       16 kB
HighTotal:     7458816 kB
HighFree:      6899376 kB
LowTotal:      605468 kB
LowFree:       538476 kB
SwapTotal:     102396 kB
SwapFree:      102396 kB
Dirty:         0 kB
Writeback:     0 kB
AnonPages:     24572 kB
Mapped:        33448 kB
```

```
pi@raspberrypi:~ $ df -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/root	118G	4.3G	109G	4%	/
devtmpfs	3.8G	0	3.8G	0%	/dev
tmpfs	3.9G	0	3.9G	0%	/dev/shm
tmpfs	3.9G	8.5M	3.9G	1%	/run
tmpfs	5.0M	4.0K	5.0M	1%	/run/lock
tmpfs	3.9G	0	3.9G	0%	/sys/fs/cgroup
/dev/mmcblk0p1	253M	48M	205M	19%	/boot
tmpfs	788M	0	788M	0%	/run/user/1000

Install and Enable Firewall

Prevent unauthorized accesses

- `sudo apt install ufw`
- `sudo ufw enable`
- `sudo ufw allow 22 (ssh)`
- `sudo ufw status`