Convolutional Neural Network

LEOW CONG SHENG

RIYAN ANDRIKA

MLDA @EEE

MACHINE LEARNING AND DATA ANALYTICS

Who are we?

Leow Cong Sheng

EEE Year 3

LinkedIn

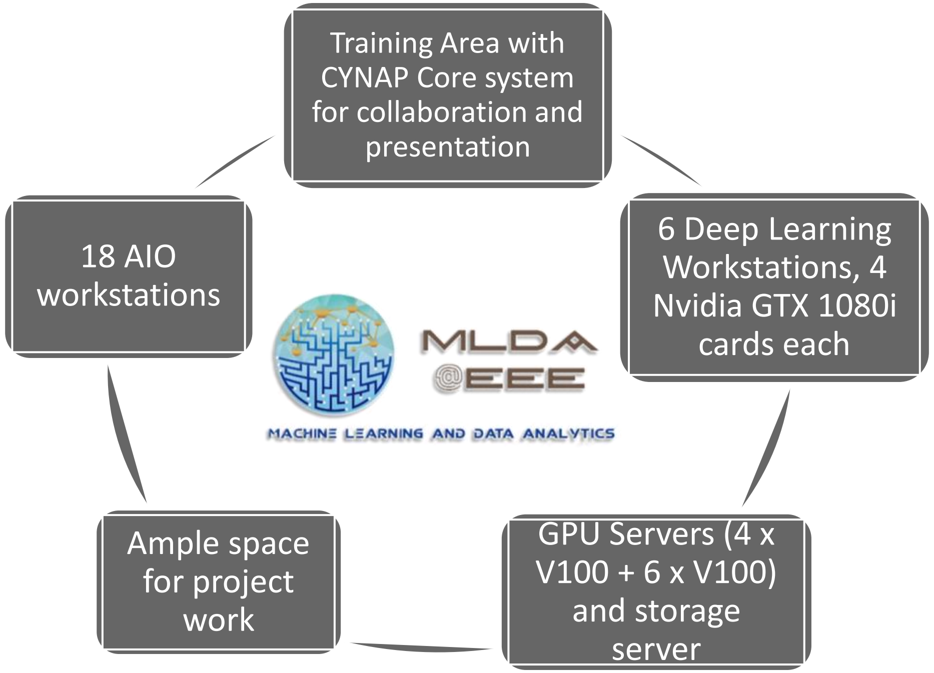
Riyan Andrika

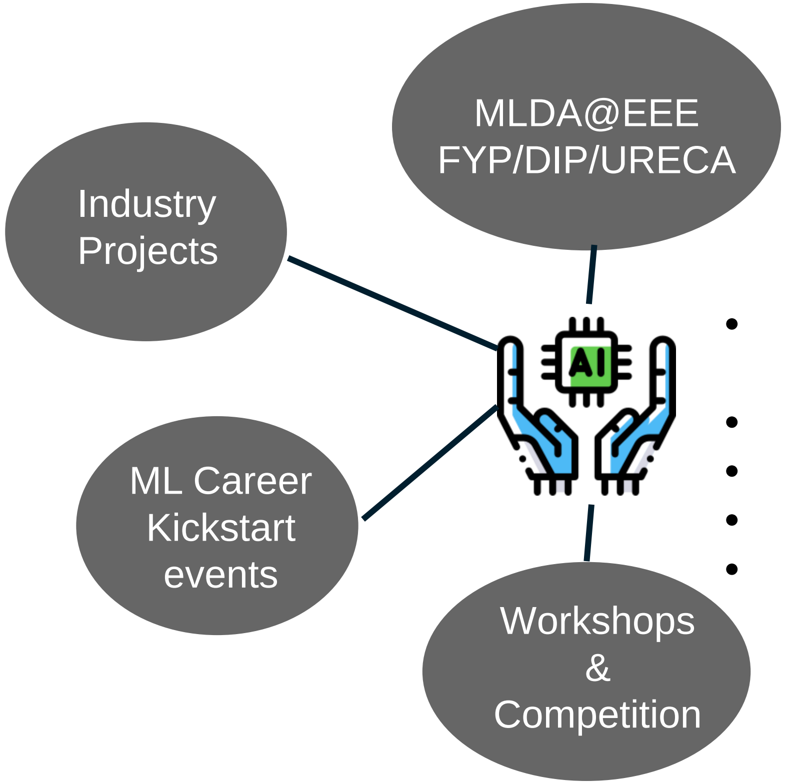
EEE Year 2

LinkedIn

Our Mission

Provide an integrated platform for EEE/IEM students to learn and implement Machine Learning, Data Science & AI, as well as facilitate connections with the industry.





- >1000 Trained ML practitioners
- >10 Academic Projects
- >30 Industry projects
- >5 competition
- >15 Industry Partners

Aims and Scope of Workshop

Participants should ideally have:

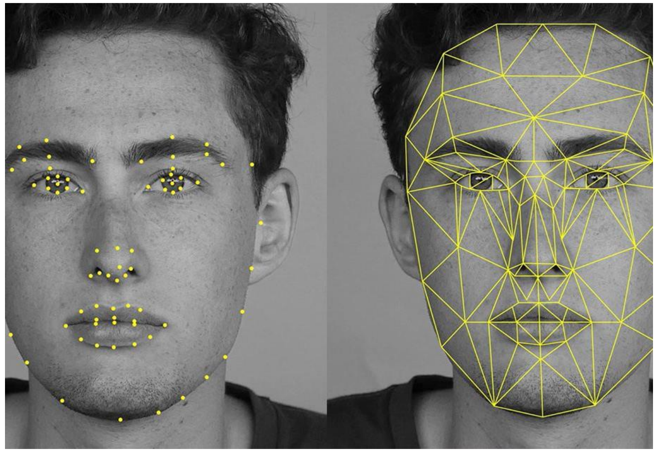
- Basic Python programming knowledge.
- Basic knowledge on machine learning.
- Basic idea of Neural Networks.

Identify	Identify what is CNN and its applications for it.
Differentiate	Differentiate between Neural Networks and CNN.
Understand	Understand the basic ideas behind CNN architecture and parameters (convolution, padding, depth, stride, and the different layers).
Implement	Implement a simple CNN model for simple image classification.
Answer	Answer some simple CNN interview questions relevant for internship or job interviews.

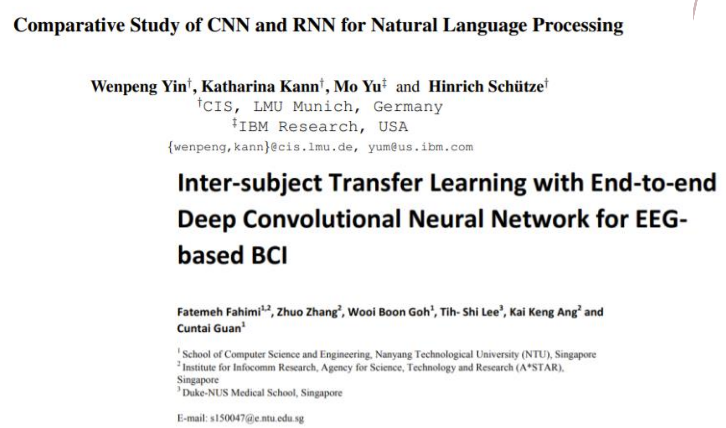
What is CNN and why use it?

- Neural Network with additional "magic" layers.
- Biologically-inspired, introduced as "Neocognitron" by Kunihiko Fukushima with convolutional and down-sampling layers.
- Performs very well for image-related tasks.
- Applications in image-recognition, video analysis, natural language processing, anomaly detection, time-series forecasting and more.

X-rays image diagnosis using CNN: https://www.smart2zero.com/news/algorithm-beats-radiologists-diagnosing-x-rays



Facial Recognition with CNN: https://hackernoon.com/building-a-facial-recognition-pipeline-with-deep-learning-in-tensorflow-66e7645015b8



Comparative Study of CNN and RNN for Natural Language Processing

Wenpeng Yin†, Katharina Kann†, Mo Yu‡ and Hinrich Schütze†
†CIS, LMU Munich, Germany
‡IBM Research, USA
{wenpeng,kann}@cis.lmu.de, yum@us.ibm.com

Inter-subject Transfer Learning with End-to-end Deep Convolutional Neural Network for EEG-based BCI

Fatemeh Fahimi[1,2], Zhuo Zhang[2], Wooi Boon Goh[1], Tih-Shi Lee[3], Kai Keng Ang[2] and Cuntai Guan[1]

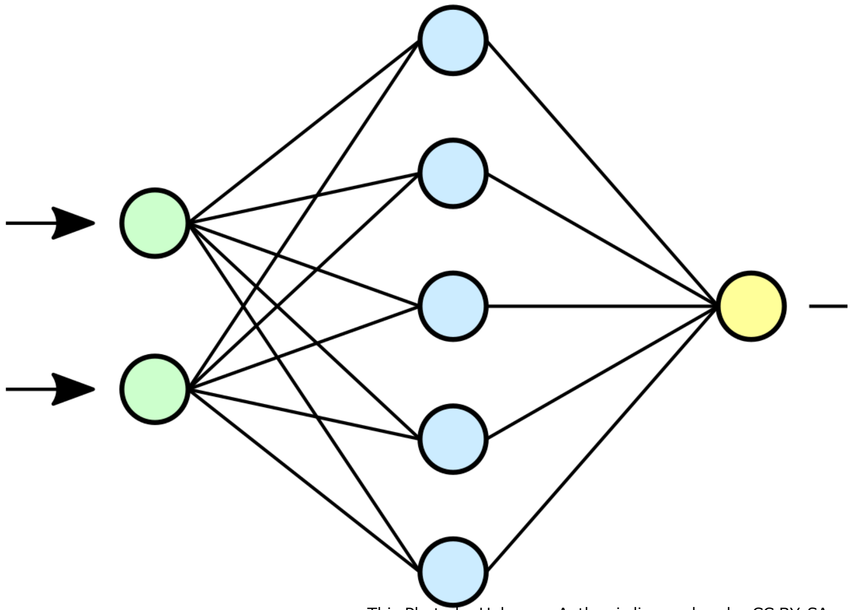
[1] School of Computer Science and Engineering, Nanyang Technological University (NTU), Singapore
[2] Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), Singapore
[3] Duke-NUS Medical School, Singapore

E-mail: s150047@e.ntu.edu.sg

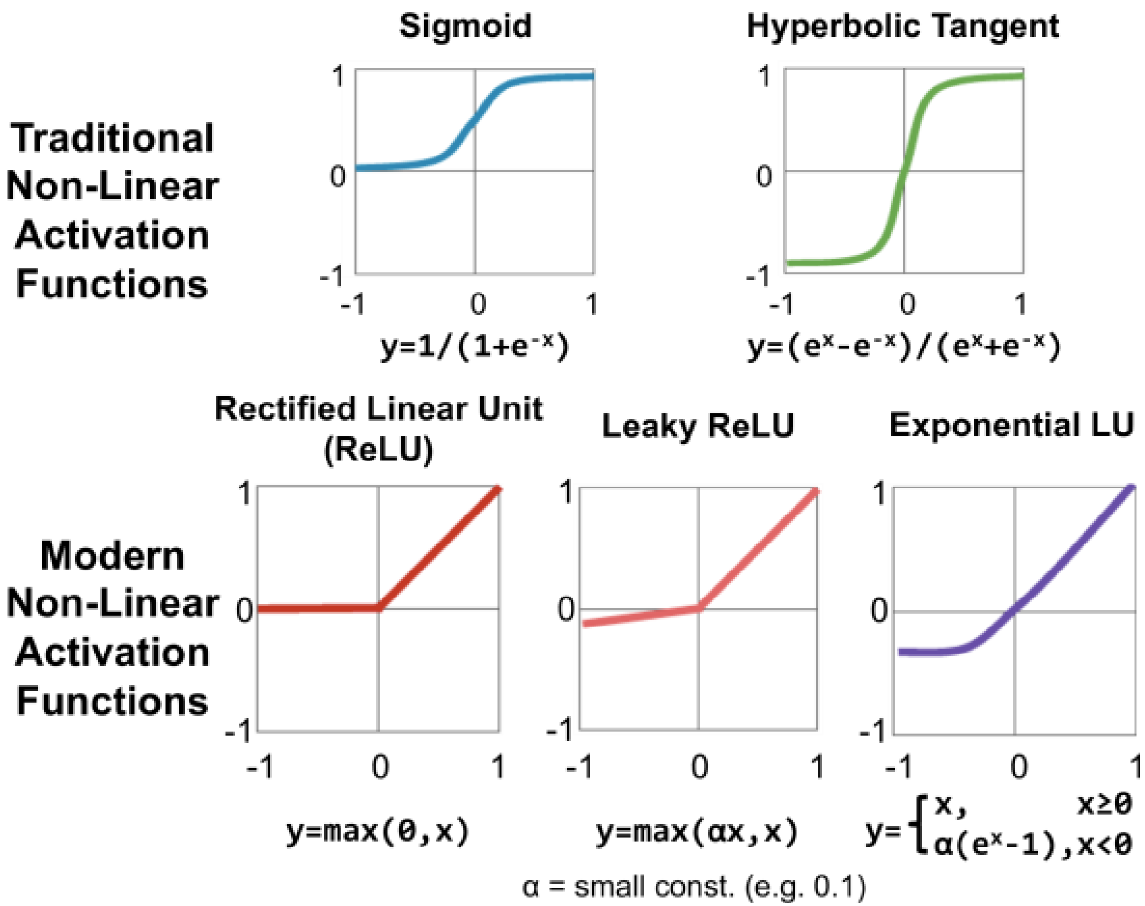
CNN successfully implemented for non-image tasks.

How does CNN Works?

- To understand how CNN works, a simple approach is to first look at an artificial neural network (ANN) before looking at the “convolutional” aspect.
- A neural network takes certain number of input/features and pass it through multiple layers of calculations.
- Each neuron, also depicted as a node in the figure, computes a linear function before pushing it to a function called activation function which operates on its input to give an output.
- Forward propagation for calculation and backward propagation for loss reduction

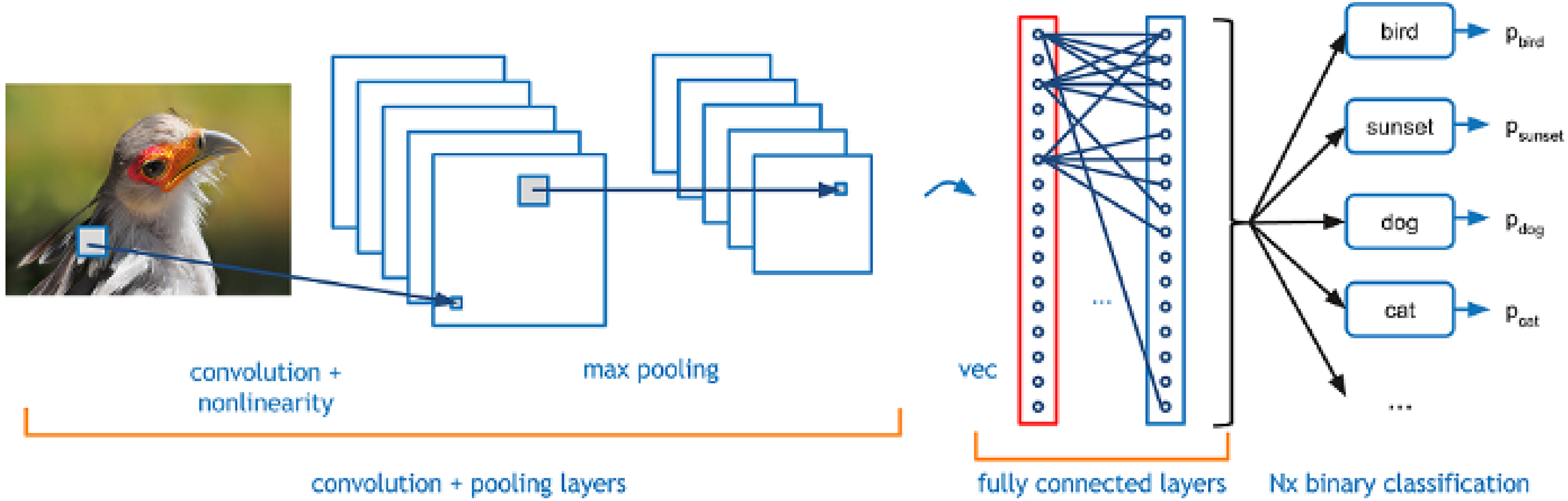


Activation Functions



ANN (SIMPLE NEURAL NETWORK)	CNN
General-purpose	Specially good on data with spatial relationship
Great with non-linear and implicit relationships	Great with high-dimensionality
Manual feature input	Implicit feature detection
Relatively lightweight	Relatively heavier

When to use CNN or ANN?

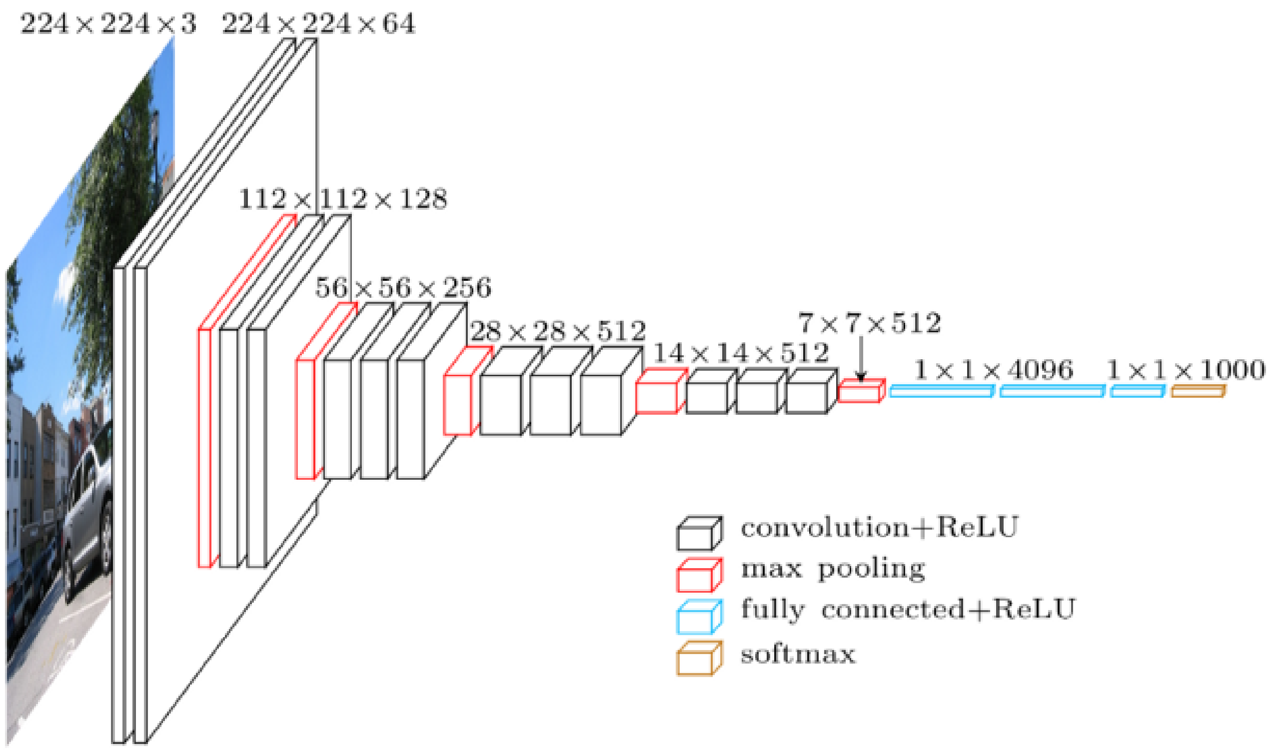


CNN Architecture

Layers of CNN

Input, Hidden, Output

- Input
- Convolution
- Pooling
- (Dropout)
- Fully Connected
- Output

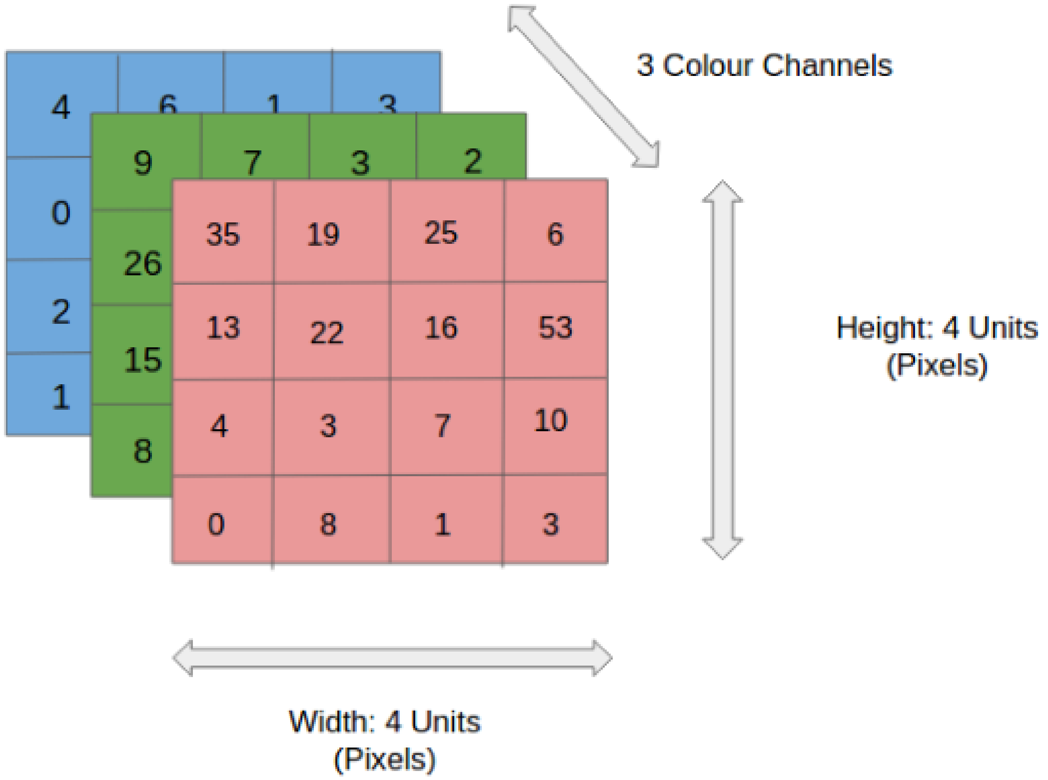


Input Layer

Matrix of pixels with a given width (W) and height (H)

(W X H X C)

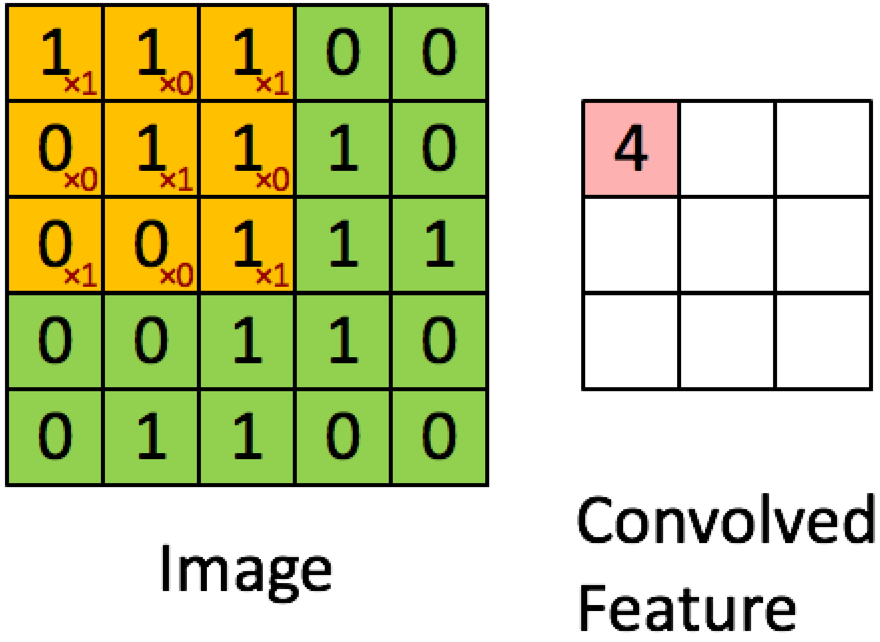
- A grayscale image: channel = 1
- An RGB image: channel = 3



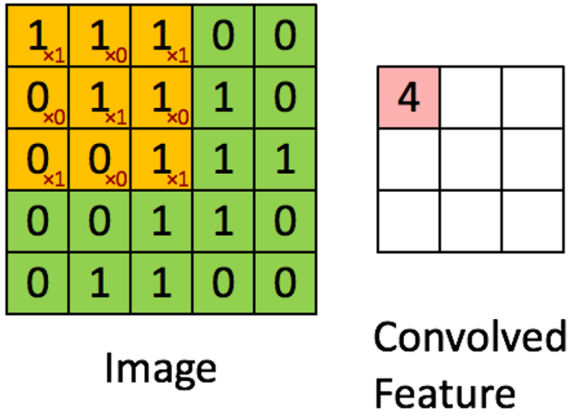
Convolution Layer - Convolution

$$Recall: f(t) * g(t) = \int_{-\infty}^{\infty} f(\tau) g(t - \tau) d\tau$$

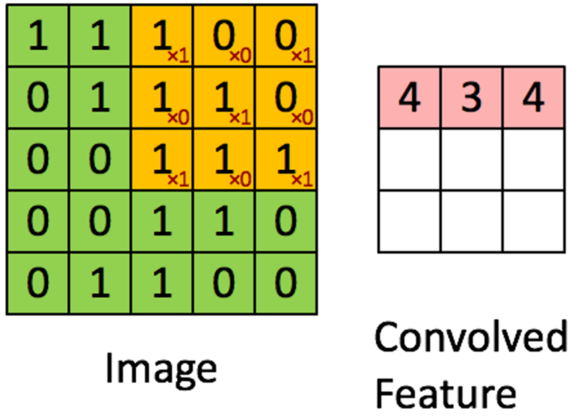
- Cross-correlation
- Flipping does not affect purpose
- Linear operation
- Kernel
- Generation for feature/activation map
- Multiple filters, multiple channels



Convolution Layer - Convolution

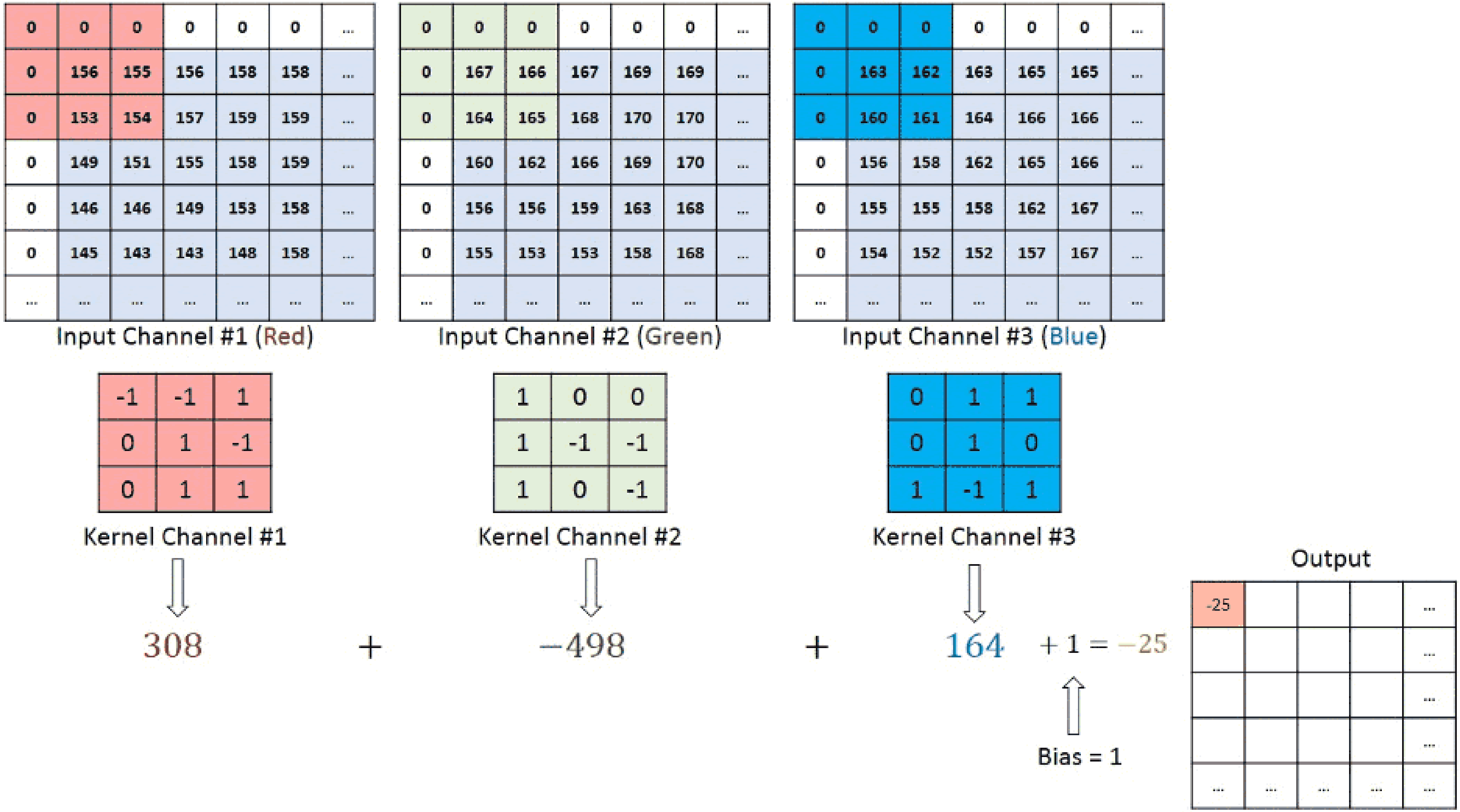


Feature 0



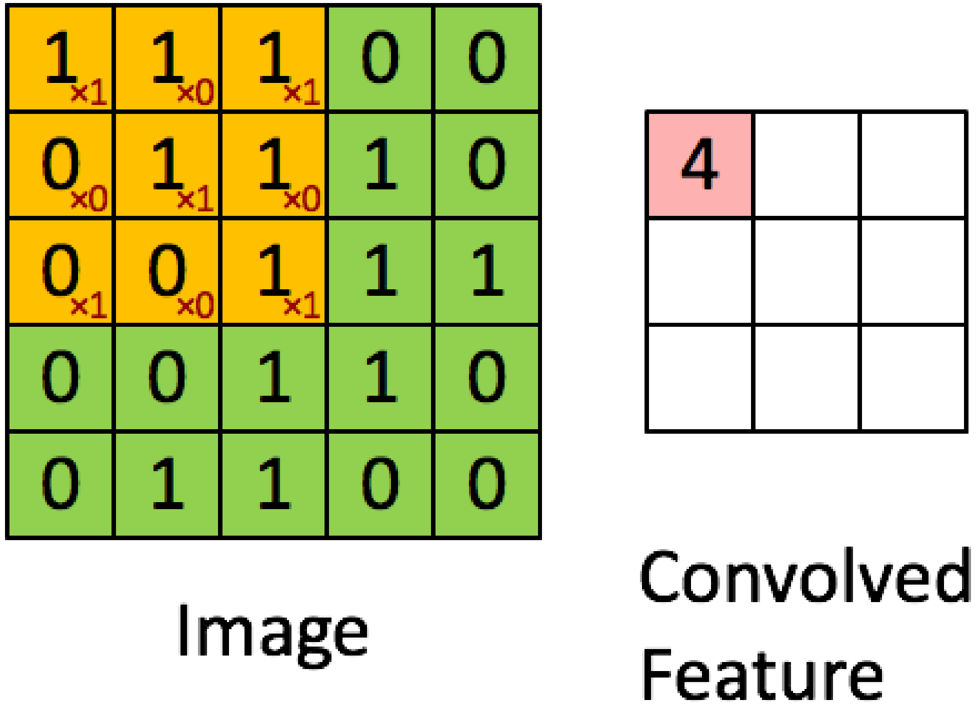
Feature 2

- Example of computation
- Feature 0:
  - (1x1) + (1X0) + (1X1) + (0X0) + (1X1) + (1X0) + (0X1) + (0X0) + (1X1) = 4
- Feature 2:
  - (1x1) + (0X0) + (0X1) + (1X0) + (1X1) + (0X0) + (1X1) + (1X0) + (1X1) = 4



Input Channel #1 (Red)
Input Channel #2 (Green)
Input Channel #3 (Blue)
Kernel Channel #1
Kernel Channel #2
Kernel Channel #3
Output
-25
308
+
−498
+
164
+ 1 = −25
Bias = 1

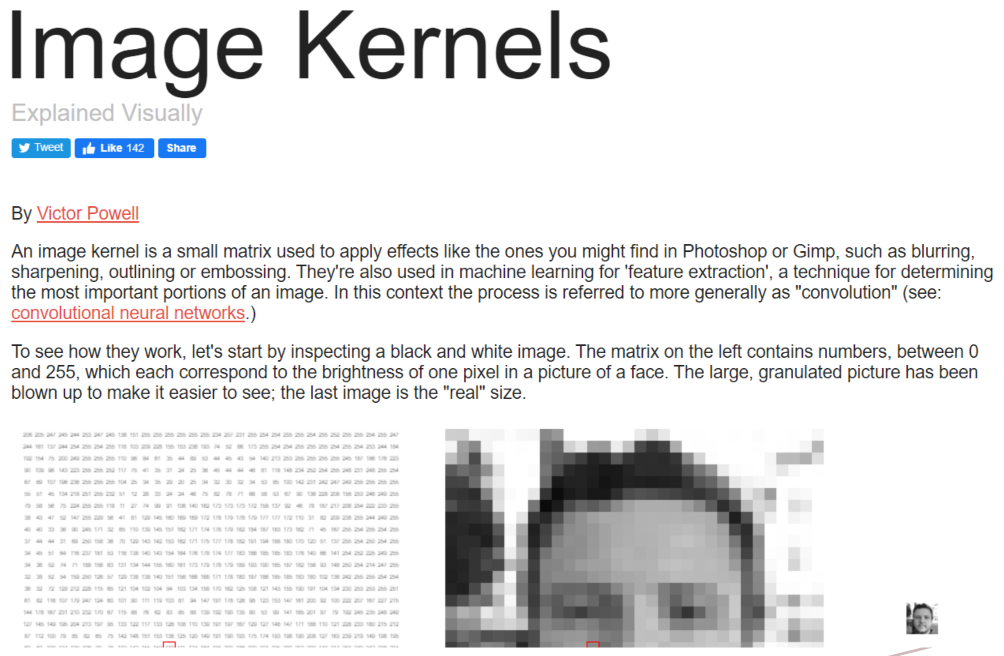
Convolution Layer - Kernel



- “Activate” the specified locations
- Each kernel have a corresponding activation map/feature map
- Small in spatial dimensionality
- Spread along the entirety of the depth of input

<u>Convolution Layer</u> - Kernels

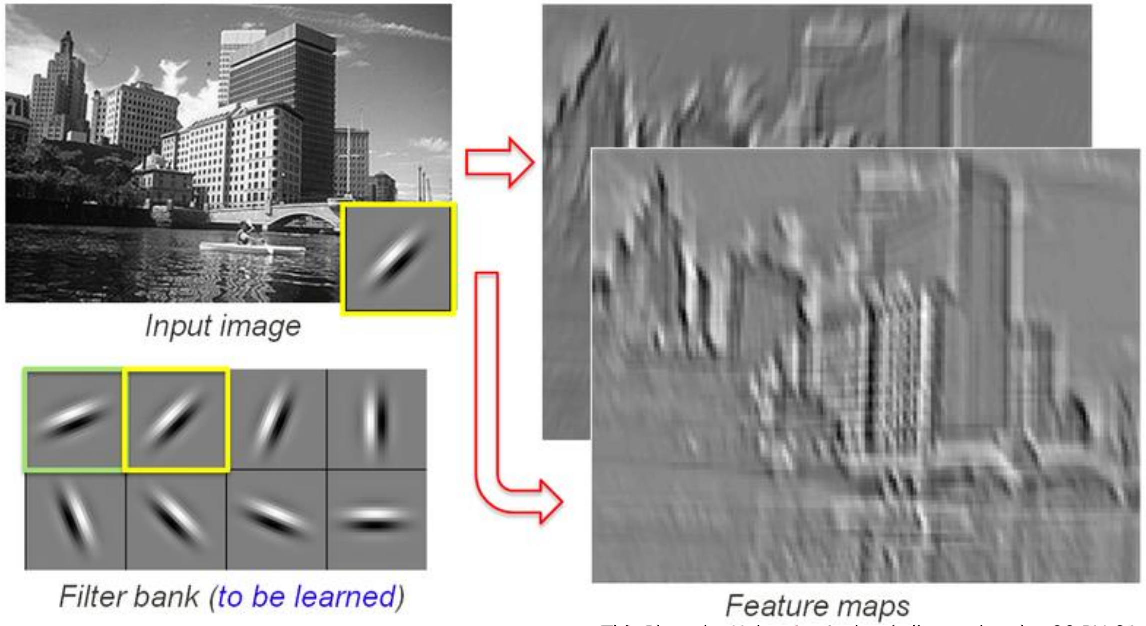
- Visit https://setosa.io/ev/image-kernels/
- Different types of kernels result in different feature obtained
- Depends on data and task e.g. sharpen, sobel and etc



How image kernels work.

Convolution Layer – Filter

- Filter is a concatenation of kernels based on channels
- One of the hyperparameters
- Equivalent to neurons for CNN
- Each output is different



$$Filter: f \ x \ f \ x \ c \ x \ c'$$

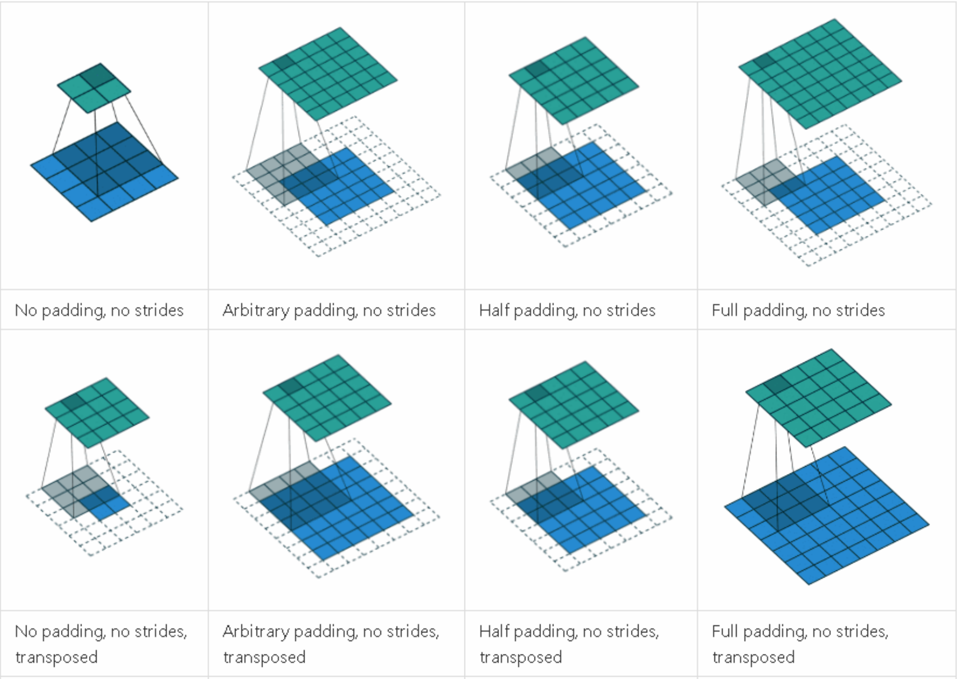
Where f is filter size, c is number of channel & c′ is number of output channel

Convolution Layer - Stride

- How much to move from one receptive field to another
- Setting stride = 1 means moving the kernel by 1 value before obtaining the next feature/activation map
- Low stride = large activations
- High stride = lower dimension

Convolution Layer - Padding

- Border information may be loss during convolution
- Zero-padding -> be more inclusive and manage data size
- Prevent network output size to shrink with depth

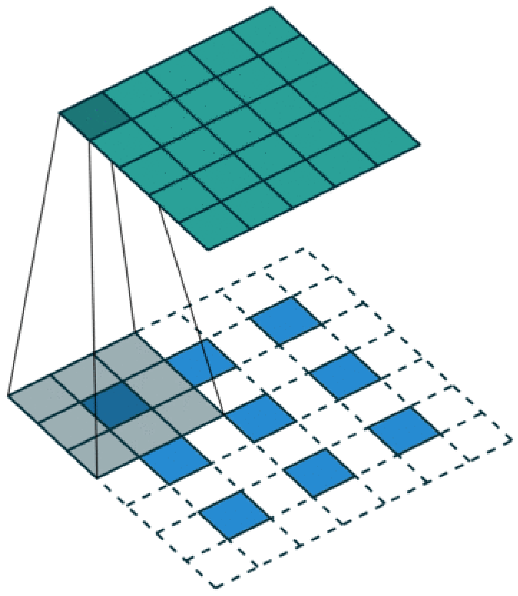


Convolution Layer – Output Size

Depends on the following

- Input Size (W,H)
- Number of Kernel (K)
- Kernel Size (F)
- Stride (S)
- Padding (P)

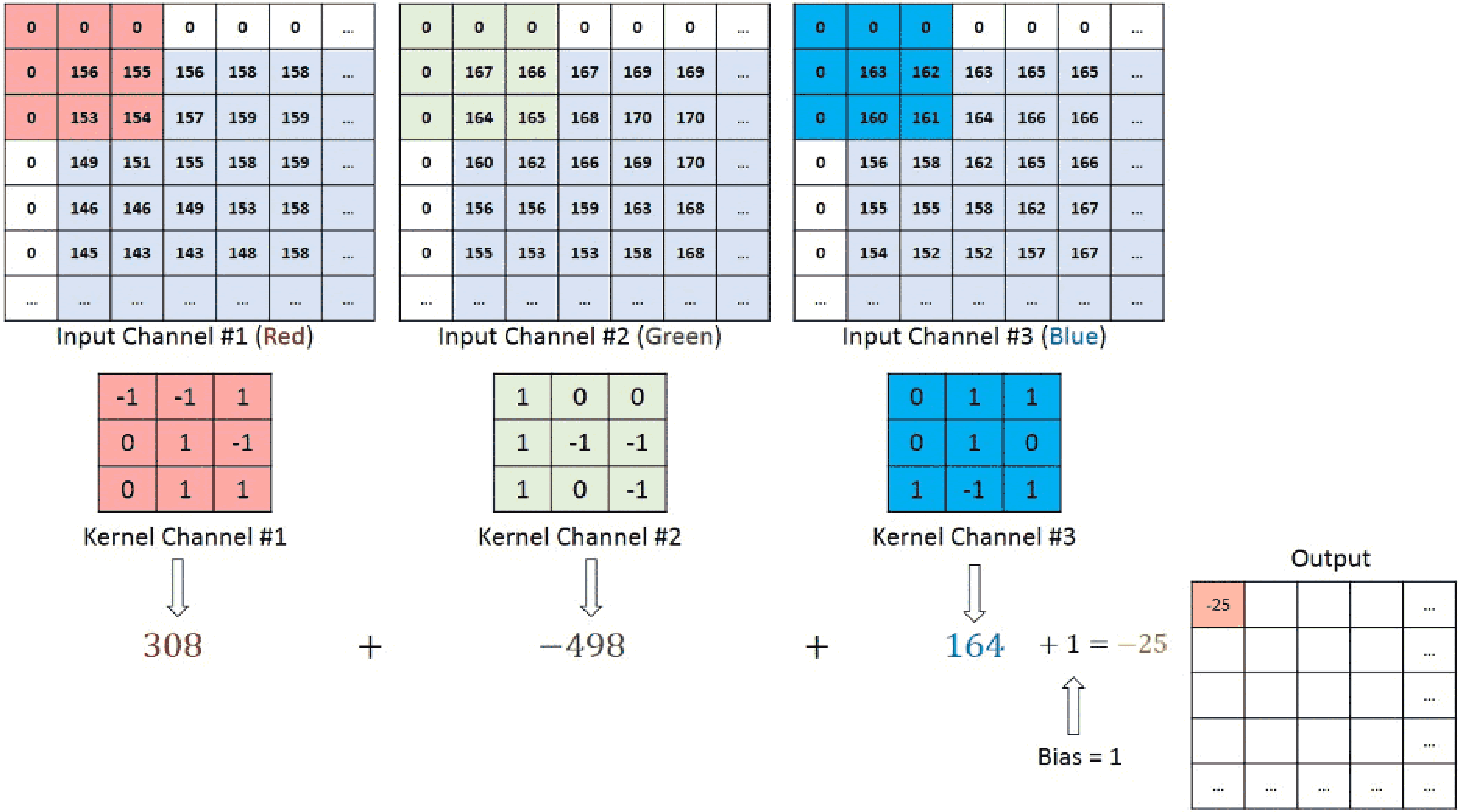
General formula for output size, where D is dimension:



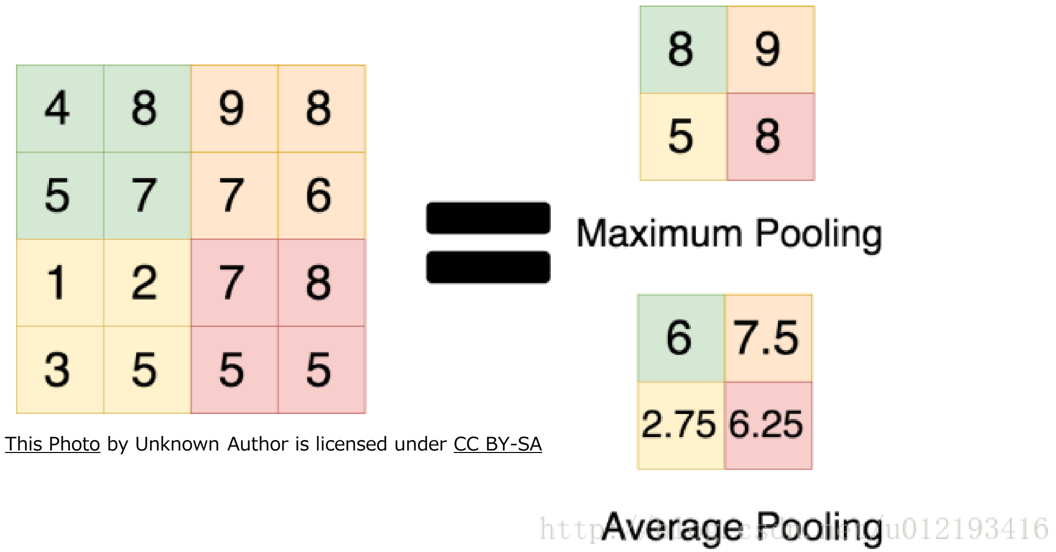
$$W_2 = \frac{(W_1 - F + 2P)}{S} + 1$$

$$H_2 = \frac{(H_1 - F + 2P)}{S} + 1$$

$$D_2 = K$$


Input Channel #1 (Red)
Input Channel #2 (Green)
Input Channel #3 (Blue)
Kernel Channel #1
Kernel Channel #2
Kernel Channel #3
308
+
−498
+
164
+ 1 = −25
Bias = 1
Output
-25

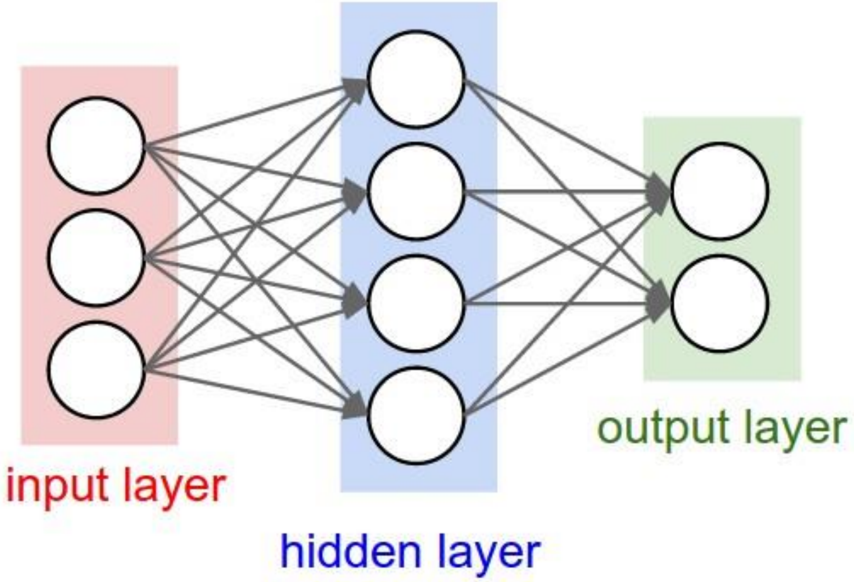
Pooling Layer



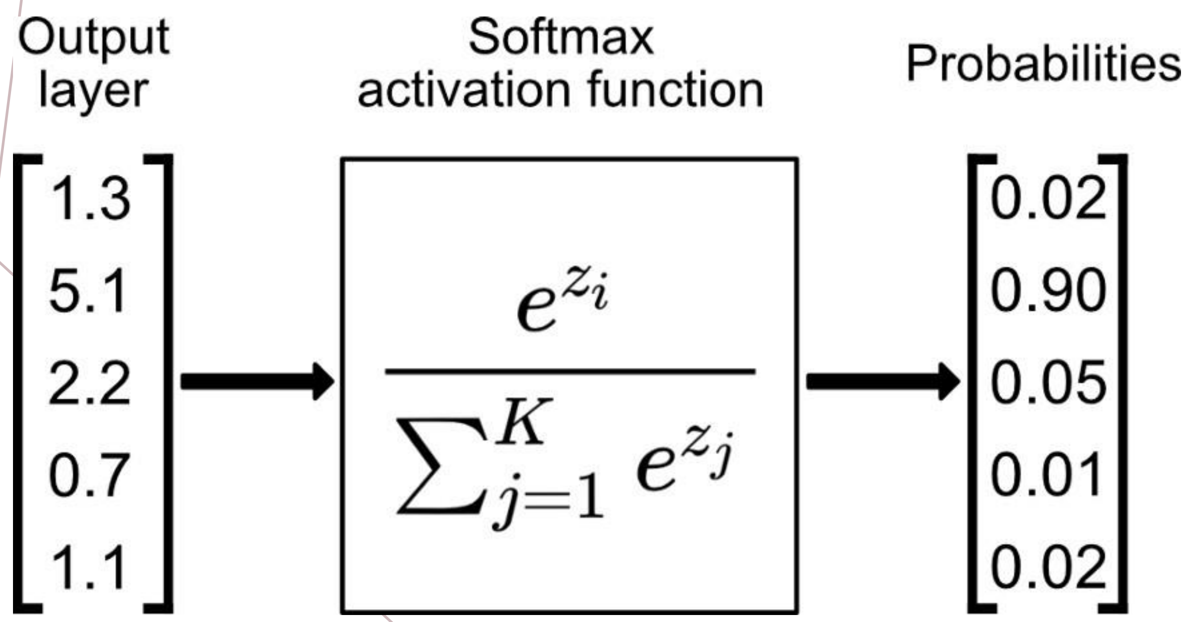
- Down-sampling for dimensionality reduction
- Destructive – 2x2 or overlapping pooling
- Max-pooling (Get brightest), Min-pooling(darkest) and Average Pooling(Smoothing)

Fully-connected Layers

- Typical ANN layers
- Each node connected to every node in both previous and next layer
- Input features into ANN layers
- May require complex computation with large parameter number
- Spatial information discarded



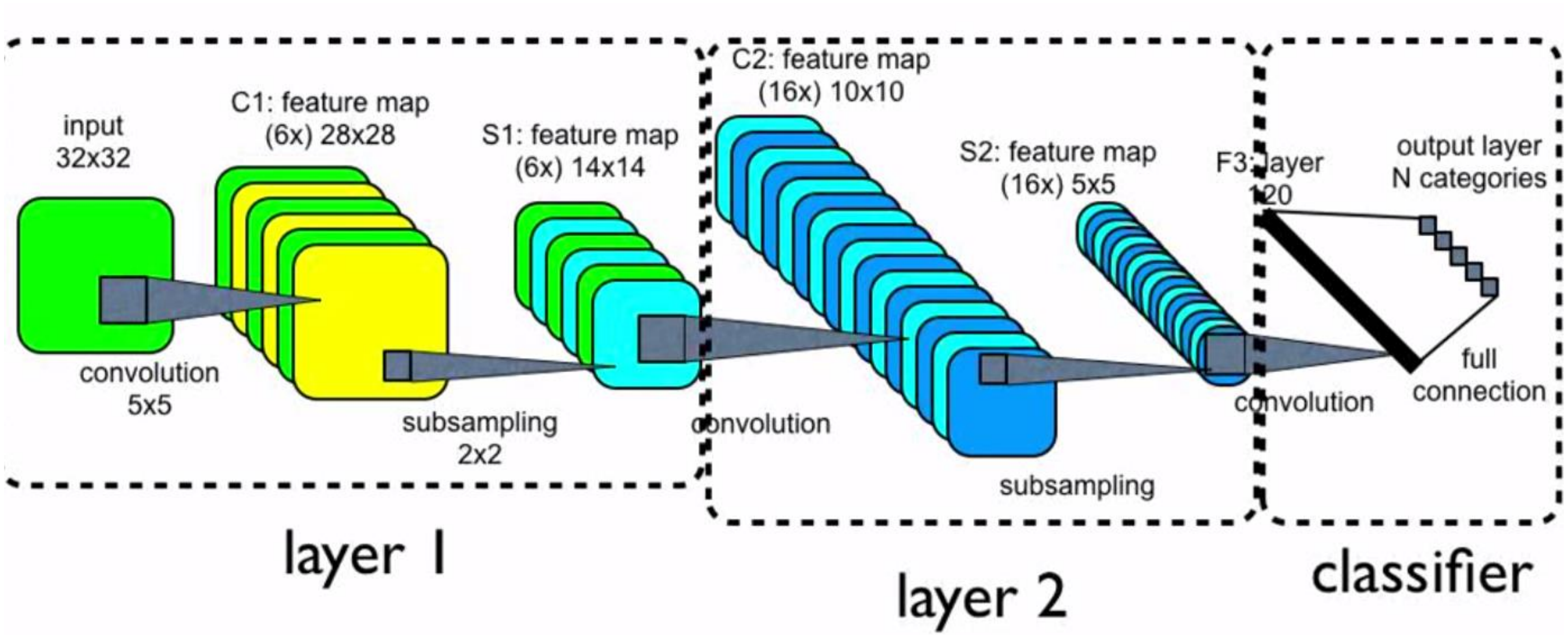
The Other Activation Function - Softmax



How Softmax works.

- Applying standard exponential function to each output layer element and normalizing it.
- Gives a probability distribution

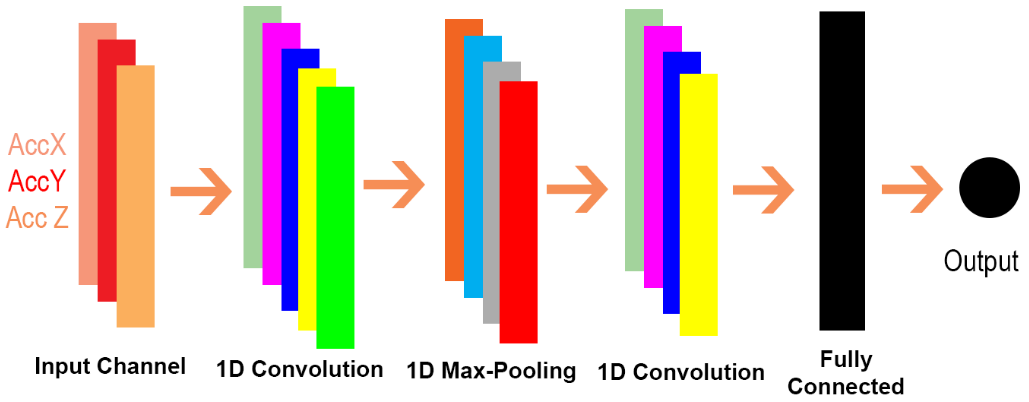
Big Picture



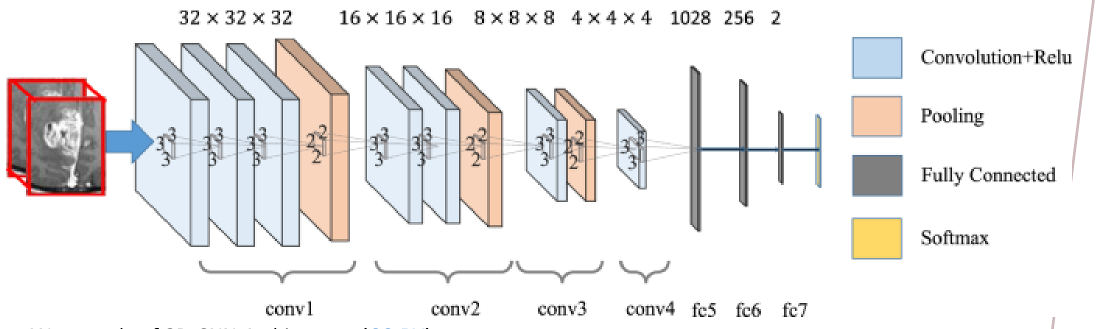
$$W_2 = \frac{(W_1 - F + 2P)}{S} + 1$$
$$H_2 = \frac{(H_1 - F + 2P)}{S} + 1$$
$$D_2 = K$$

Further Applications + Architecture

1D CNN	2D CNN	3D CNN
Natural Language Processing	Image Classification	Object segmentation in 3D imaging (medical)
Time-series forecasting	Audio Processing	Action detection (temporal)



An example of 1D CNN Architecture (CC BY-SA).



AN example of 3D CNN Architecture (CC BY)

BREAK & INTERVIEW QUESTIONS

Practical Session

https://tinyurl.com/MLDA-CNN2021

BREAK & INTERVIEW QUESTIONS

Q&A

Now is your turn to ask us questions!

Moving Forward

- Own a project:
    - Face masks detection
    - Named Entity Recognition
    - Time series forecasting
    - Natural Language Processing
- Industry projects under MLDA: https://forms.gle/Fxue6vwNKYoK4Bsg6
- Learn more about different types of CNN or Neural networks online or through MLDA workshops

THANK YOU!

https://tinyurl.com/mlda2021-cnn-feedback

Please help fill in the feedback form , we appreciate it! The link to the additional materials is available in the feedback form.