What happens in a deep learning image classifier?

Demystifying the black box

Dirk, 2021-05-11

Agenda

1 Input Layer

2 Convolutional Layer

3 Pooling Layer

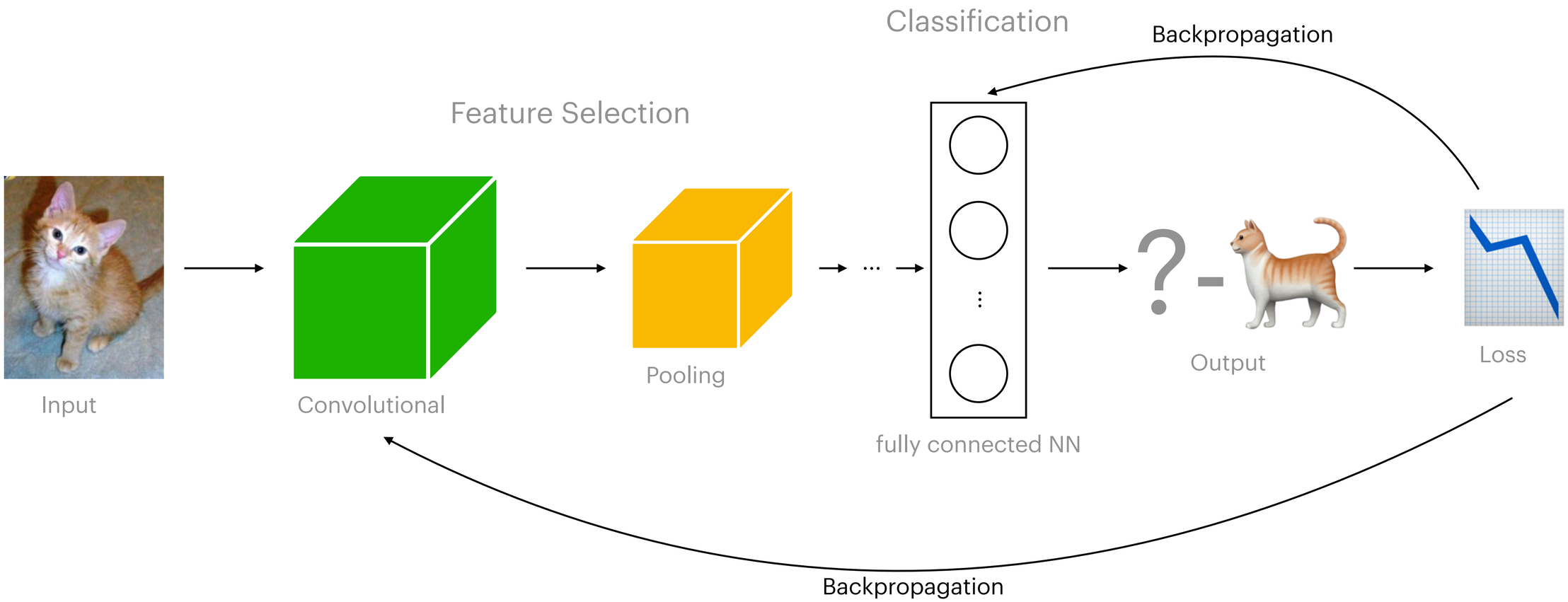
4 Fully Connected Layer

What happens in a deep learning image classifier?

Demystifying the black box

Dirk, 2021-05-11

Rough Overview of ConvNN



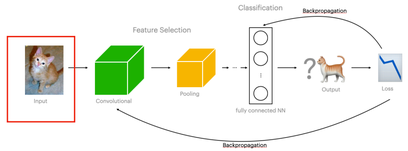
for every image: "walk" thru the NN and calculate loss by comparing our output with label

⇩

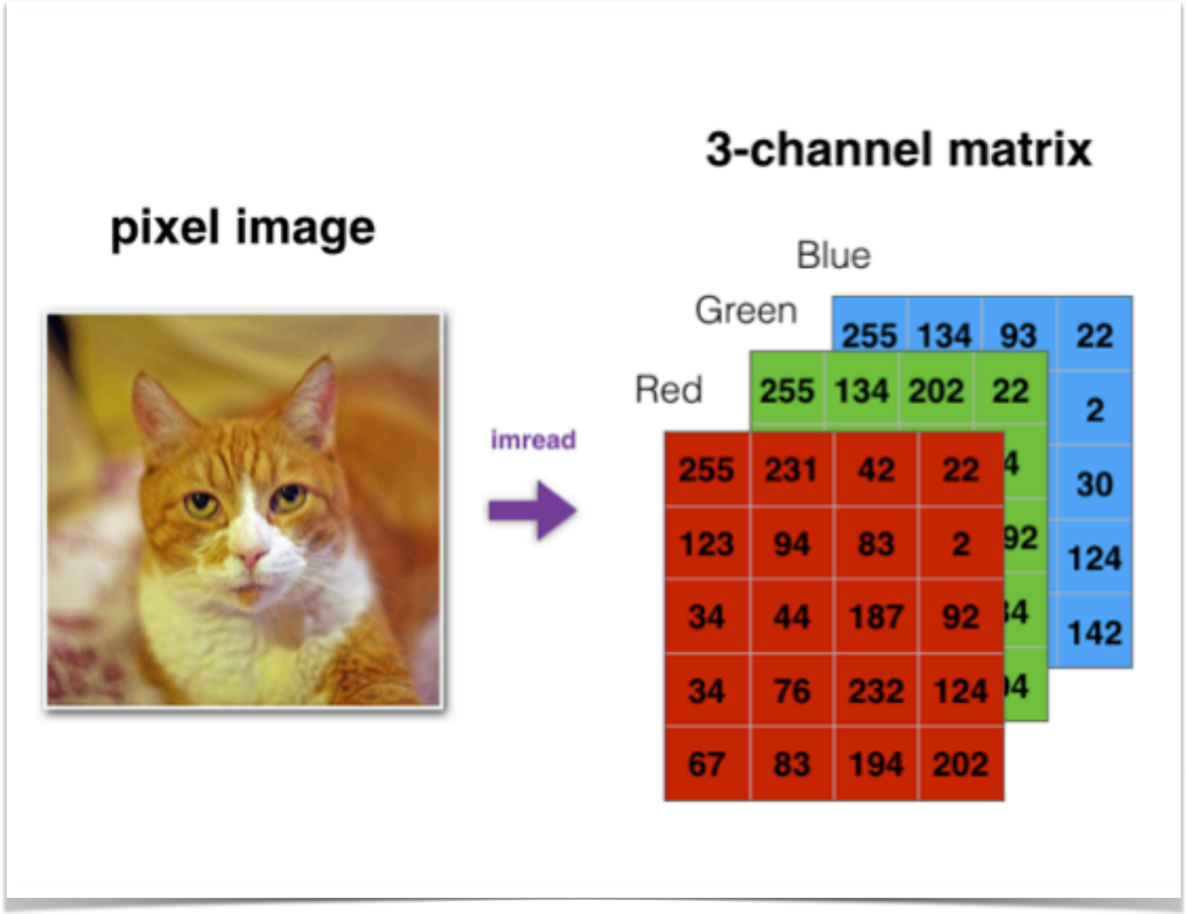
after last image: calculate 🤓average of all losses and update parameters (w, b, f) thru backprop

⇩

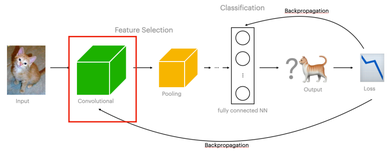
do this all over again for every epoch



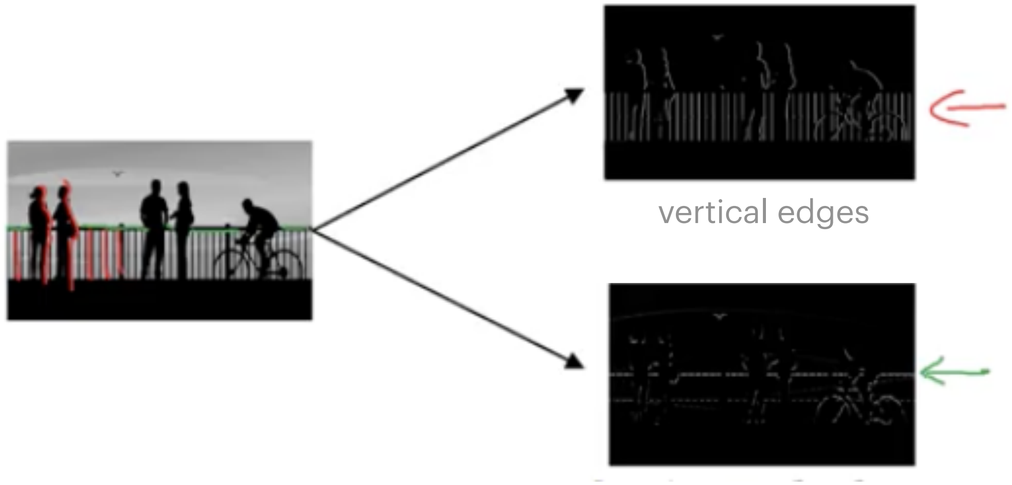
Input Layer



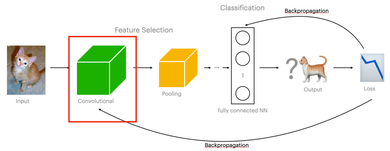
- 224px color -> 224x224x3 Input Dimensions
- 100px Grayscale -> 100x100 Input Dimensions
- Function for cropping images in PyTorch 🕺🎊
- does not have to be square size, but easier & kind of standard 🆒



Conv Layer: why it works



- combination of basic edges gives more complex shapes
- an edge is just a line with different pixel values
- with 3 channels: any color of edges are findable



Filter: Vertical edge detection

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

Input

X

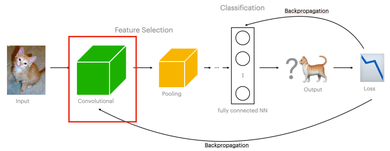
1	0	-1
1	0	-1
1	0	-1

Filter

=

0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0

Output



Filter: Vertical edge detection

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

Input

X

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

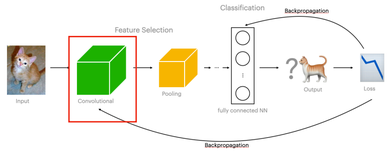
Filter

😎 learnt by backpropgation

=

0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0

Output



Conv Layer: Filter example

```
(3x1 + 1x1 + 2x1) + (0x0 + 5x0 + 7x0) + (1x-1 + 8x-1 + 2x-1) = -5
(7x1 + 1x1 + 2x1) + (2x0 + 3x0 + 1x0) + (5x-1 + 1x-1 + 6x-1) = -2
(8x1 + 2x1 + 3x1) + (9x0 + 5x0 + 1x0) + (2x-1 + 1x-1 + 7x-1) = 2
```

3	0	1	2	7	4
1	5	8	9	2	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

Input (e.g.: image matrix)
6x6x3

X

1	0	-1
1	0	-1
1	0	-1

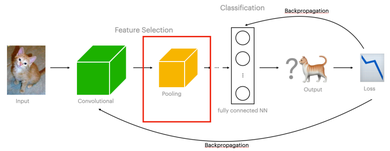
Filter

=

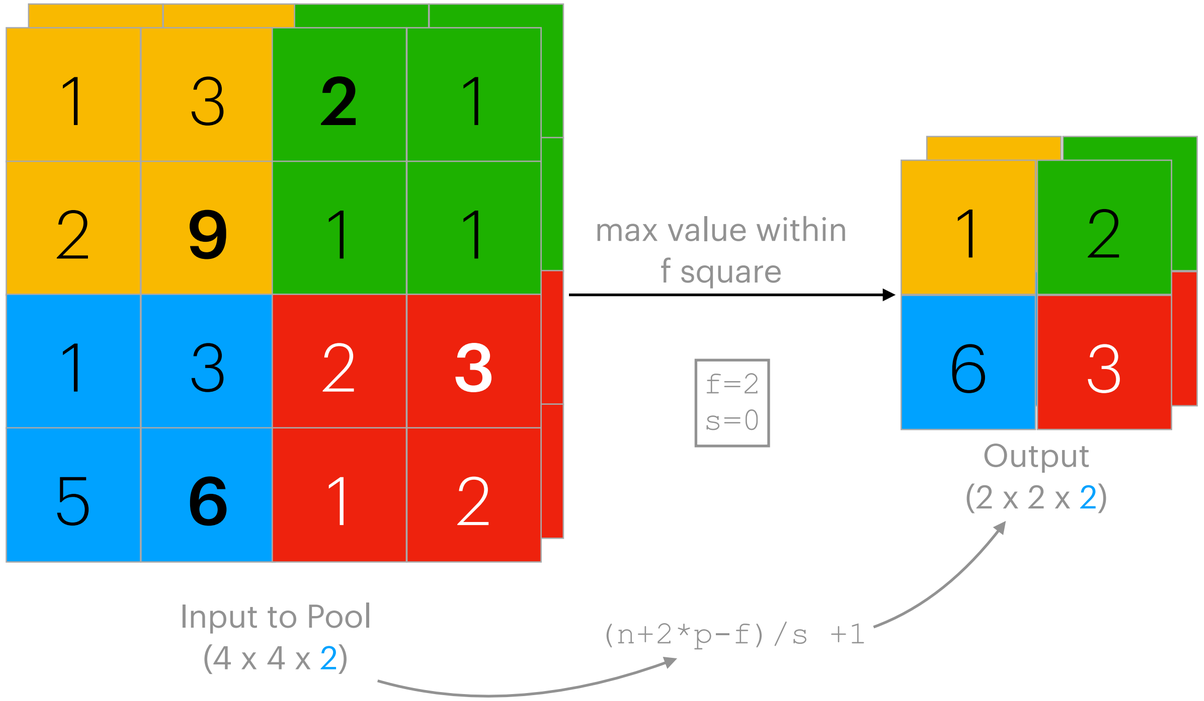
-5	-4	0	8
-10	-2	2	3
0	-2	-4	-7
-3	-2	-3	-16

Output (e.g.: image matrix)
4x4x3

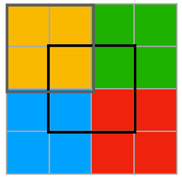
Filter

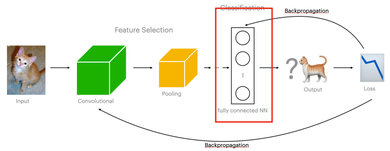


Pooling Layer

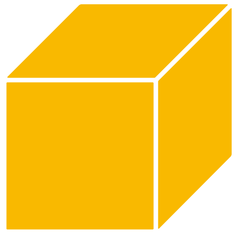


- 2 Hyperparameters (to chose): f, s 🤔
- no parameters to learn
- reduces size of NN 🥳
- here: `max-pooling`
- example for s=1 (overlapping boxes 🤓):

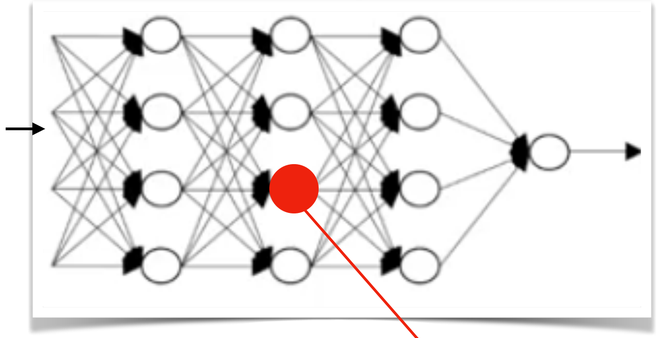




Fully connected layer

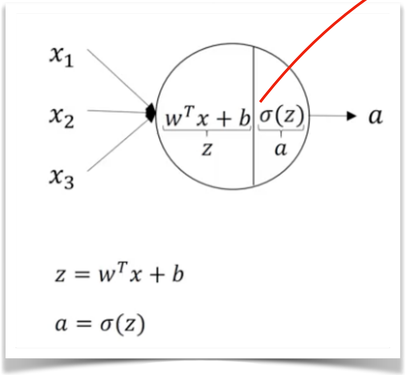


Pooling

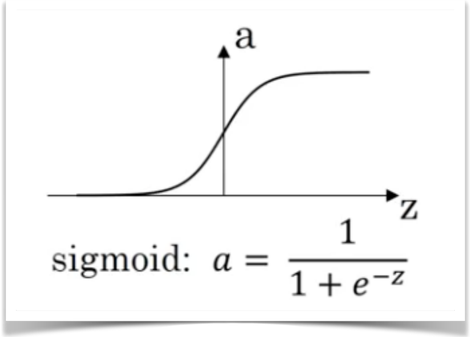


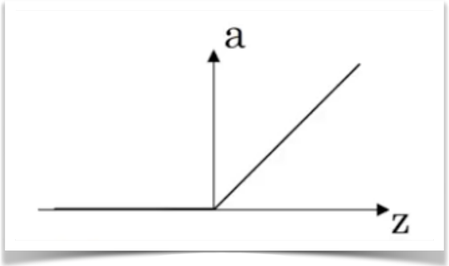
Output

Neuron

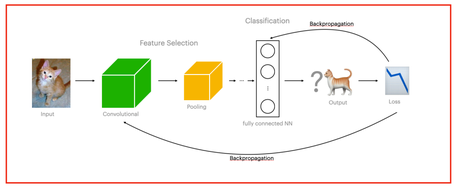


Activation Functions

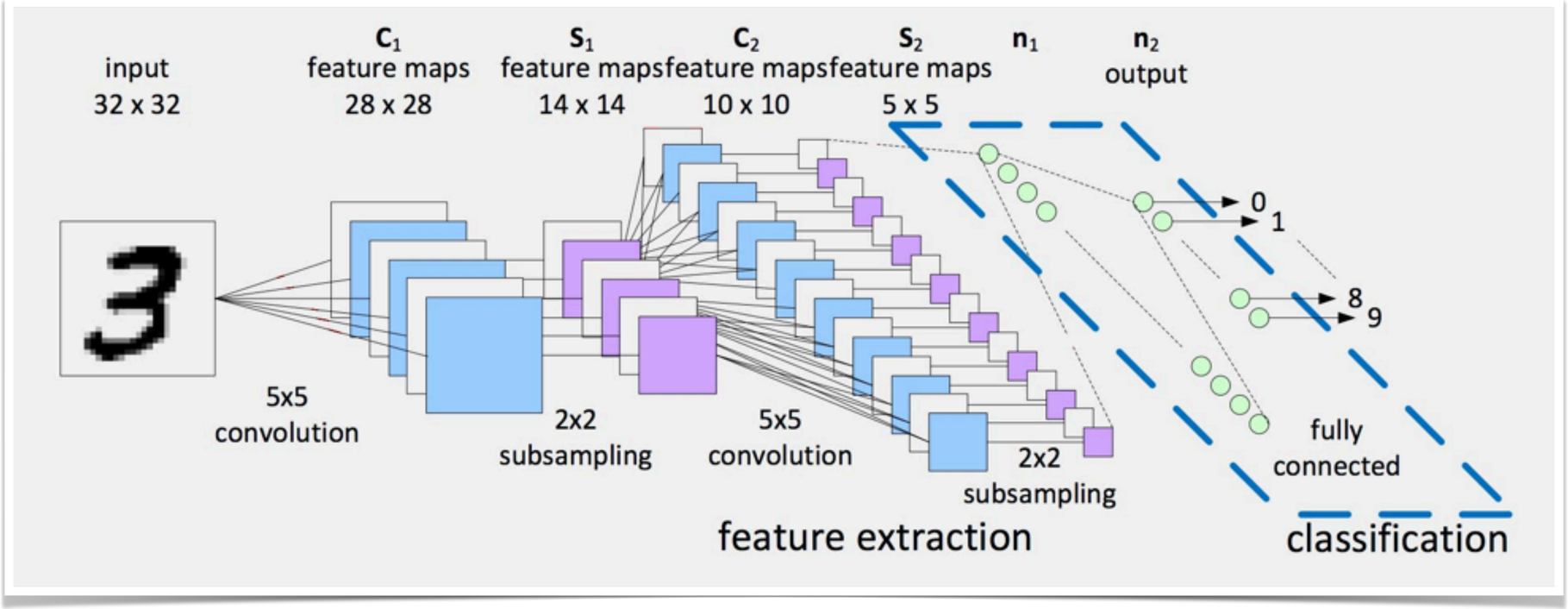




ReLU



Example of ConvNN



greyscale images with handwritten digits
10 labels (output classes)

Example of ConvNN

Takeaway

🤓 ConvNN have 4 types of layer: input, filter, pooling, NN, output

🕵️ ConvLayers finds basic shapes - 👷 fullNN Layers classifies them

🧮 Values in filters are learned by backprop (but can be set manually)

👏 NN size smaller than fully connected NN

🤷🏻‍♀️ more hyperparameters (pats, stride, ...) to tune & architecture (pretrained model class, number of hidden layers, neurons per layer, ...) *more knowledge ➡ easier find best model*

References

1. arXiv 2013 | Visualizing and Understanding Convolutional Networks *by M. Zeiler & R. Fergus*
2. Coursera Course 2019 | Convolutional Neural Networks *by Andrew Ng*