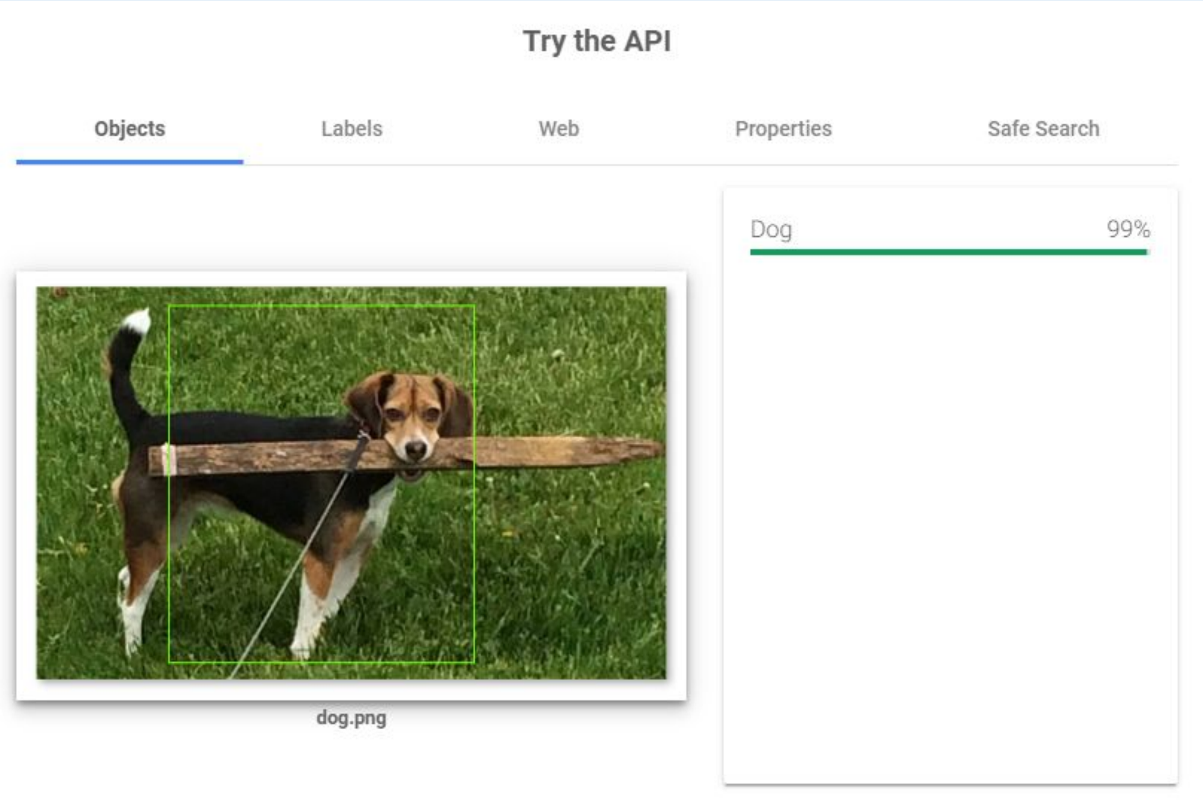
Deep Learning for Computer Vision in Python

08/27/2019

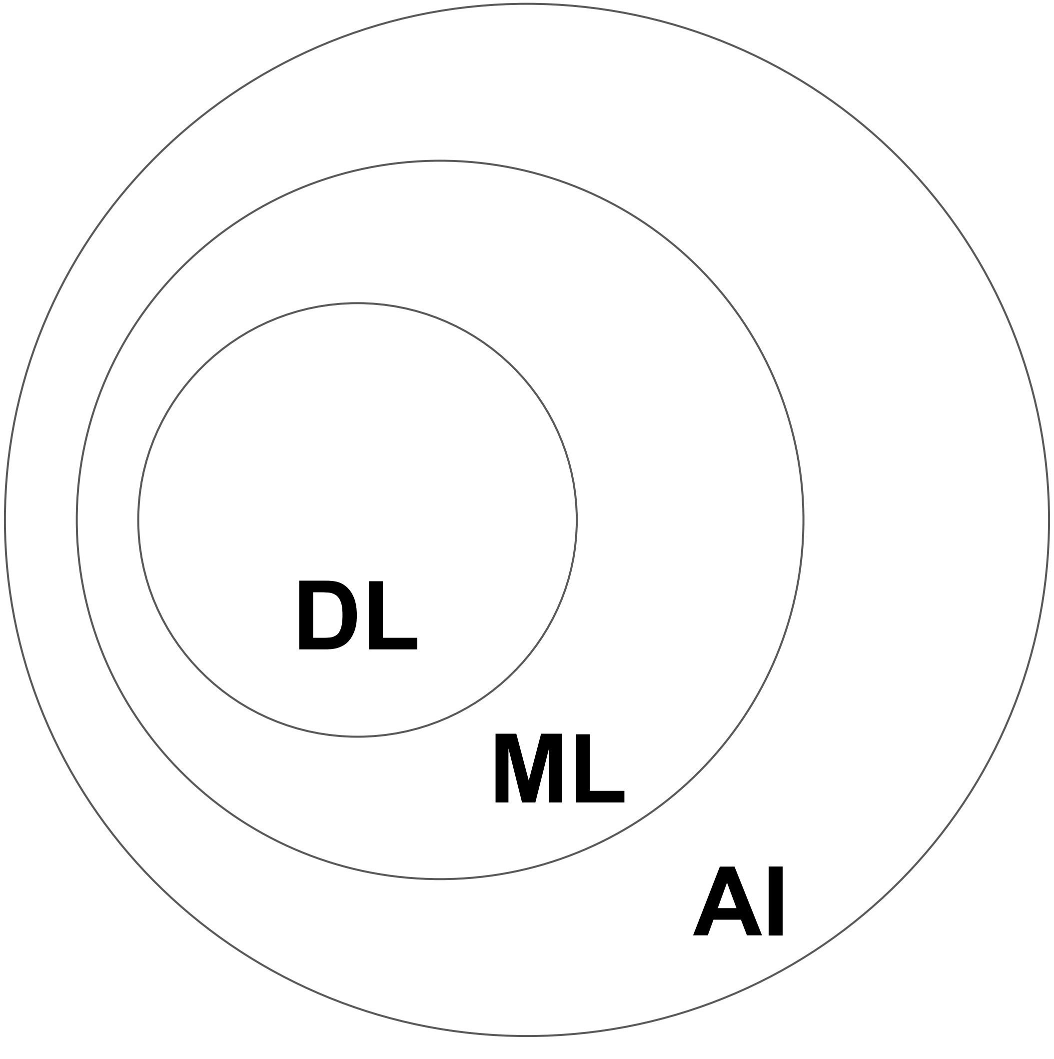
Test an existing model

https://cloud.google.com/vision/docs/drag-and-drop



Discussions

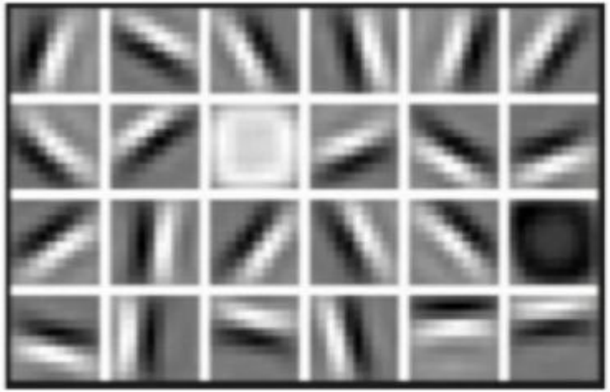
Some theory


DL
ML
AI

ML	DL
Hand defined algorithms to extract features of an image (e.g., shape, texture, color)	**The features are automatically learned from the training process**

DL

Low Level Features



Lines & Edges

Mid Level Features

Eyes & Nose & Ears

High Level Features

Facial Structure

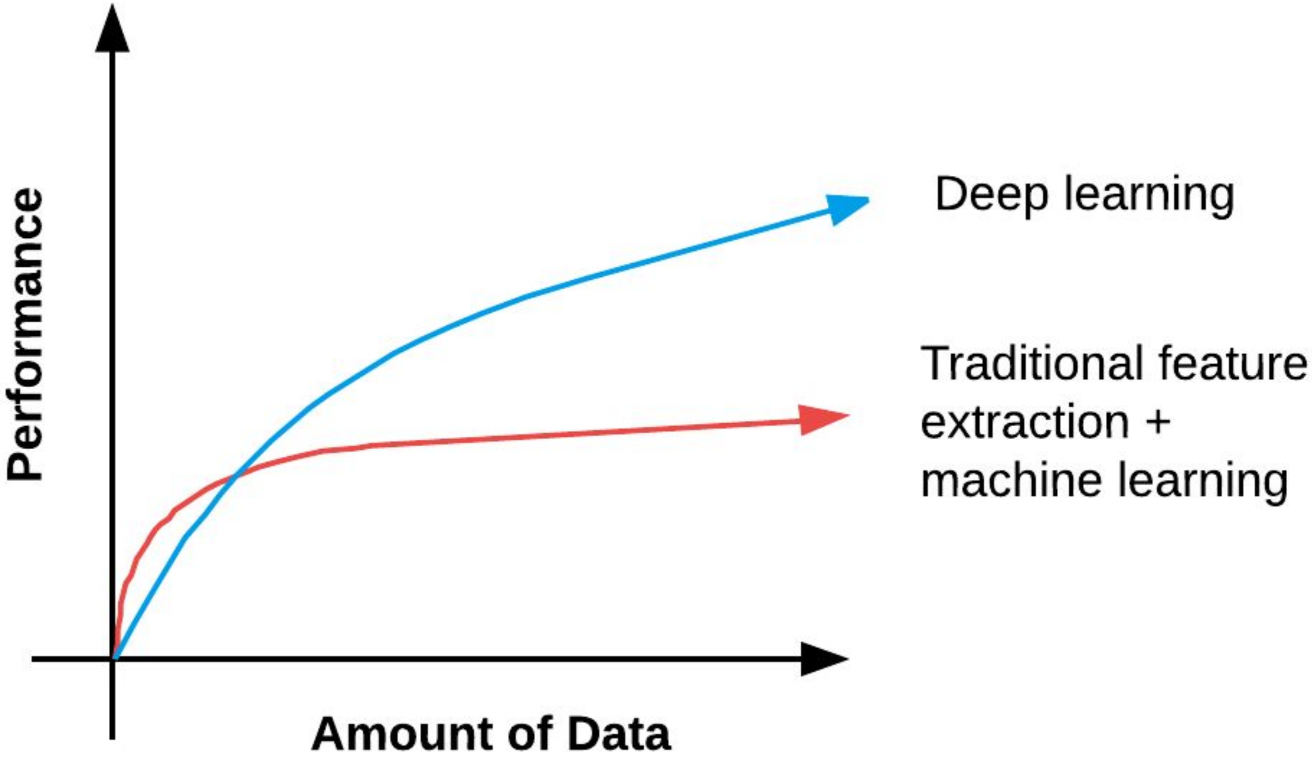

Performance
Deep learning
Traditional feature extraction + machine learning
Amount of Data

Image Classification

- <u>Goal</u>: assign a label to an image from predefined set of categories

```
categories = {cat, dog, panda}
```

dog: 95%
cat: 4%
panda: 1%

The Semantic Gap

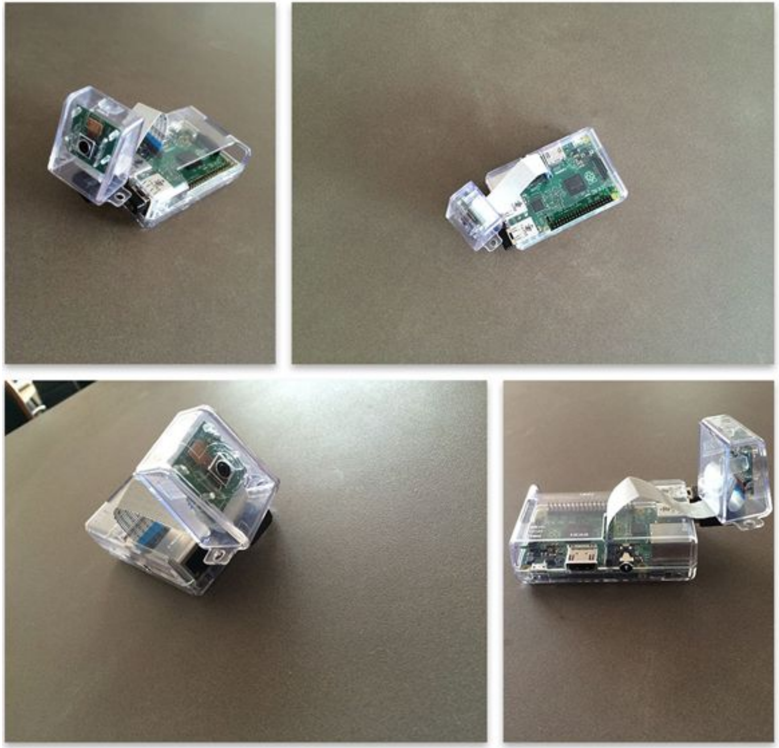
151	121	1	93	165	204	14	214	28	235
62	67	17	234	27	1	221	37	189	141
20	168	155	113	178	228	25	130	139	221
236	136	158	230	10	5	165	17	30	155
174	148	93	70	95	106	151	10	160	214
103	126	58	16	138	136	98	202	42	233
235	103	52	37	94	104	173	86	223	113
212	15	179	139	48	232	194	46	174	37
119	81	241	172	95	170	29	210	22	194
129	19	33	253	229	5	152	233	52	44
88	200	194	185	140	200	223	190	164	102
113	16	220	215	143	104	247	29	97	203
9	210	102	246	75	9	158	104	184	129
124	52	76	148	249	107	65	216	187	181
6	251	52	208	46	65	185	38	77	240
150	194	28	206	148	197	208	28	74	93
33	183	248	153	168	205	146	100	254	218
130	53	128	212	61	226	201	110	140	183
165	246	22	102	151	213	40	138	8	93
152	251	101	230	23	162	70	238	75	24
187	105	152	83	167	98	125	180	136	121
139	197	55	209	28	124	208	208	104	40
123	19	144	223	62	253	202	108	47	242
220	144	31	16	136	123	227	62	183	163

29	142	142	75	22	109	111	28	6	5
137	168	41	206	100	70	219	127	114	191
205	154	226	14	89	86	242	67	203	15
247	47	128	123	253	229	181	251	232	28
68	75	24	99	93	63	215	222	102	180
206	246	85	103	215	3	62	64	77	216
126	80	165	149	196	75	186	60	179	193
44	253	164	253	14	216	175	30	46	254
137	23	33	203	241	21	144	63	244	188
32	214	142	121	249	109	99	232	183	71
45	36	152	27	190	137	61	1	237	247
1	14	241	70	2	30	151	67	169	205
32	80	102	32	99	169	91	166	73	214
186	219	9	203	209	240	40	249	119	122
177	252	38	203	119	0	217	139	139	157
154	145	49	251	150	185	235	23	230	156
157	168	223	60	247	118	5	180	16	206
102	208	195	246	140	138	54	191	139	79
17	233	85	169	166	24	49	40	160	97
84	242	247	144	203	3	19	24	198	88
67	67	185	98	123	106	168	105	127	153
37	113	214	252	203	80	146	211	7	16
142	241	66	86	214	133	146	253	189	200
67	215	174	111	189	54	144	56	59	163

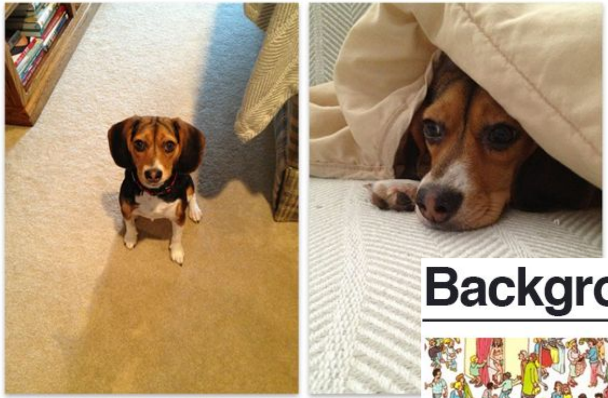
How we can describe the image of a beach?

Challenges: factors of variation (1)

Viewpoint Variation



Occlusion Variation



Background Clutter

Deformation

Challenges: factors of variation (2)

Scale Variation

Illumination Variation

Intra-class Variation

Overwhelming?

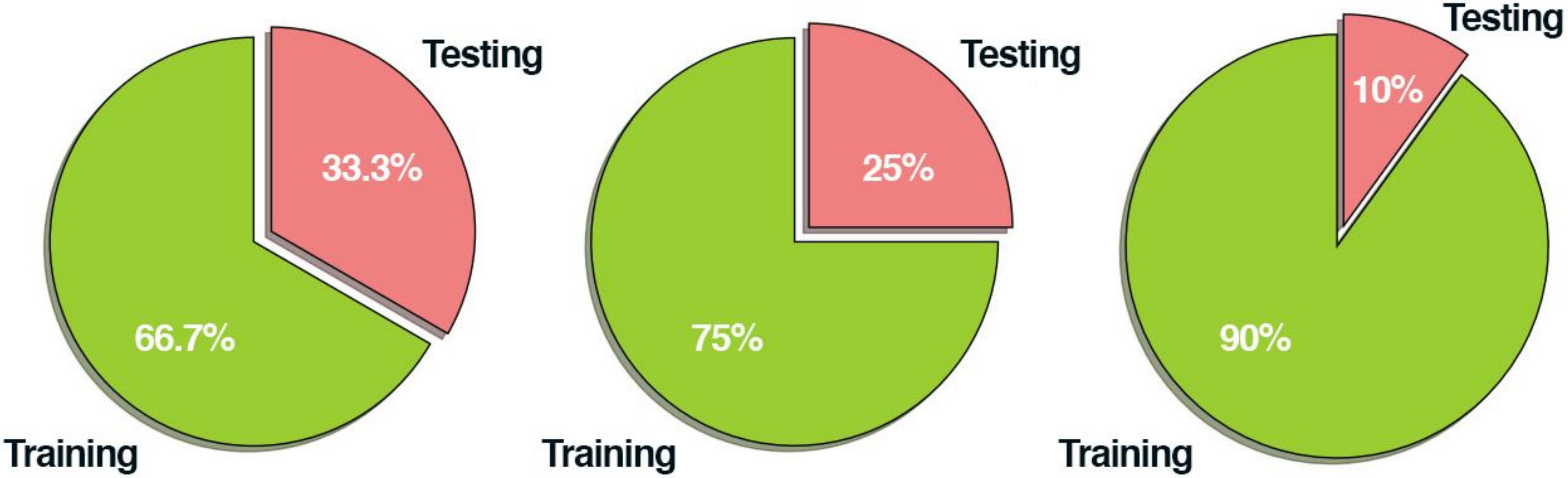
Frame your problem

Deep Learning Classification Pipeline

1. Gather Your Dataset: images + labels

Deep Learning Classification Pipeline

1. Gather Your Dataset: images + labels
2. Split Your Dataset



Deep Learning Classification Pipeline

1. Gather Your Dataset: images + labels
2. Split Your Dataset
3. Train Your Network

Deep Learning Classification Pipeline

1. Gather Your Dataset: images + labels
2. Split Your Dataset
3. Train Your Network
4. Evaluate

Classify Handwritten Digits 0-9

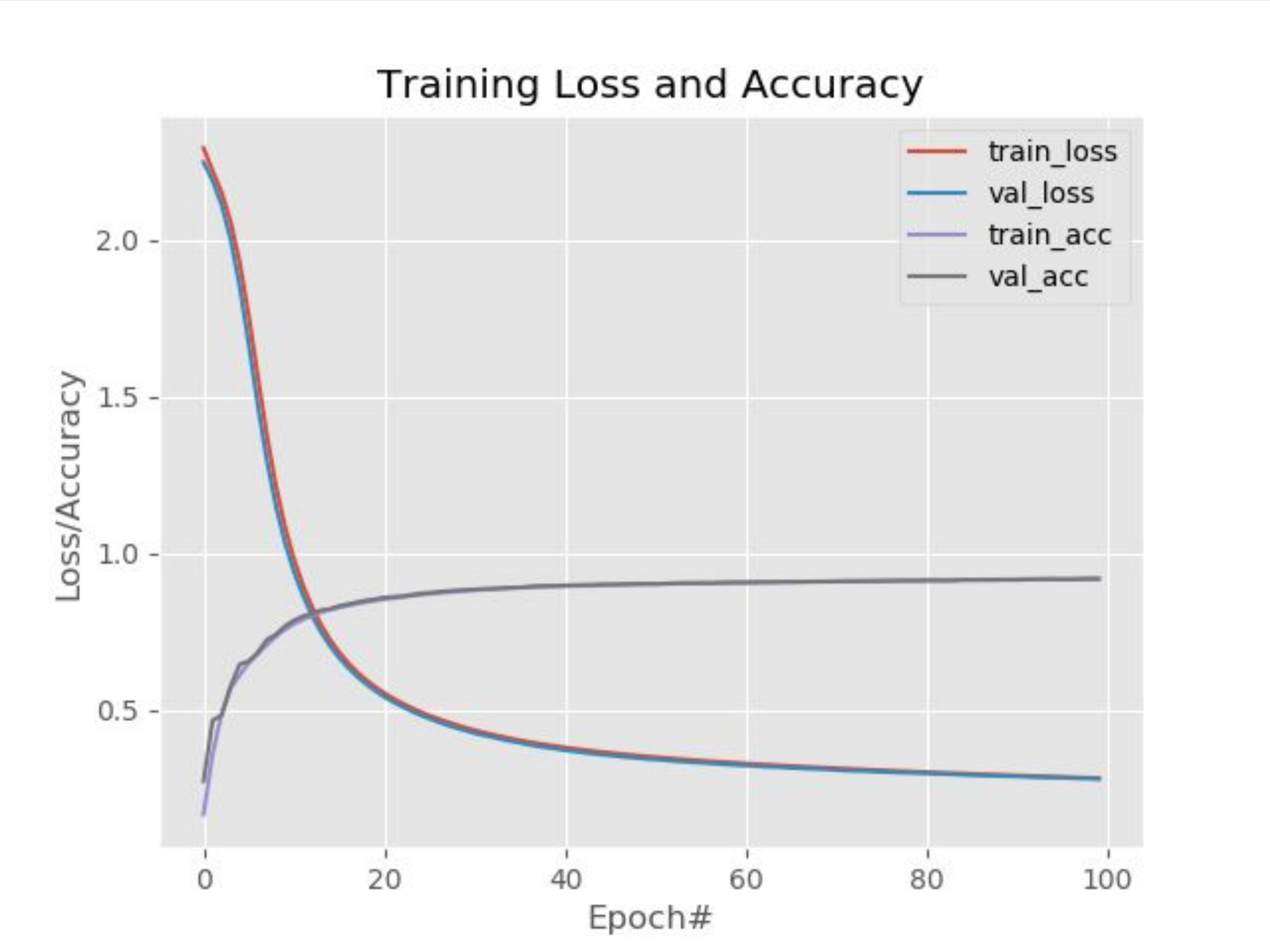
https://bit.ly/2NyFze8

Discussion

```
[INFO] evaluating network...
              precision    recall  f1-score   support

           0       0.95      0.97      0.96      1683
           1       0.93      0.98      0.95      1958
           2       0.91      0.90      0.91      1762
           3       0.91      0.89      0.90      1862
           4       0.92      0.93      0.92      1722
           5       0.88      0.87      0.87      1539
           6       0.94      0.96      0.95      1675
           7       0.93      0.92      0.93      1821
           8       0.90      0.87      0.89      1751
           9       0.90      0.89      0.89      1727

    accuracy                           0.92     17500
   macro avg       0.92      0.92      0.92     17500
weighted avg       0.92      0.92      0.92     17500
```

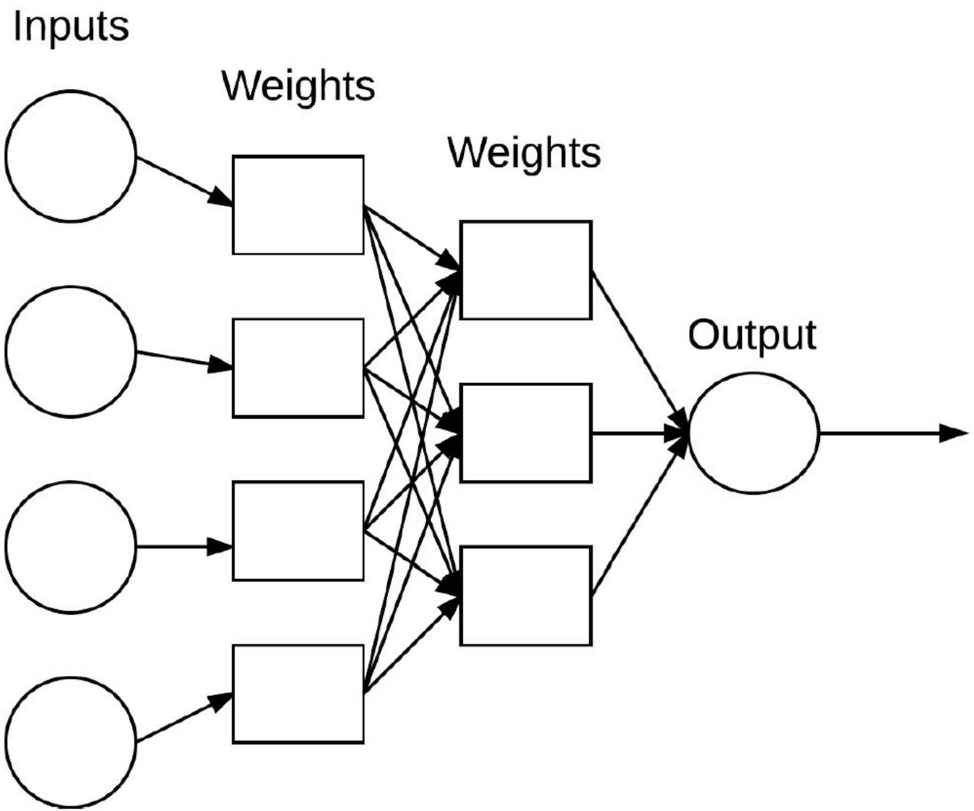


Training Loss and Accuracy
train_loss
val_loss
train_acc
val_acc
Loss/Accuracy
2.0
1.5
1.0
0.5
0
20
40
60
80
100
Epoch#

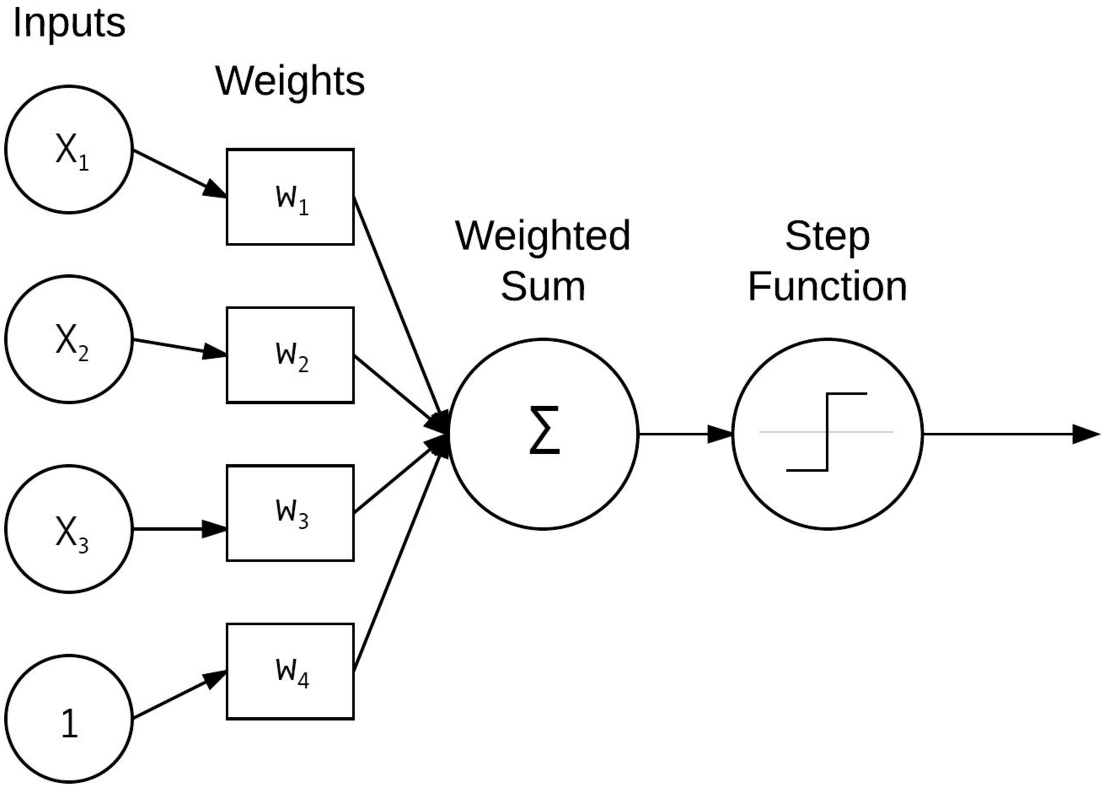
Some theory

Neural Network Fundamentals

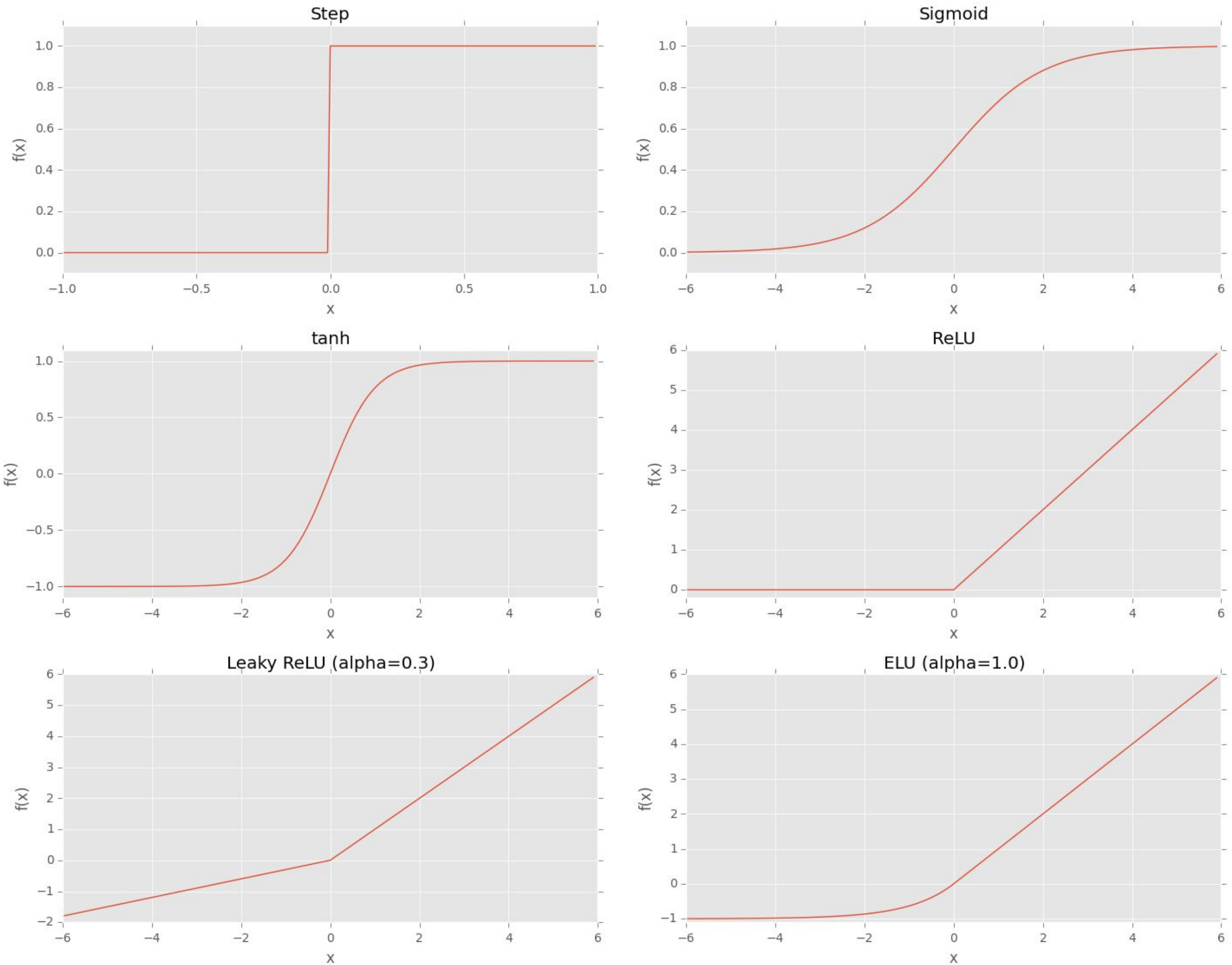
Neural Network (NN) Architecture



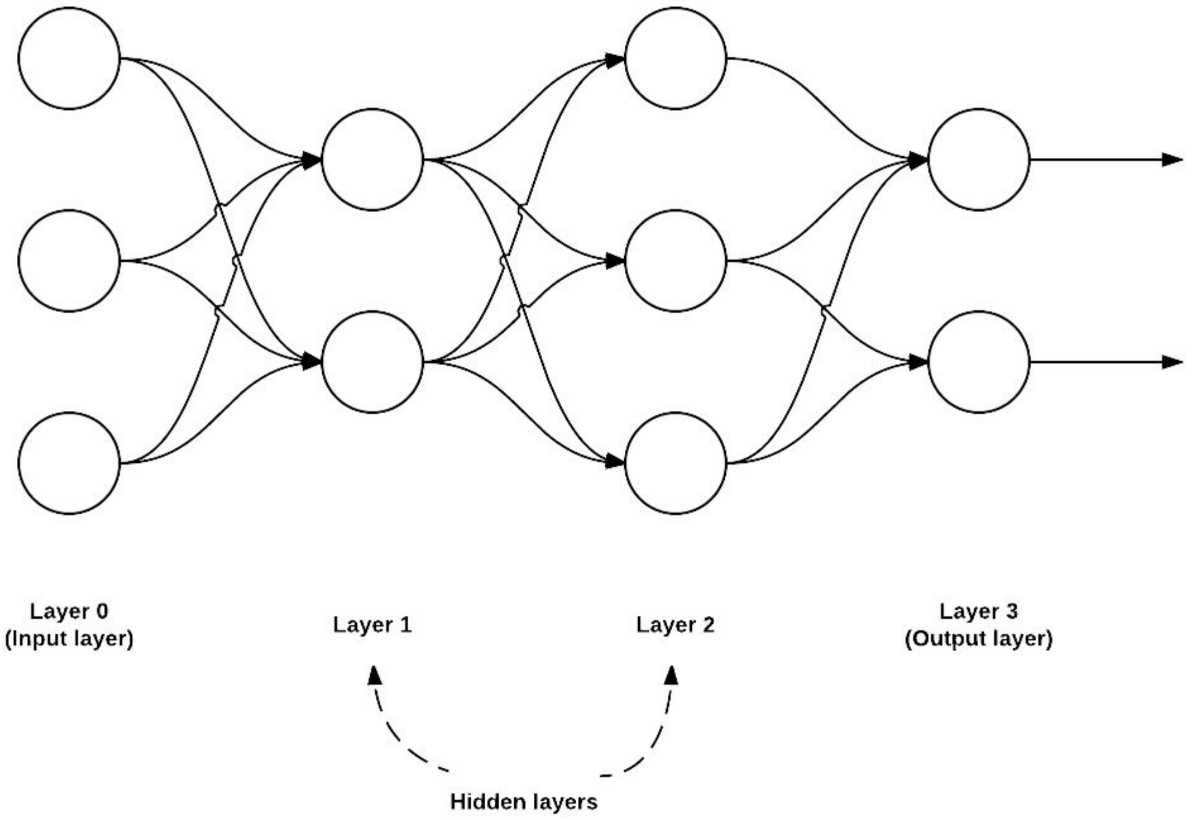
NN: Weighted Sum



Activation Functions



Feedforward Network Architectures



Backpropagation

1. **Forward pass**: The inputs are passes through the network and the predictions are obtained (propagation phase)

2. **Backward pass**: We computer the gradient of the loss function at the final layer (i.e., predictions layer) of the network and use this gradient to recursively apply the chain rule to update the weights in our network (weight update phase)

What are the 4 ingredients of a NN Recipe?

Ingredients of a NN Recipe

Ingredients List

- 5 Roma tomatoes
- 4 limes
- 2 medium avocados
- 1/2 - 3/4 of a bell pepper (not green)
- Shrimp: I used 12 medium shrimp, because that's what we had left in the bag.
- 2 Persian cucumbers (any will work)
- 2 cloves of garlic
- 1/3 c purple onion
- Scallops: I used 11 because that's what we had left in the bag.
- Cilantro, salt & pepper (to taste)

The REAL Ingredients of a NN Recipe

The Four ingredients of a NN Recipe

1. Dataset

The Four ingredients of a NN Recipe

1. Dataset
2. Model/Architecture

The Four ingredients of a NN Recipe

1. Dataset
2. Model/Architecture
3. Loss Function

The Four ingredients of a NN Recipe

1. Dataset
2. Model/Architecture
3. Loss Function
4. Optimization Method

Convolutional Neural Networks

1. **Traditional Foreforward Networks:**
   - Each neuron in the input layer is connected to every output neuron in the next layer (*Fully Connected (FC)* layer)

2. **Convolutional Neural Networks**
   - We don’t use FC layers until the very last layer(s) in the network

CNN may learn to:

1. Detect edges from raw pixel data in the first layer

2. Use these edges to detect shapes (e.g., blobs) in the second layer

3. Use these shapes to detect higher level features such as facial structures, part of a car, etc. in the highest layers of the network.

CNN - main benefits

- Local invariance
- Compositionality

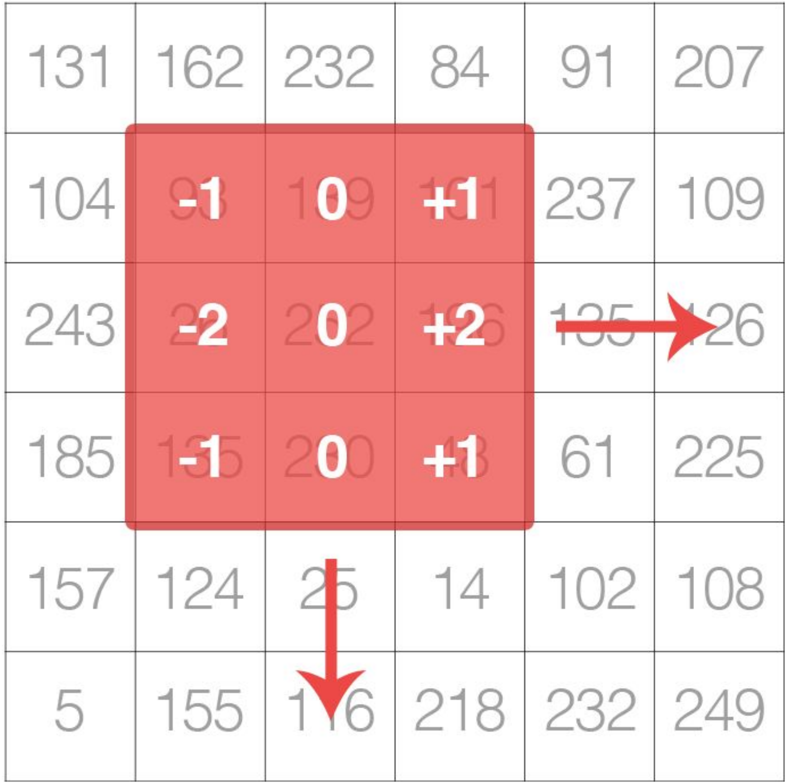
Understanding Convolutions

Answer, in teams, to the following questions (10 min):

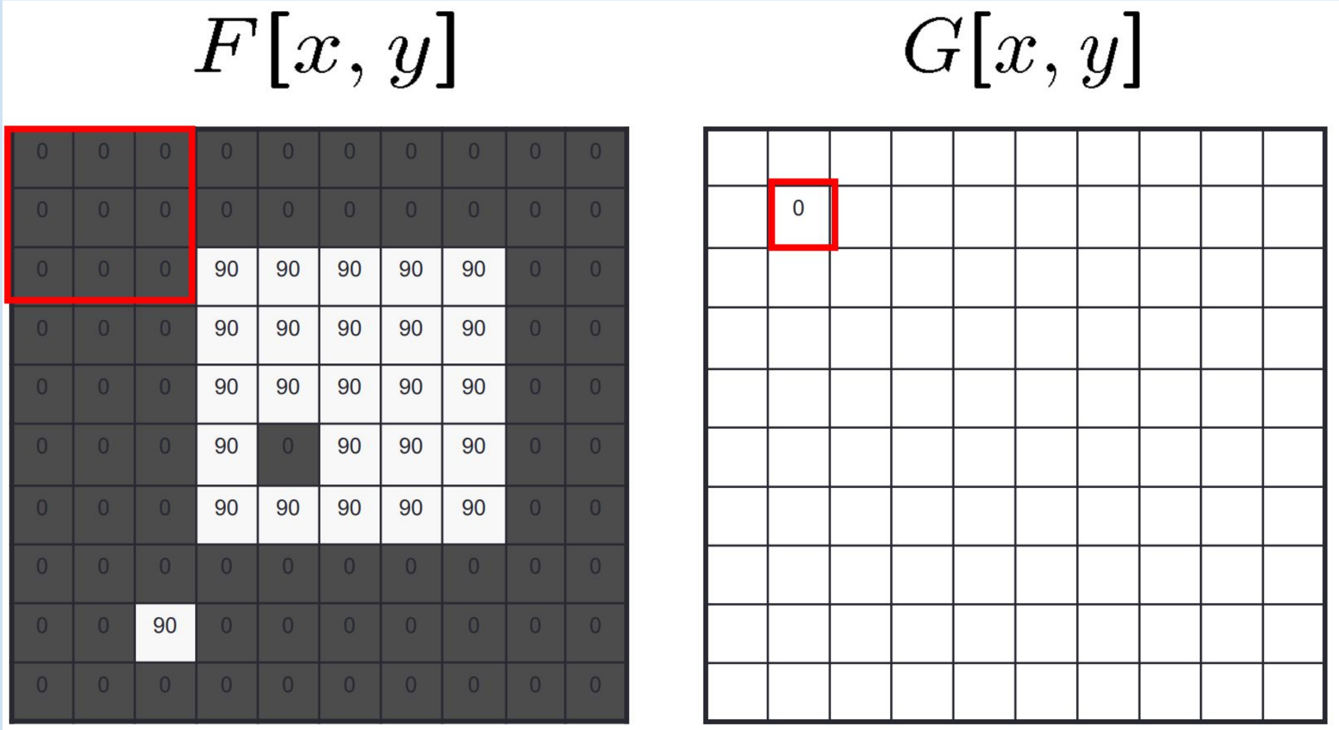
1. What are image convolutions?
2. What do they do?
3. Why do we use them?
4. How to we apply them to images?
5. **What role do convolutions play in deep learning?**

Discussion

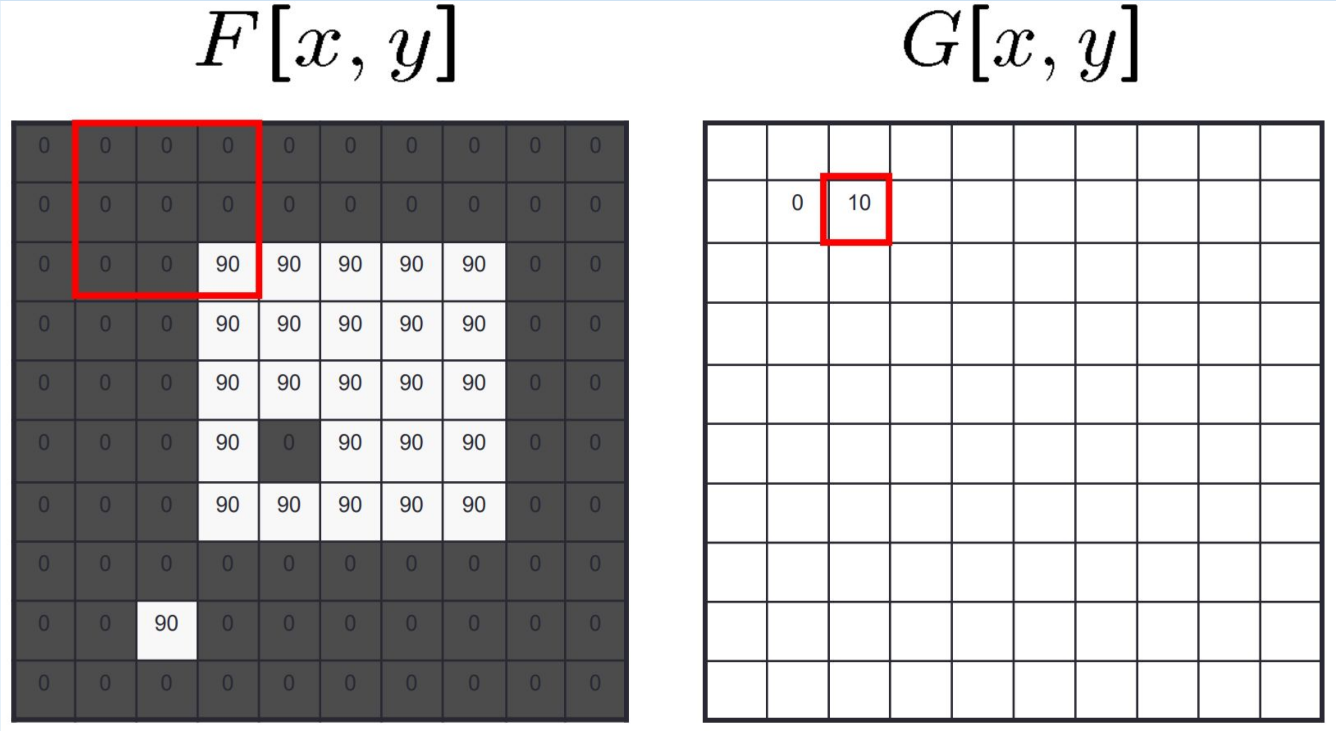
Convolution of an image with a kernel



What is the kernel used in this example?



What is the kernel used in this example?



What is the kernel used in this example?

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G[x, y]$

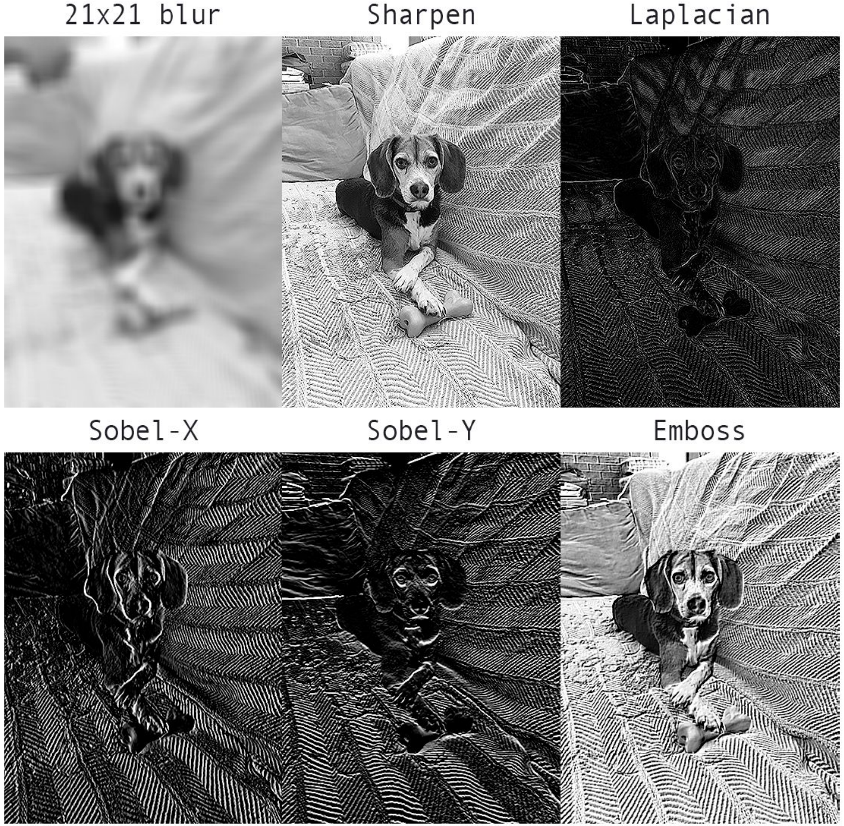
	0	10	20						

Convolution of an image with a kernel

$$S(i,j) = (I \star K)(i,j) = \sum_m \sum_n K(i-m, j-n) I(m,n)$$

Convolution of an image with a kernel

Convolution of an image with a kernel



Smile Detection

https://bit.ly/2ZgsSvr

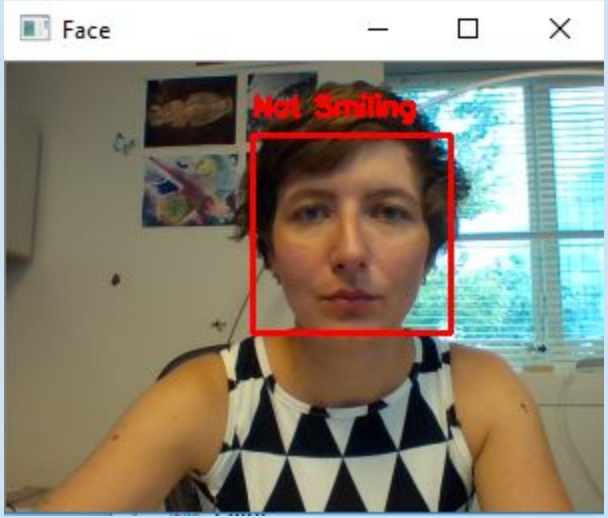
Discussion

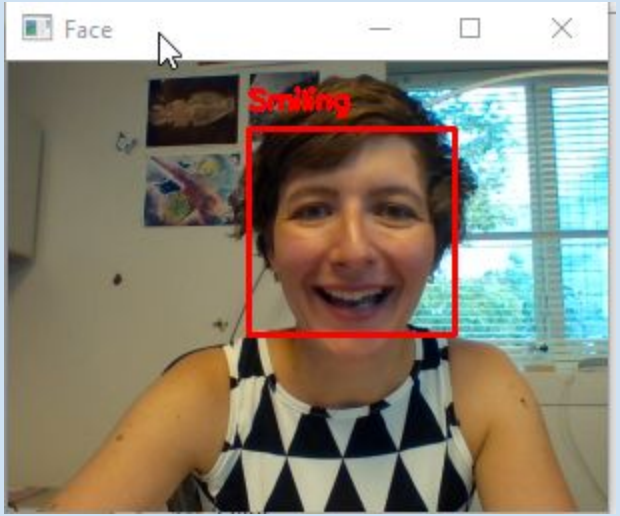
```
[INFO] evaluating network...
              precision    recall  f1-score   support

 not_smiling       0.93      0.95      0.94      1895
     smiling       0.86      0.81      0.83       738

    accuracy                           0.91      2633
   macro avg       0.89      0.88      0.89      2633
weighted avg       0.91      0.91      0.91      2633
```


Training Loss and Accuracy
Loss/Accuracy
0.2
0.4
0.6
0.8
0
2
4
6
8
10
12
14
Epoch #
train_loss
val_loss
acc
val_acc


Face
Not Smiling


Face
Smiling

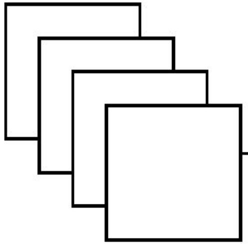
Some theory

Layer Types in CNN

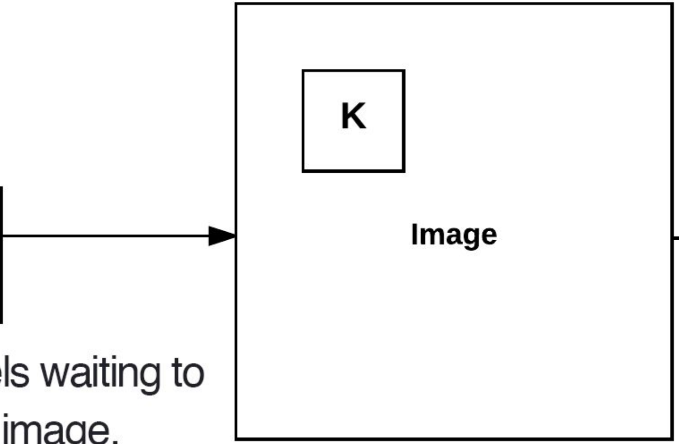
1. Convolutional (CONV)
2. Activation (ACT or RELU)
3. Pooling (POOL)
4. Fully-connected (FC)
5. Batch normalization (BN)
6. Dropout (DO)

Example of CNN: `INPUT -> CONV -> RELU -> FC -> SOFTMAX`

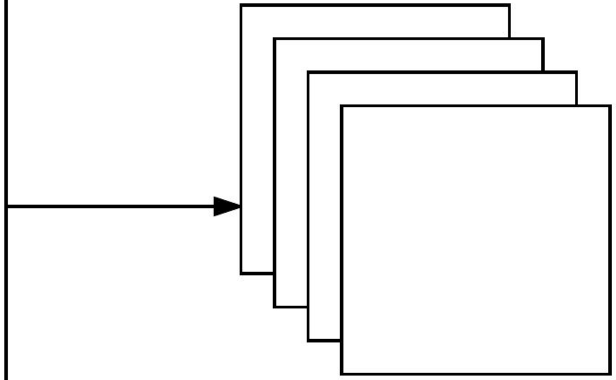
CONV Layers



Step #1: *K* kernels waiting to be applied to the image.

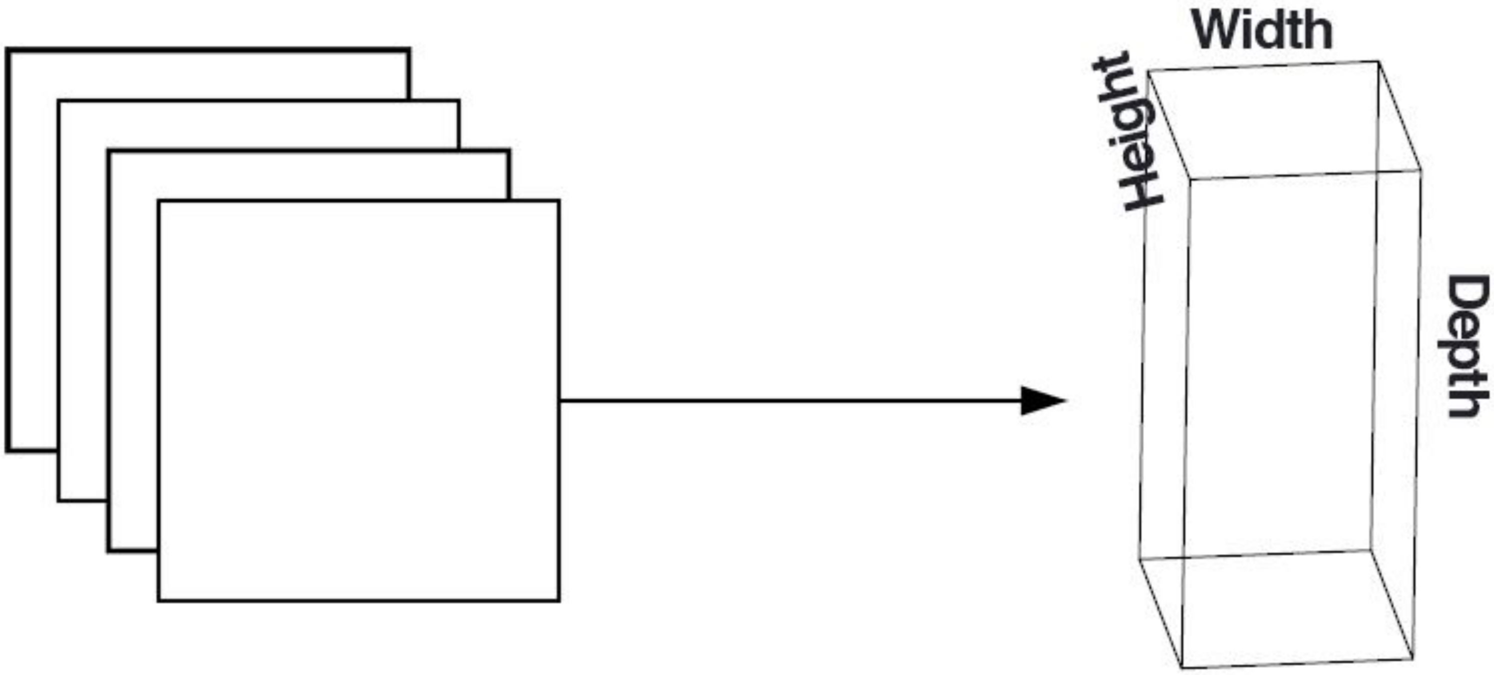


Step #2: Each kernel is convolved with the input volume.



Step #3: The output of each convolution operation produces a 2D output, called an “activation map”.

CONV Layers - the concept of <u>depth</u>



CONV Layers - the concept of strides

95	242	186	152	39
39	14	220	153	180
5	247	212	54	46
46	77	133	110	74
156	35	74	93	116

Image

0	1	0
1	-4	1
0	1	0

Kernel

692	-315	-6
-680	-194	305
153	-59	-86

Result with S=1

692	-6
153	-86

Result with S=2

CONV Layers - the concept of <u>padding</u>

692	-315	-6
-680	-194	305
153	-59	-86

0	0	0	0	0	0	0
0	95	242	186	152	39	0
0	39	14	220	153	180	0
0	5	247	212	54	46	0
0	46	77	133	110	74	0
0	156	35	74	93	116	0
0	0	0	0	0	0	0

-99	-673	-130	-230	176
-42	692	-315	-6	-482
312	-680	-194	305	124
54	153	-59	-86	-24
-543	167	-35	-72	-297

ACT Layers

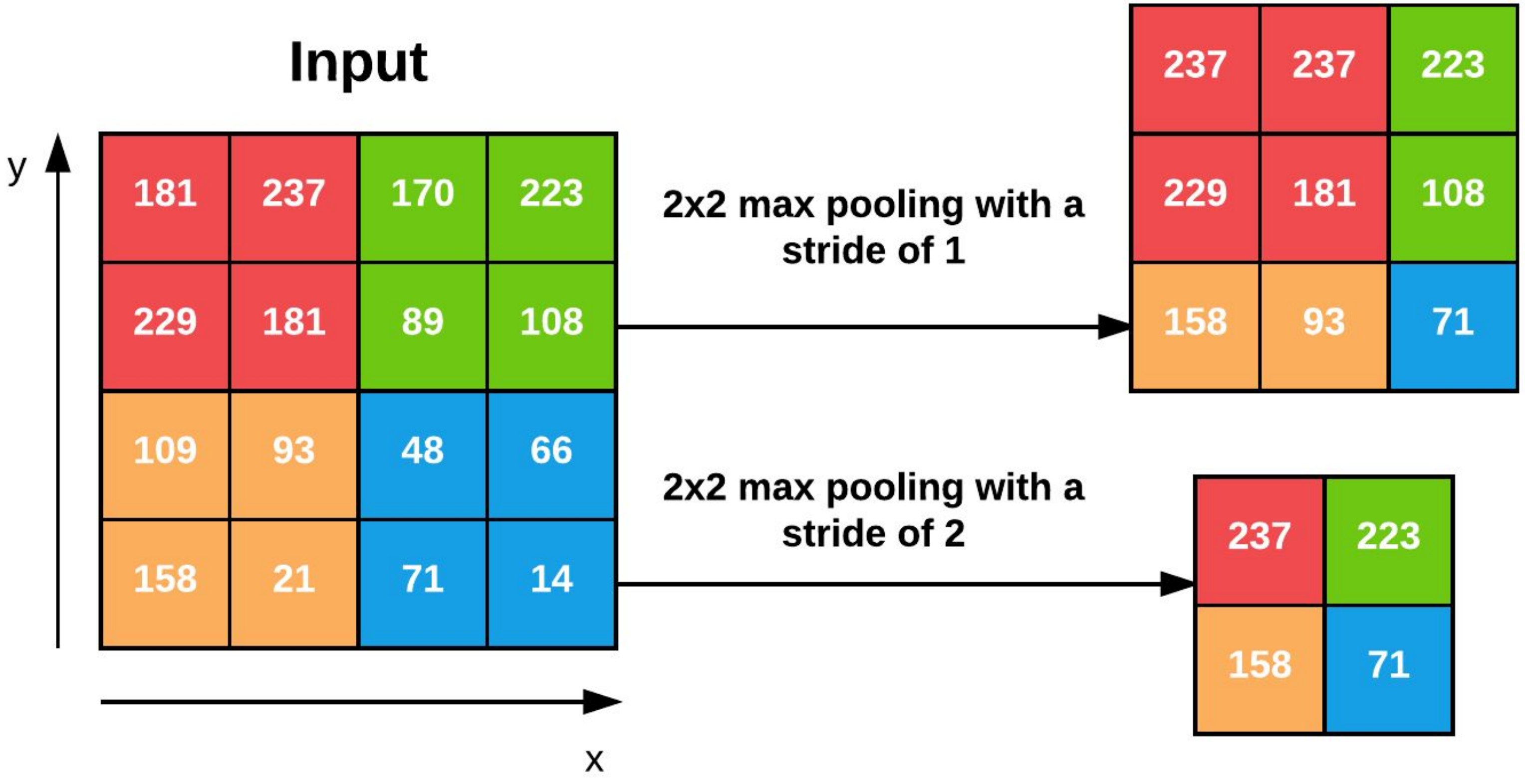
Input

-249	-91	-37
250	-134	101
27	61	-153

ReLU

0	0	0
250	0	101
27	61	0

POOL Layers


Input
y
181
237
170
223
229
181
89
108
109
93
48
66
158
21
71
14
x
2x2 max pooling with a stride of 1
237
237
223
229
181
108
158
93
71
2x2 max pooling with a stride of 2
237
223
158
71

FC Layers

```
INPUT -> CONV -> RELU -> POOL -> CONV -> RELU -> POOL -> FC -> FC -> SOFTMAX
```

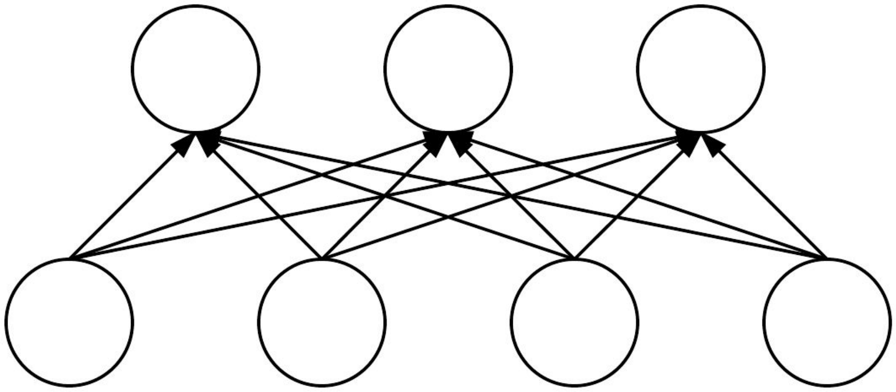
BN Layers

```
INPUT -> CONV -> RELU -> BN -> ...
```

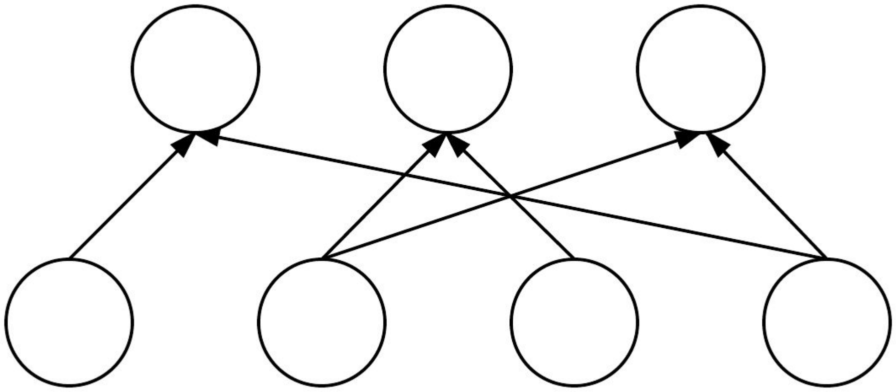
DO Layers

- Form of regularization
- Helps prevent overfitting by increasing testing accuracy

No Dropout



Dropout (50%)



Exploring the TPU capabilities