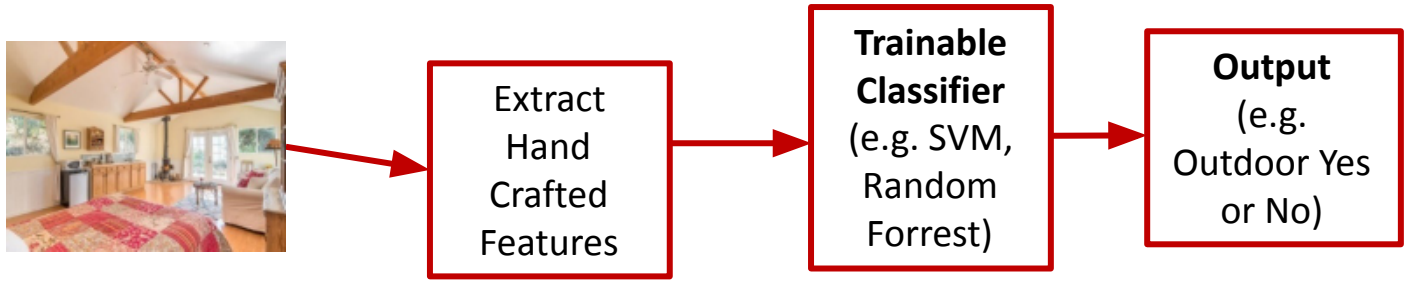
Foundational Concepts of Convolutional Neural Networks

Traditional Machine Learning

- **Traditional pattern recognition models work with hand crafted features and relatively simple trainable classifiers.**

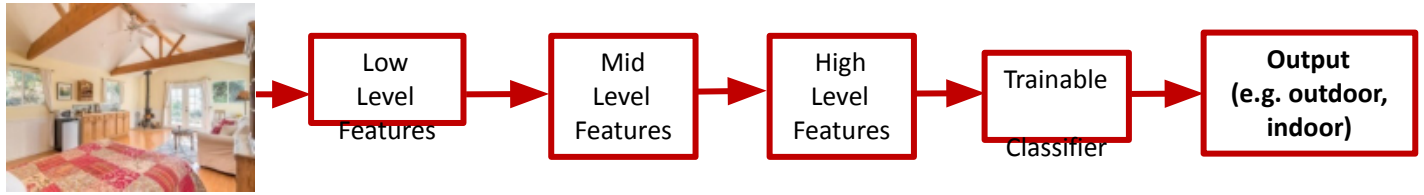


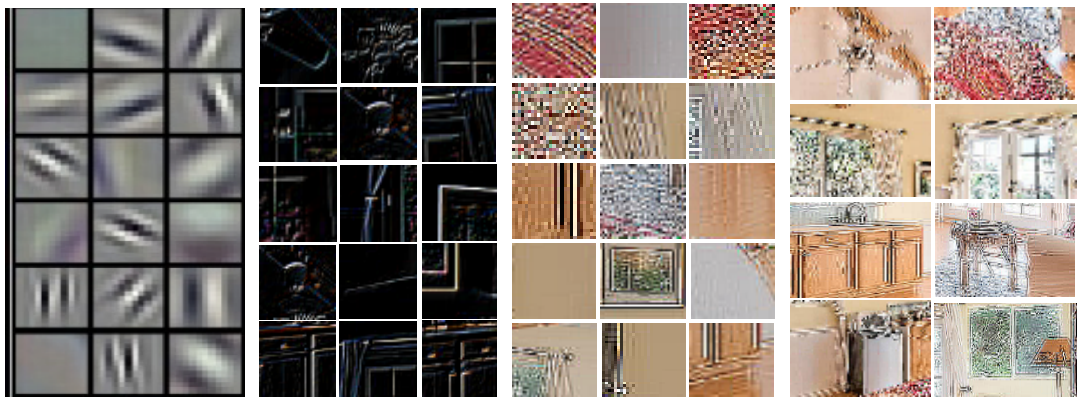
Limitations

- **Very tedious and costly to develop hand crafted features.**
- **The hand-crafted features are usually highly depend on one application.**

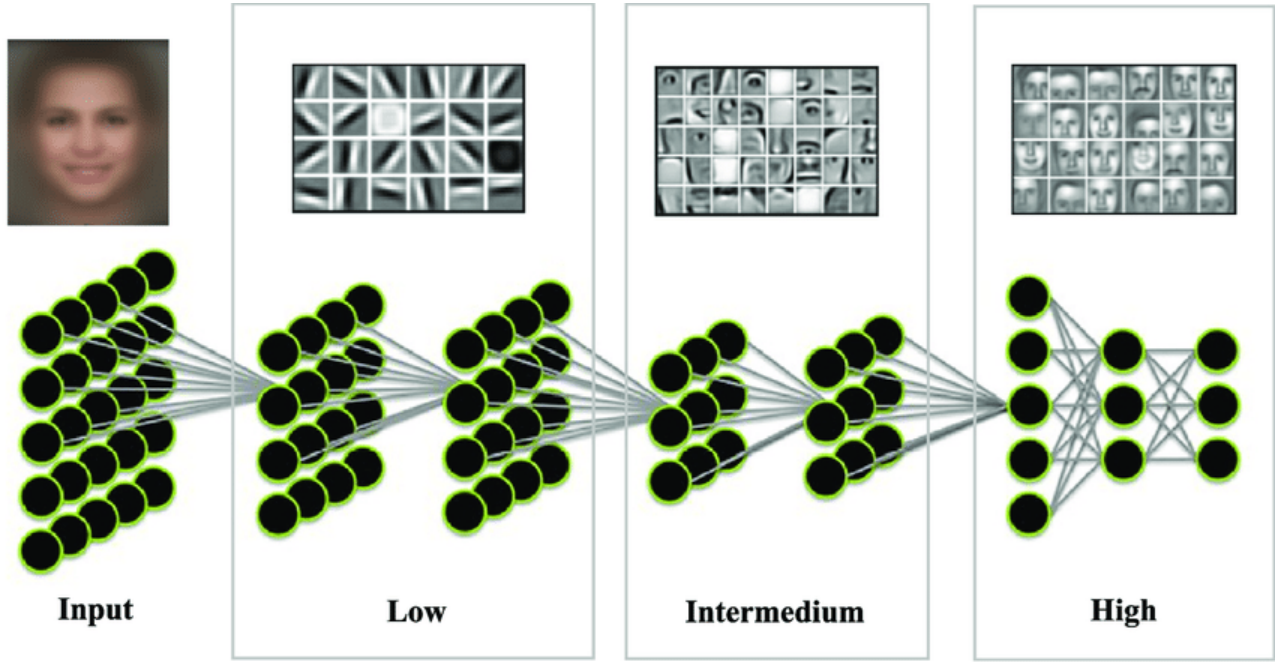
Deep Learning

- **Deep learning has an inbuilt automatic multi stage feature learning process that learns rich hierarchical representations (i.e. features).**





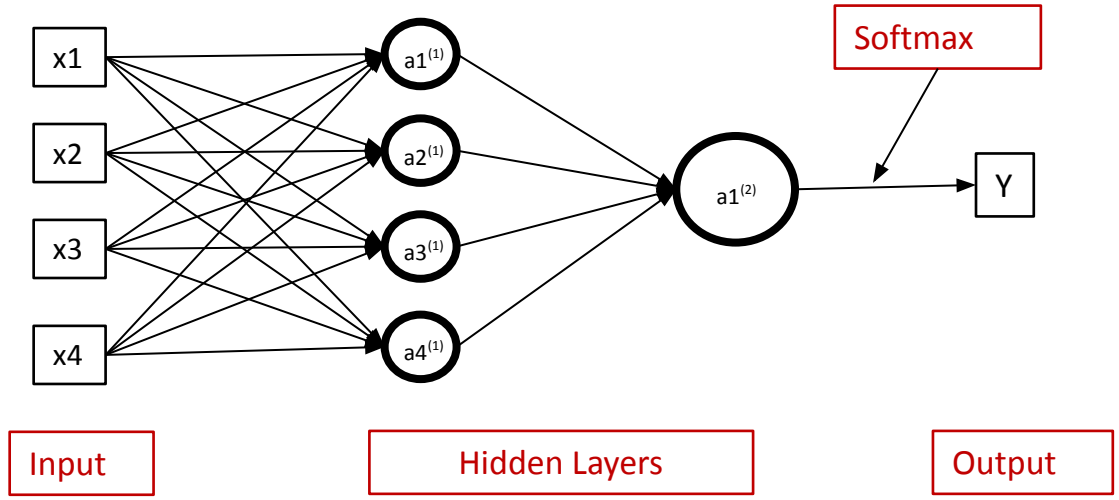
Face Recognition Problem



Research gate

Drawbacks of Fully Connected Neural Networks !!

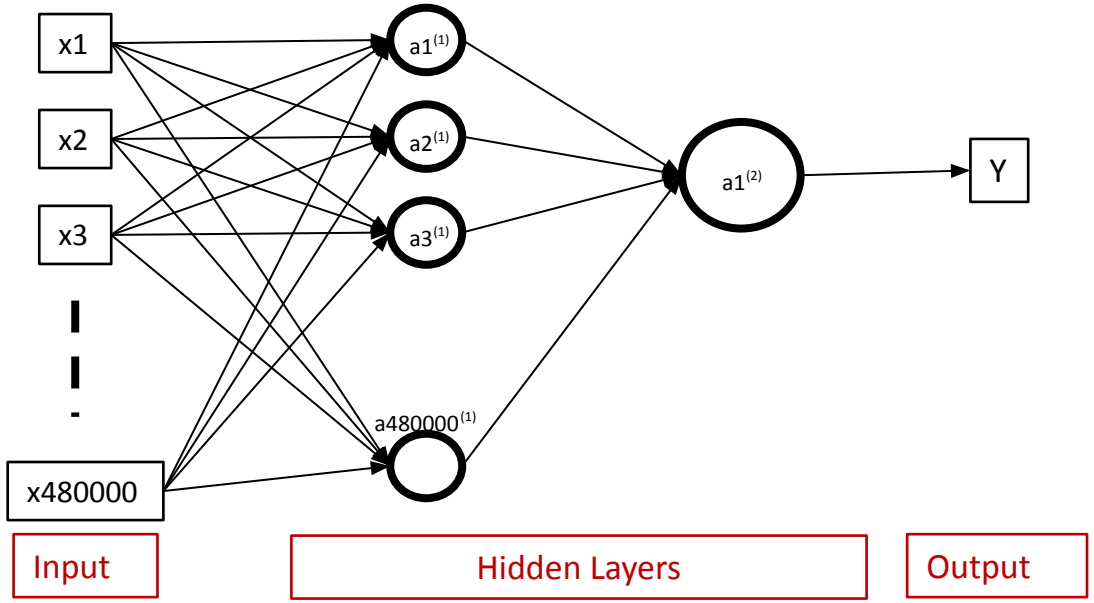
Simple Neural Network



Number of parameters = 4*4 + 4 +1 = 21

If the input is an Image?

400 X 400 X 3

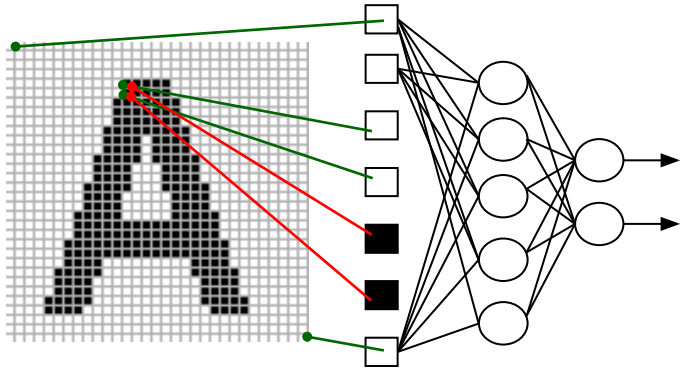


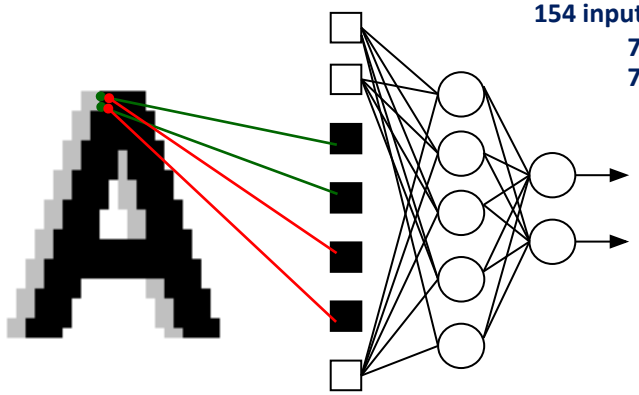
Number of Parameters

480000*480000 + 480000 +1 = approximately 230 Billion !!!

480000*1000 + 1000 +1 = approximately 480 million !!!

Little or no invariance to shifting, scaling, and other forms of distortion





154 input change from 2 shift left
77 : black to white
77 : white to black

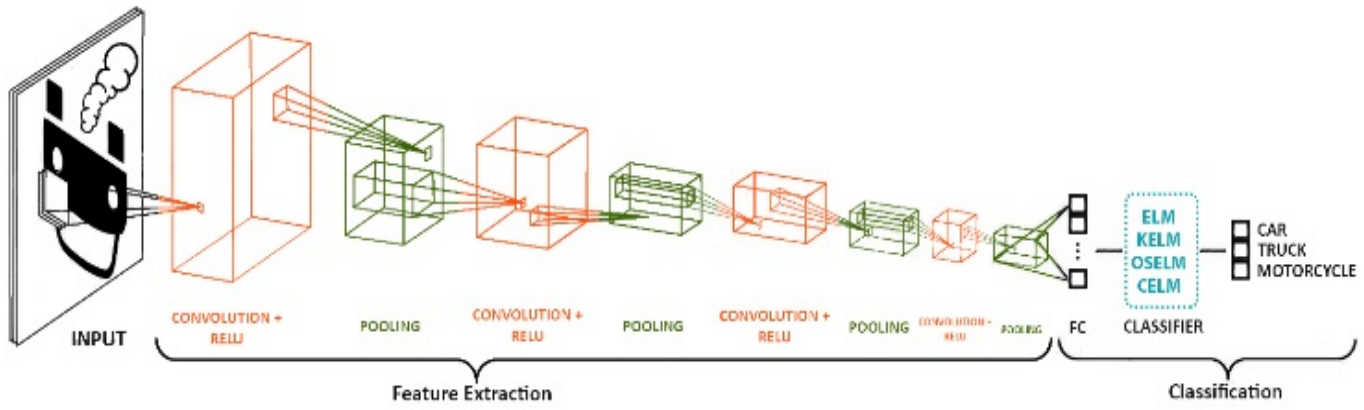
Scaling and other
forms of distortion

No Spatial Awareness
Inefficient Learning of Local Features

How Convolutional Neural Networks (CNN) Helps ?

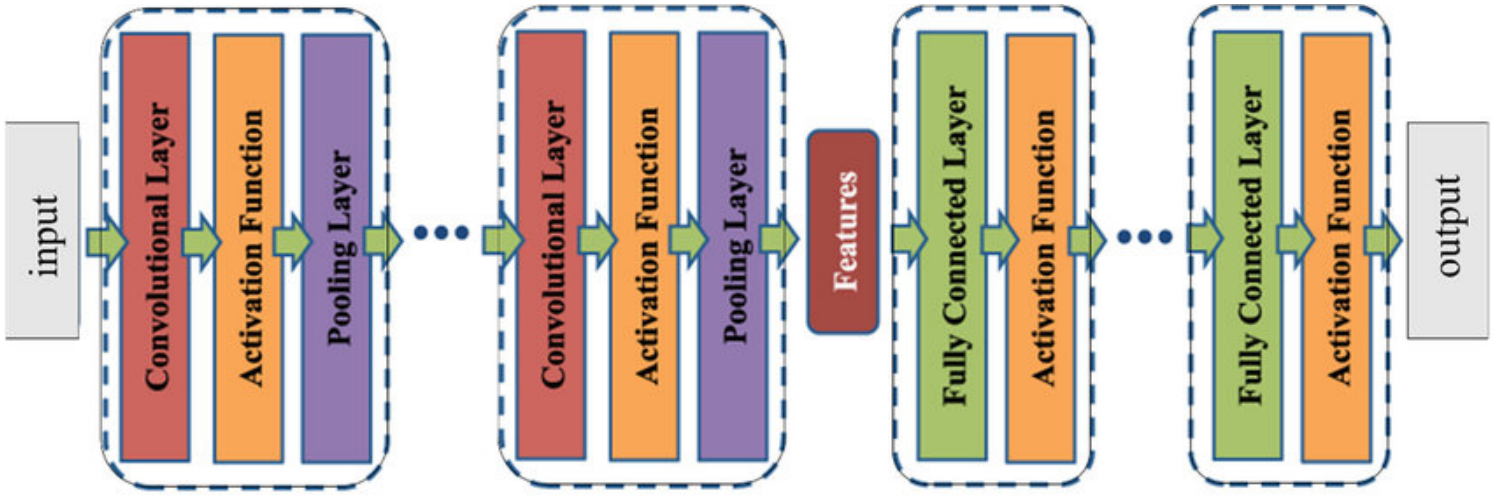
What is CNN ?

- A sub-category of neural networks
- Outperforms other models on image data.
- Architecture is composed of two main blocks.
  - Feature extraction phase
  - Classification phase

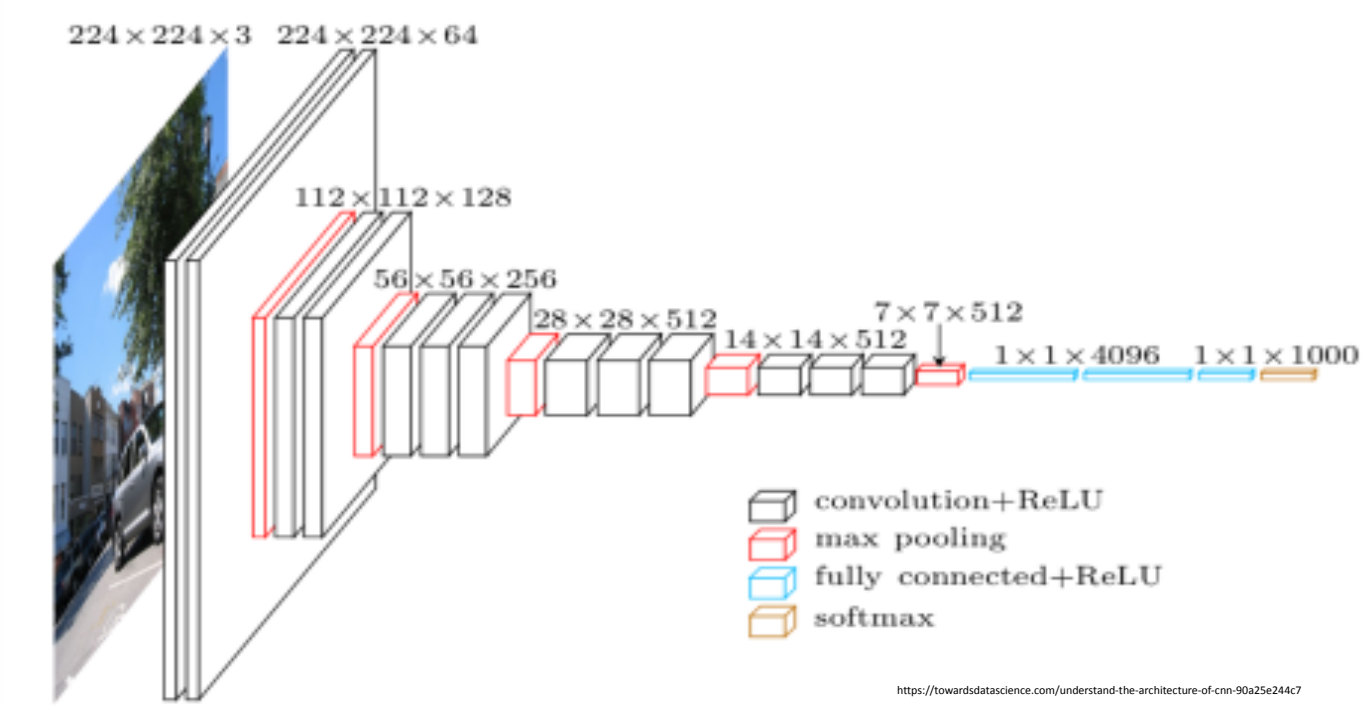


http://norc.aut.ac.ir/convolution-neural-network-joint-with-mixture-of-extreme-learning-machines-for-feature-extraction-and-classification-of-accident-images/

CNN Architecture



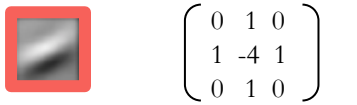
Understanding CNN Architecture



https://towardsdatascience.com/understand-the-architecture-of-cnn-90a25e244c7

Convolutional Layers

- **Filter**



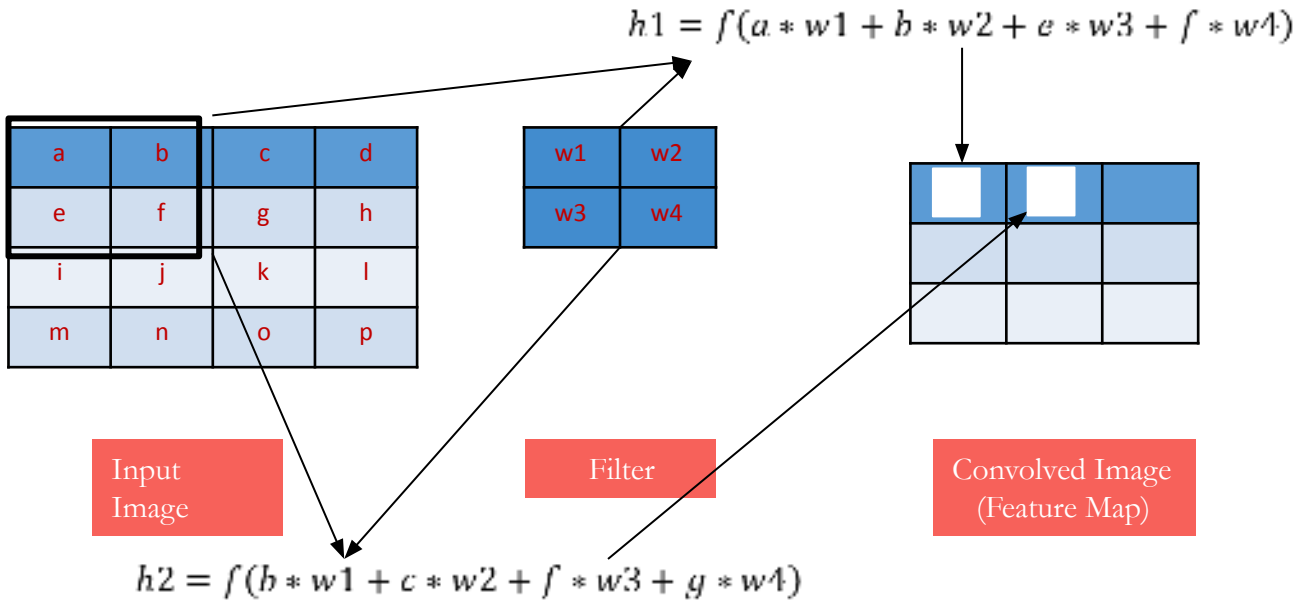
$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Input Image

Convoluted Image

Convolutional Layers

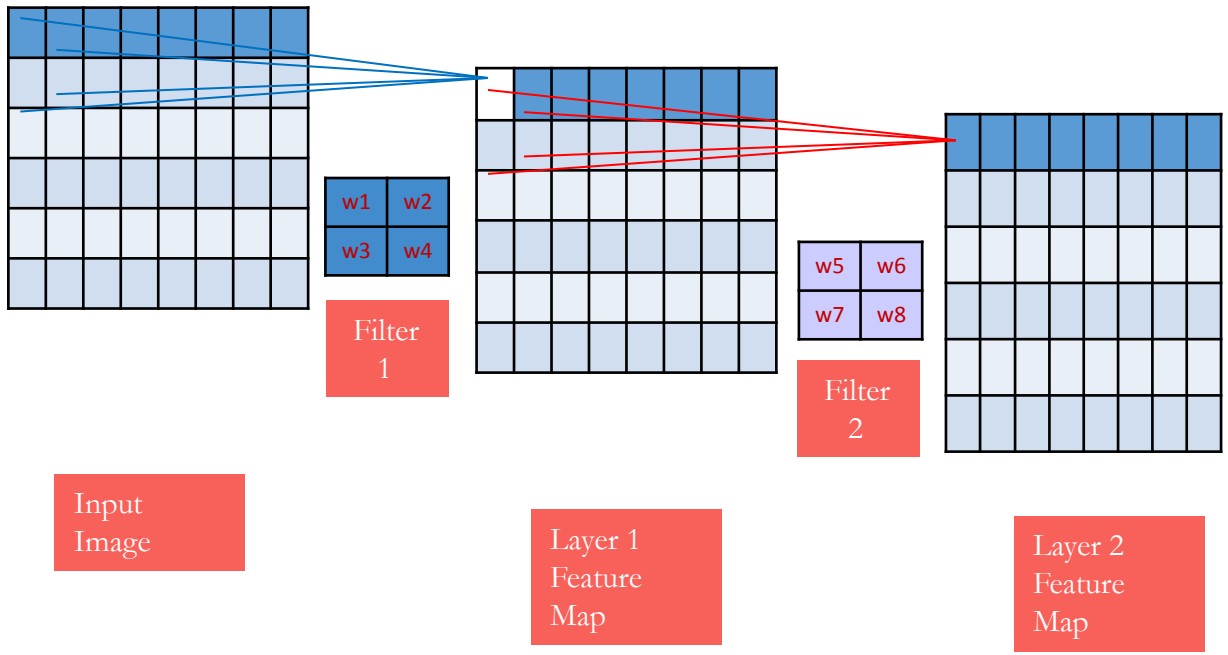
- **What is Convolution?**



Number of Parameters for one feature map = 4

Number of Parameters for 100 feature map = 4*100

Lower Level to More Complex Features



- In Convolutional neural networks, hidden units are only connected to local receptive field.

Understanding Convolutional Filters

- Convolutions have been used in image and signal processing for a long time.
- Convolutions in machine learning are different than those in image processing.
- Instead of using these filters, we can create our own as well and treat them as a parameter which the model will learn using backpropagation.

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Blur

0	-1	0
-1	5	-1
0	-1	0

Sharpen

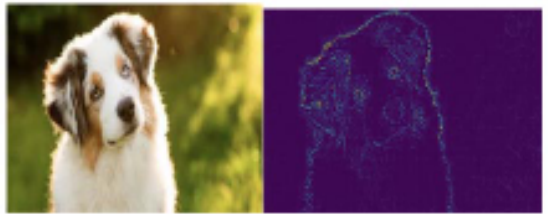
-1	-2	-1
0	0	0
1	2	1

Horizontal

-1	0	1
-2	0	2
-1	0	1

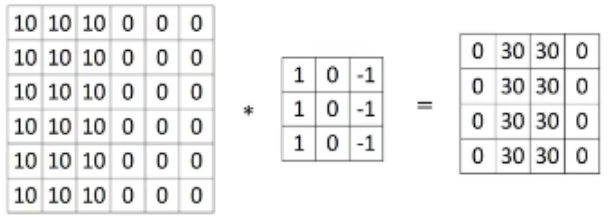
Vertical

Edges



https://www.saama.com/different-kinds-convolutional-filters/

- **Question1: convolving an input of 6 X 6 dimension with a 3 X 3 filter results in _____output.**
  - **Answer: 4 X 4**



- **Question 2: If the input is n X n and the filter size is f X f, What will be the output size ?**
  - **Input: n X n**
  - **Filter size: f X f**
  - **Output: (n-f+1) X (n-f+1)**

Padding

- Without padding
  - Input: n X n
  - Filter size: f X f
  - Output: (n-f+1) X (n-f+1)
- There are primarily two disadvantages here:
  - Every time we apply a convolutional operation, the size of the image shrinks
  - Pixels present in the corner of the image are used only a few number of times during convolution as compared to the central pixels. Hence, we do not focus too much on the corners since that can lead to information loss
- To overcome these issues, we can pad the image with an additional border, i.e., we add one pixel all around the edges.

Padding

- If we have 6 X 6 input matrix, after zero-padding it will become 8 X 8 matrix.
- Applying convolution of 3 X 3 on it will result in a 6 X 6 matrix which is the original shape of the image.
  - Input: n X n
  - Padding: p
  - Filter size: f X f
  - Output: (n+2p-f+1) X (n+2p-f+1)

- There are two common choices for padding:
  - Valid: It means no padding. If we are using valid padding, the output will be (n-f+1) X (n-f+1)
  - Same: Here, we apply padding so that the output size is the same as the input size, i.e.,
    $n+2p-f+1 = n$
    So, $p = (f-1)/2$

Strided Convolutions

- Suppose we choose a stride of 2. So, while convoluting through the image, we will take two steps – both in the horizontal and vertical directions separately. The dimensions for stride *s* will be:
  - Input: n X n
  - Padding: p
  - Stride: s
  - Filter size: f X f
  - Output: [(n+2p-f)/s+1] X [(n+2p-f)/s+1]
- Stride helps to reduce the size of the image, a particularly useful feature.

Convolutions Over Volume

- **Suppose, instead of a 2-D image, we have a 3-D input image**

- **How will we apply convolution on this image?**
  - **We will use a 3D filter instead of a 2D filter.**

- **The dimensions In the figure represent the height, width and channels in the filter.**

- ***Keep in mind that the number of channels in the input and filter should be same.***

https://obaydakov.github.io/post/2018/what-do-you-mean-by-1d-2d-and-3d-convolutions-in-cnn/

Convolutions Over Volume: Example

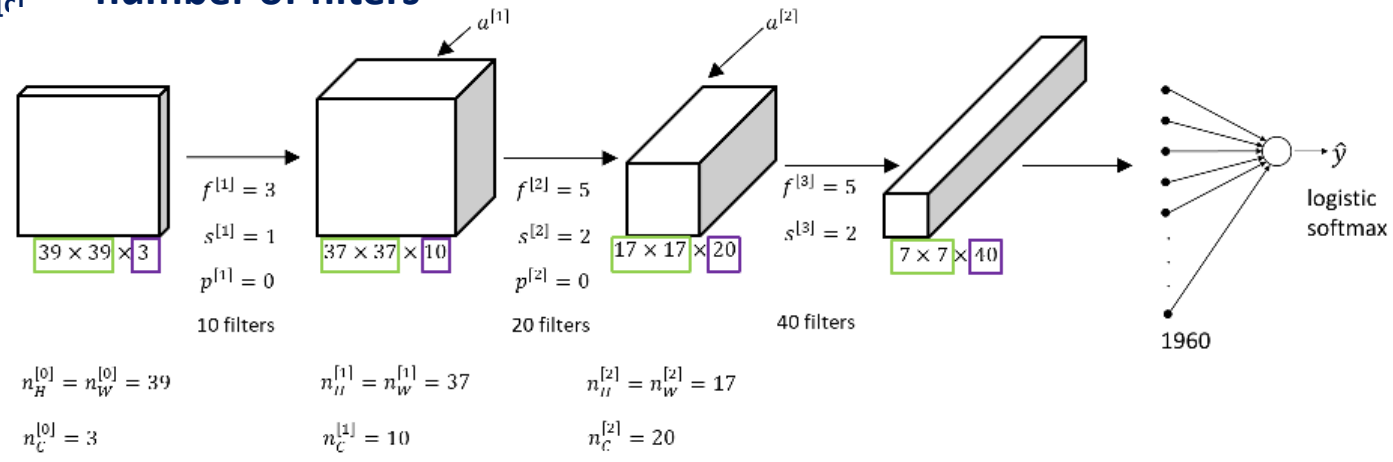
- Generalized dimensions can be given as:
  - Input: n X n X n_c
  - Filter: f X f X n_c
  - Padding: p
  - Stride: s
  - Output: [(n+2p-f)/s+1] X [(n+2p-f)/s+1] X n_c'
- Here, n_c is the number of channels in the input and filter, while n_c' is the number of filters.

One Layer of a Convolutional Network

- Convolving over the entire image using a filter
- Add a bias term to convolution outputs
- Finally apply an activation function to generate activations.
- *This is one layer of a convolutional network*.
- Question: Let the input size is (6 X 6 X 3) and filters size is (3 X 3 X 3). Suppose we have 10 filters, what will be the number of parameters in that layer?
  - Number of parameters for each filter = 3*3*3 = 27
  - There will be a bias term for each filter, so total parameters per filter = 28
  - As there are 10 filters, the total parameters for that layer = 28*10 = 280
- Clearly, the number of parameters in case of convolutional neural networks is independent of the size of the image. It essentially depends on the filter size.
- No matter how big the image is, the parameters only depend on the filter size. Awesome, isn't it?

Simple Convolutional Network Example

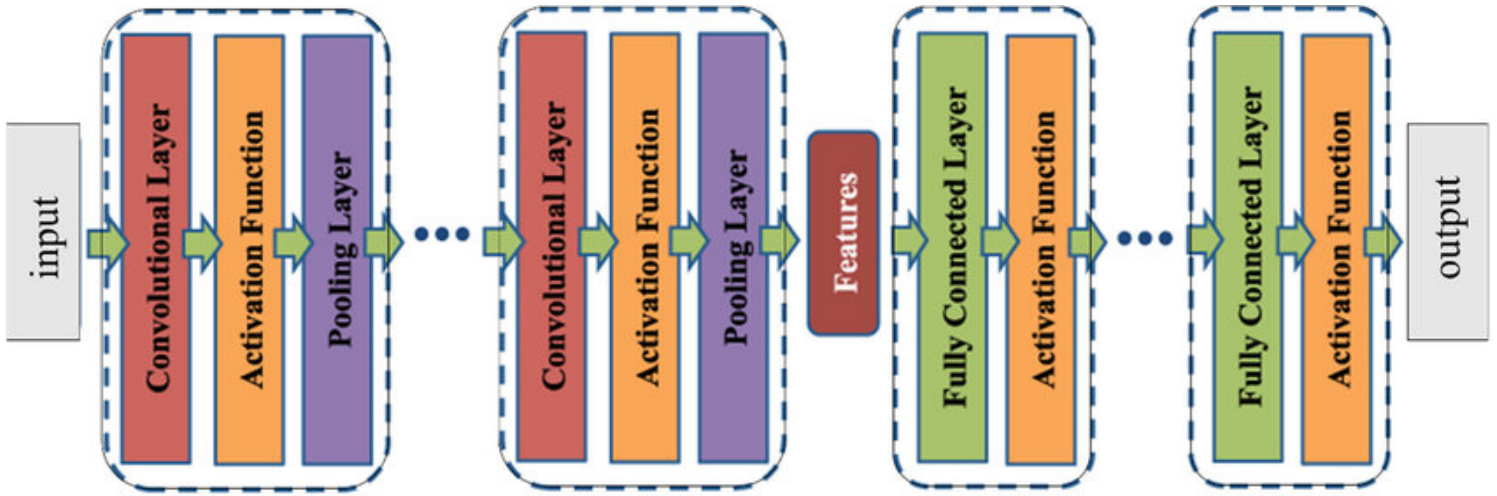
- $f^{[l]}$ = filter size
- $p^{[l]}$ = padding
- $s^{[l]}$ = stride
- $n_{[c]}^{[l]}$ = number of filters



- Layer 1: 10 comes from the fact that we use 10 filters and 37 comes from this formula $(n+2p-f)/s+1$.

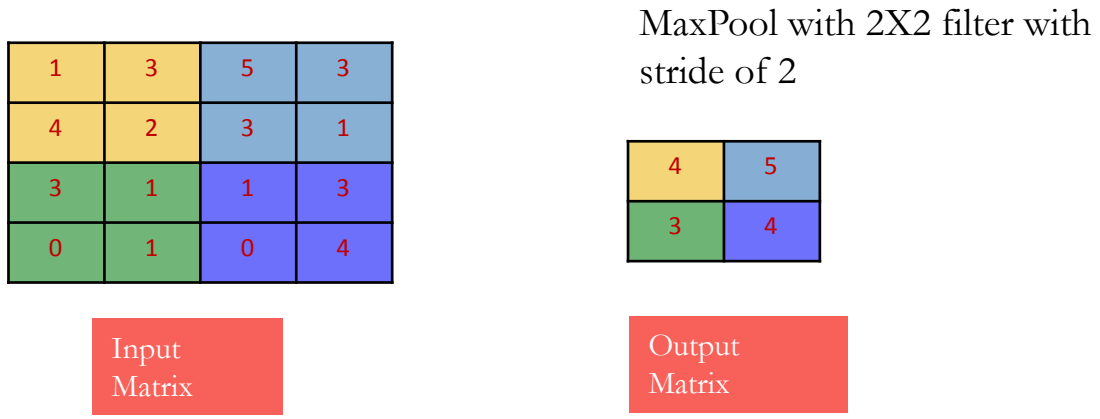
https://datahacker.rs/example-of-convolutional-neural-network/

CNN Architecture



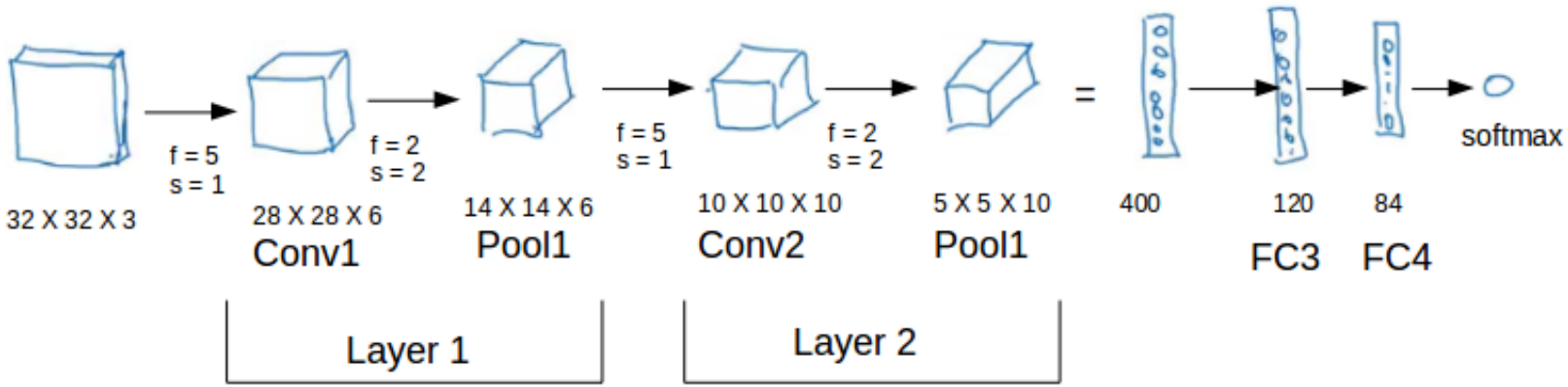
Pooling Layers

- Pooling layers are generally used to reduce the size of the inputs and hence speed up the computation.
- Retains the most important information.
  - Max pooling: reports the maximum output within a rectangular neighborhood.
  - Average pooling: reports the average output of a rectangular neighborhood.



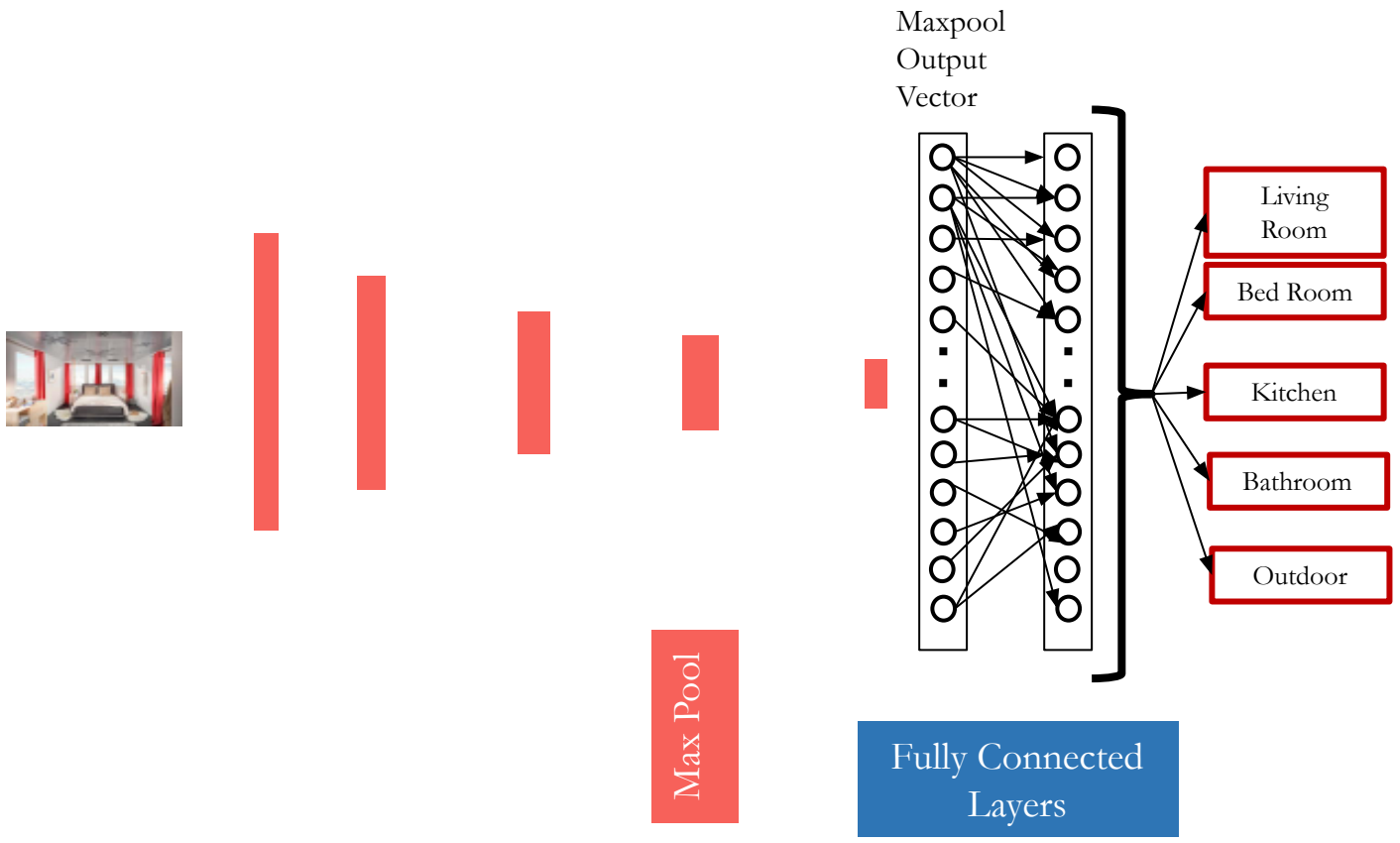
- Input of the pooling layer: n_h X n_w X n_c
- Output: $((n_h - f) / s + 1)$ X $((n_w - f) / s + 1)$ X n_c.

Example


f = 5
s = 1
f = 2
s = 2
f = 5
s = 1
f = 2
s = 2
=
softmax
32 X 32 X 3
28 X 28 X 6
Conv1
14 X 14 X 6
Pool1
10 X 10 X 10
Conv2
5 X 5 X 10
Pool1
400
120
FC3
84
FC4
Layer 1
Layer 2

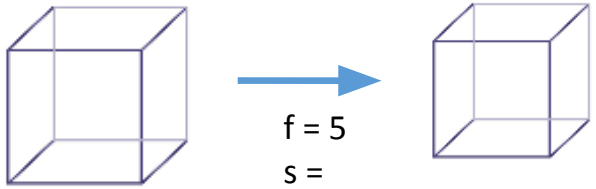
Convolutional Neural Network

Feature Extraction Architecture



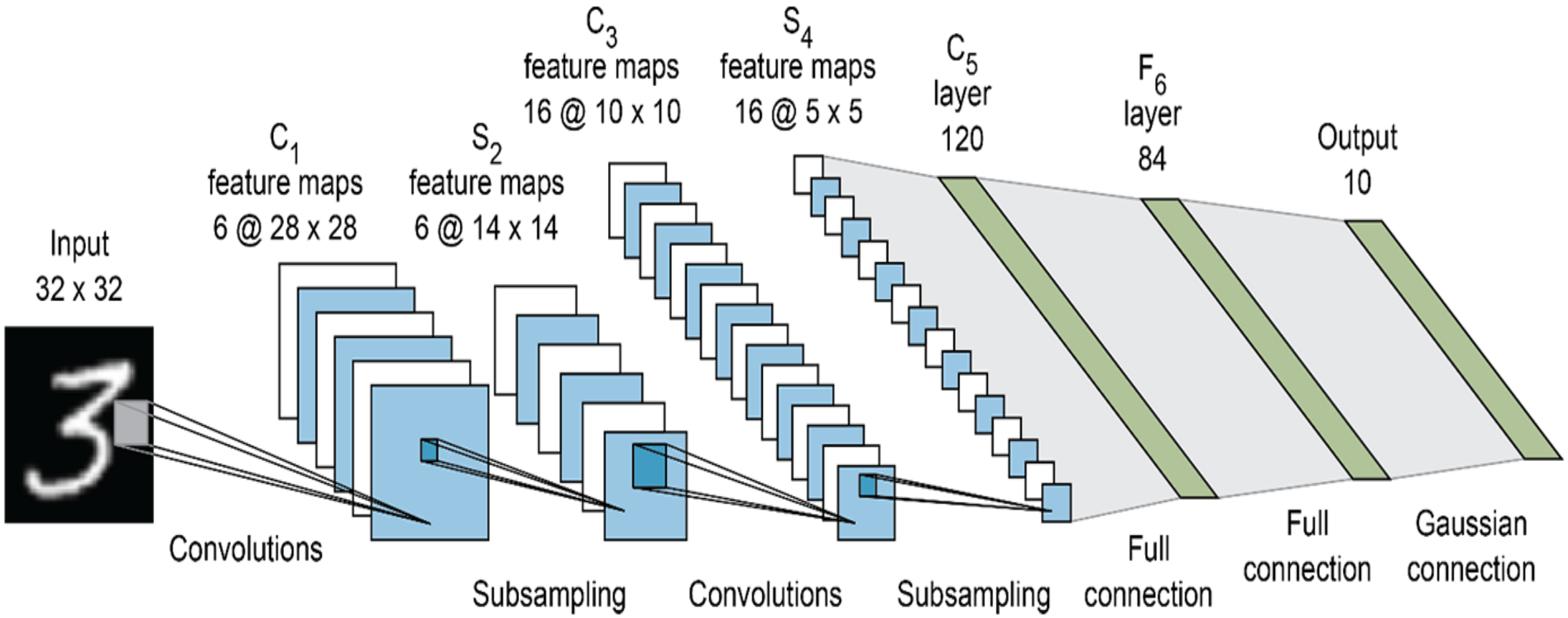
Why convolutions ?

- There are primarily two major advantages of using convolutional layers over using just fully connected layers:
  - Parameter sharing
    - In convolutional layers, the same filter (or kernel) is applied across the entire input.
    - This reduces the number of parameters compared to fully connected layers, where each input element has a unique weight.
  - Sparsity of connections
    - Convolutional layers connect each neuron to only a small region of the input (receptive field).
    - This localized connection focuses on relevant areas, capturing important local features like edges in images.
    - The model becomes more efficient and avoids unnecessary complexity compared to fully connected layers.
- How many parameters do we require if we use fully connected network for the above example ?
  - 32*32*3*28*28*6, which is nearly equal to 14 million!
  - Makes no sense, right?
- How many parameters if we use convolutional layer ?
  - (5*5 + 1) * 6 = 156



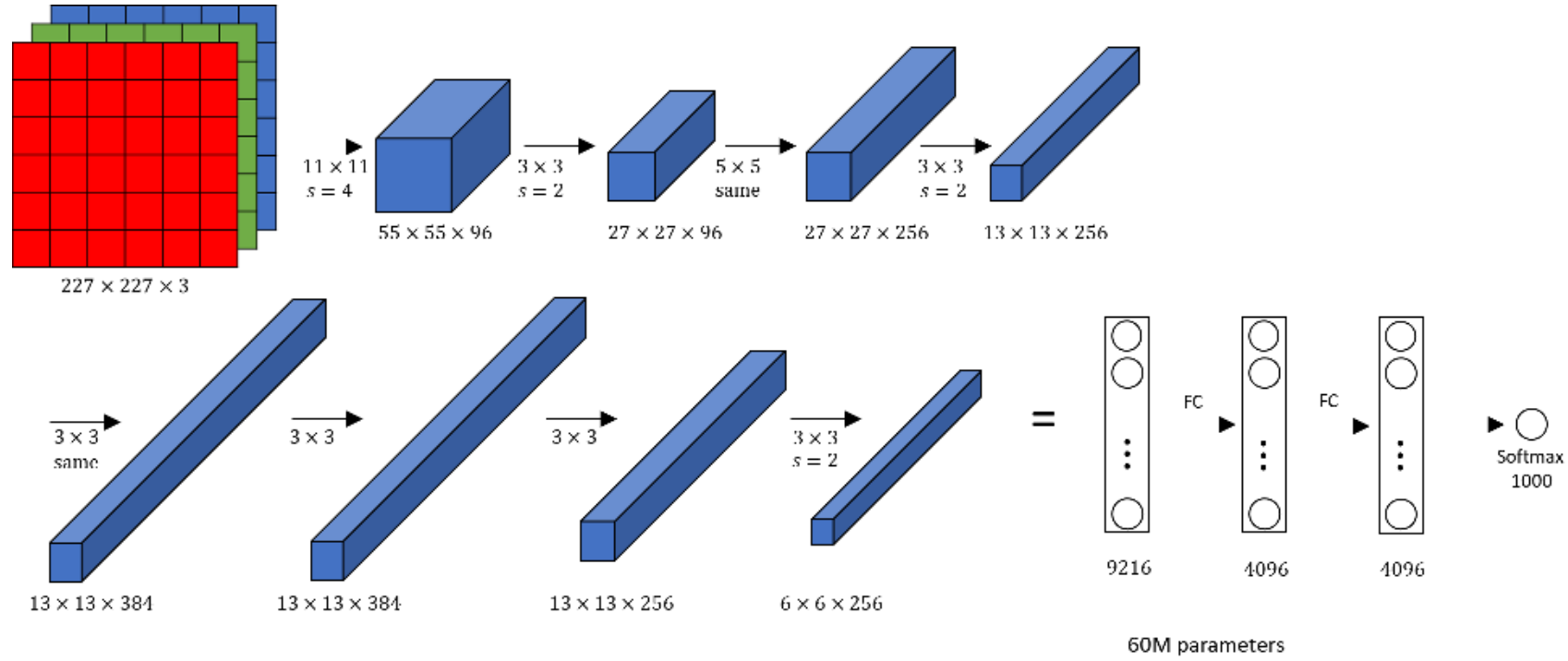
Classic Networks

LeNet-5


Input
32 x 32
C1
feature maps
6 @ 28 x 28
S2
feature maps
6 @ 14 x 14
C3
feature maps
16 @ 10 x 10
S4
feature maps
16 @ 5 x 5
C5
layer
120
F6
layer
84
Output
10
Convolutions
Subsampling
Convolutions
Subsampling
Full
connection
Full
connection
Gaussian
connection

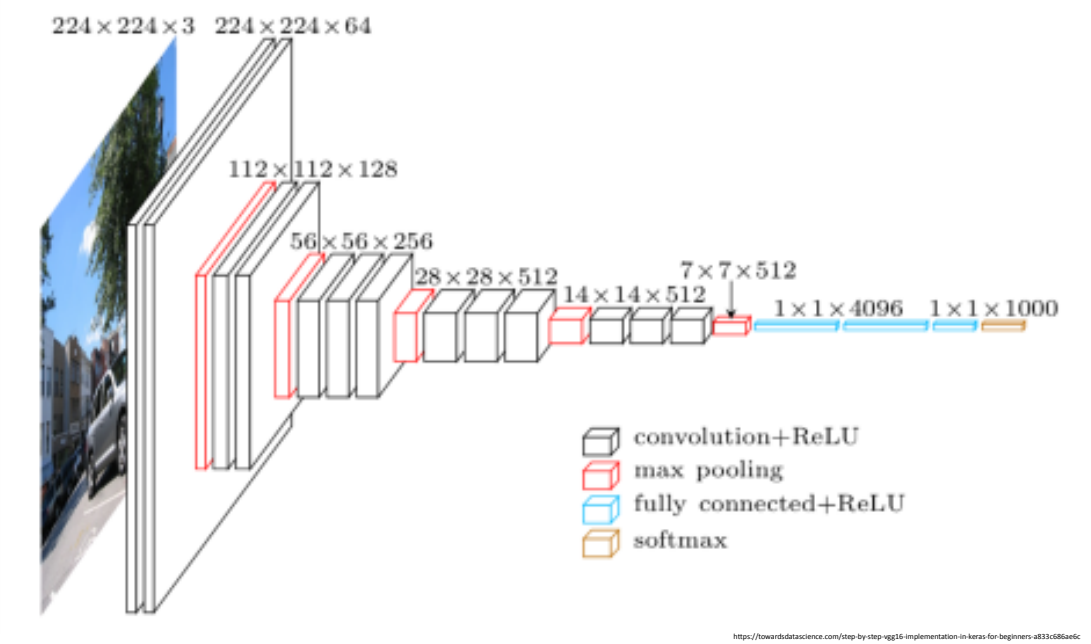
Alex Net

- Parameters: 60 million
- Activation function: ReLu



VGG-16

- **Parameters: 138 million**



https://towardsdatascience.com/step-by-step-vgg16-implementation-in-keras-for-beginners-a833c686ae6c

Many more …

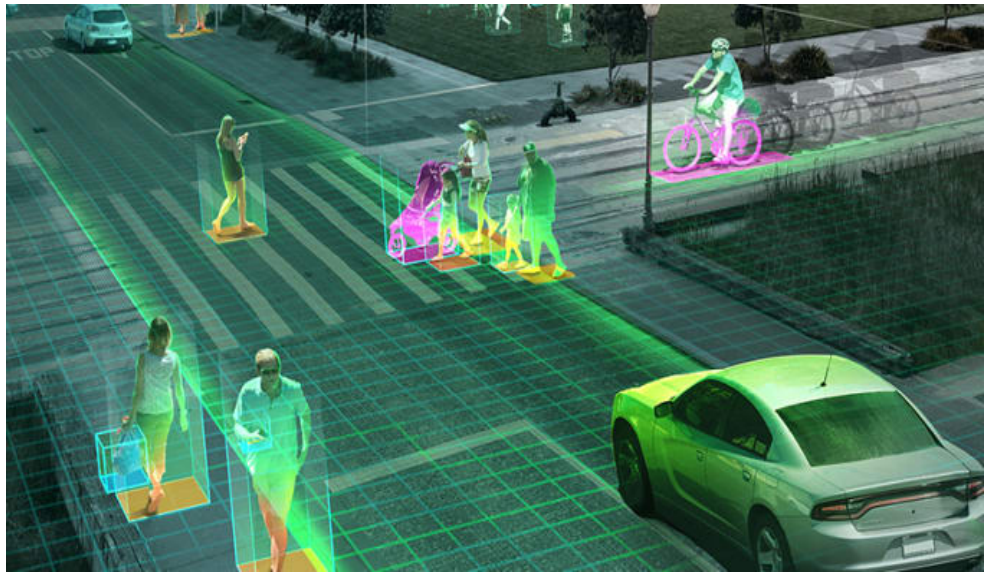
- Inception
- ResNet
- ResNeXt
- DenseNet
- …

Applications

- Image Classification
  - Search Engines, Recommender Systems, Social Media
- Face Recognition
  - Social media, entertainment, security
- Optical Character Recognition
  - Legal, Banking, Insurance, Document digitization
- Medical Image Computing
  - Risk analysis, anomaly detection, drug discovery
- Image Segmentation
- And many other

CNN for Image Segmentation

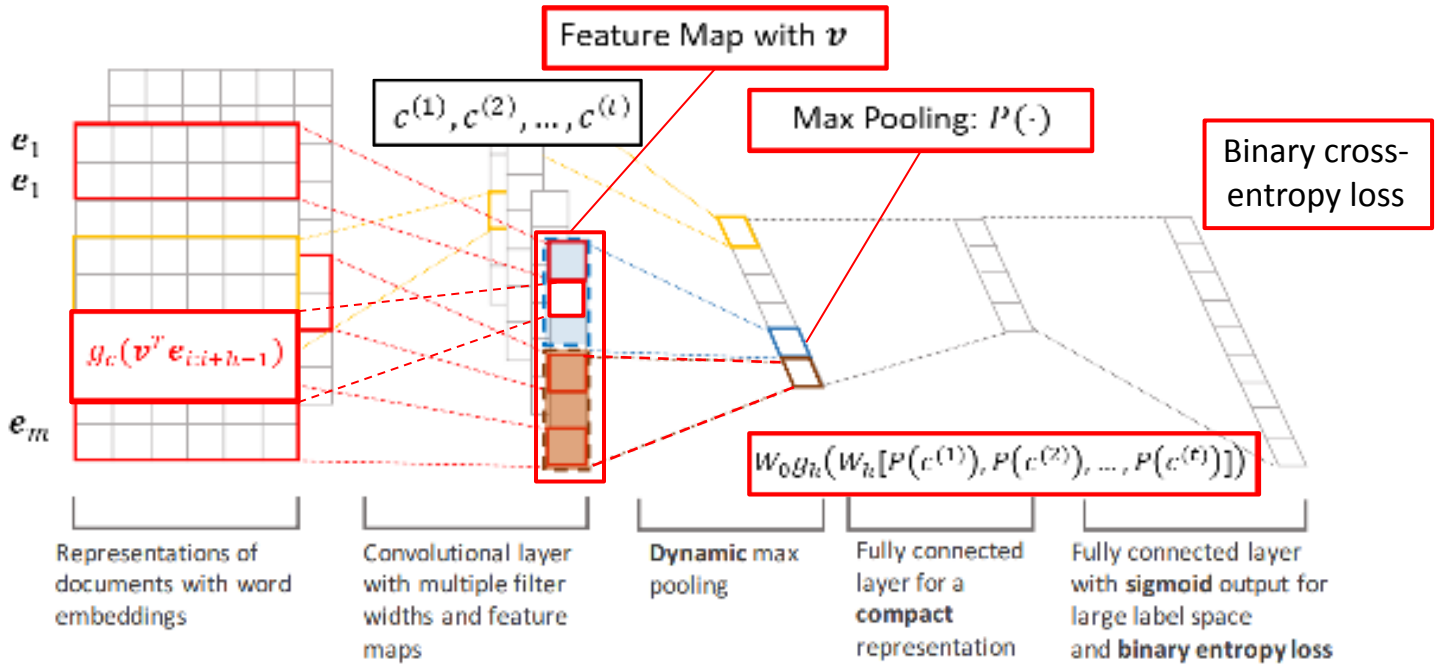
- *Dividing Images into Regions for Object Recognition*
  - *Medical Imaging*
  - *Autonomous Driving, etc.*



Deep Learning for Text Classification

- Multi-label text categorization is generally divided into two sub-tasks:
  - Text feature extraction
  - Multi-label classification
- Limitation
  - Words are treated as independent features out of context.

Deep Learning for Extreme Multi-label Text Classification



Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. "Deep learning for extreme multi-label text classification." *In Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*, pp. 115-124. 2017.

Backpropagation in CNNs

Zhifei Zhang. "Derivation of backpropagation in convolutional neural network (cnn)." *University of Tennessee, Knoxville, TN* 22 (2016): 23.

References

- Pattern recognition and machine leanring, Christopher M. Bishop.
- https://machinelearningmedium.com/2017/09/21/neural-networks/
- https://machinelearningmedium.com/2017/10/03/neural-networks-cost-function-and-back-propagation/
- https://machinelearningmedium.com/2018/03/20/backpropagation-derivation/
- https://machinelearningmedium.com/2017/09/27/neural-network-intuition/
- Machine Learning: Coursera.org
- Directional Decomposition for *Odia* Character Recognition, Chandana Mitra and Arun K. Pujari.
- https://www.analyticsvidhya.com/blog/2018/12/guide-convolutional-neural-network-cnn/
- Figures and examples are taken from various online sources.
- https://medium.com/@snaily16/what-why-and-which-activation-functions-b2bf748c0441#:~:text=ReLU%20activation%20function%20is%20widely,used%20in%20the%20hidden%20layers.
- https://blog.floydhub.com/long-short-term-memory-from-zero-to-hero-with-pytorch/
- https://towardsdatascience.com/transfer-learning-with-convolutional-neural-networks-in-pytorch-dd09190245ce
- https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be
- https://arxiv.org/pdf/1412.3555v1.pdf
- https://www.youtube.com/watch?v=9vB5nzrL4hY
- https://www.youtube.com/watch?v=2GPZlRVhQWY
- https://d2l.ai/chapter_recurrent-modern/gru.html
- https://colab.research.google.com/drive/1plN27Fgmmzgb9QhNziO02oLnJXVImYKC?authuser=1