جامعة قطر
QATAR UNIVERSITY

كلية الهندسة
COLLEGE OF ENGINEERING

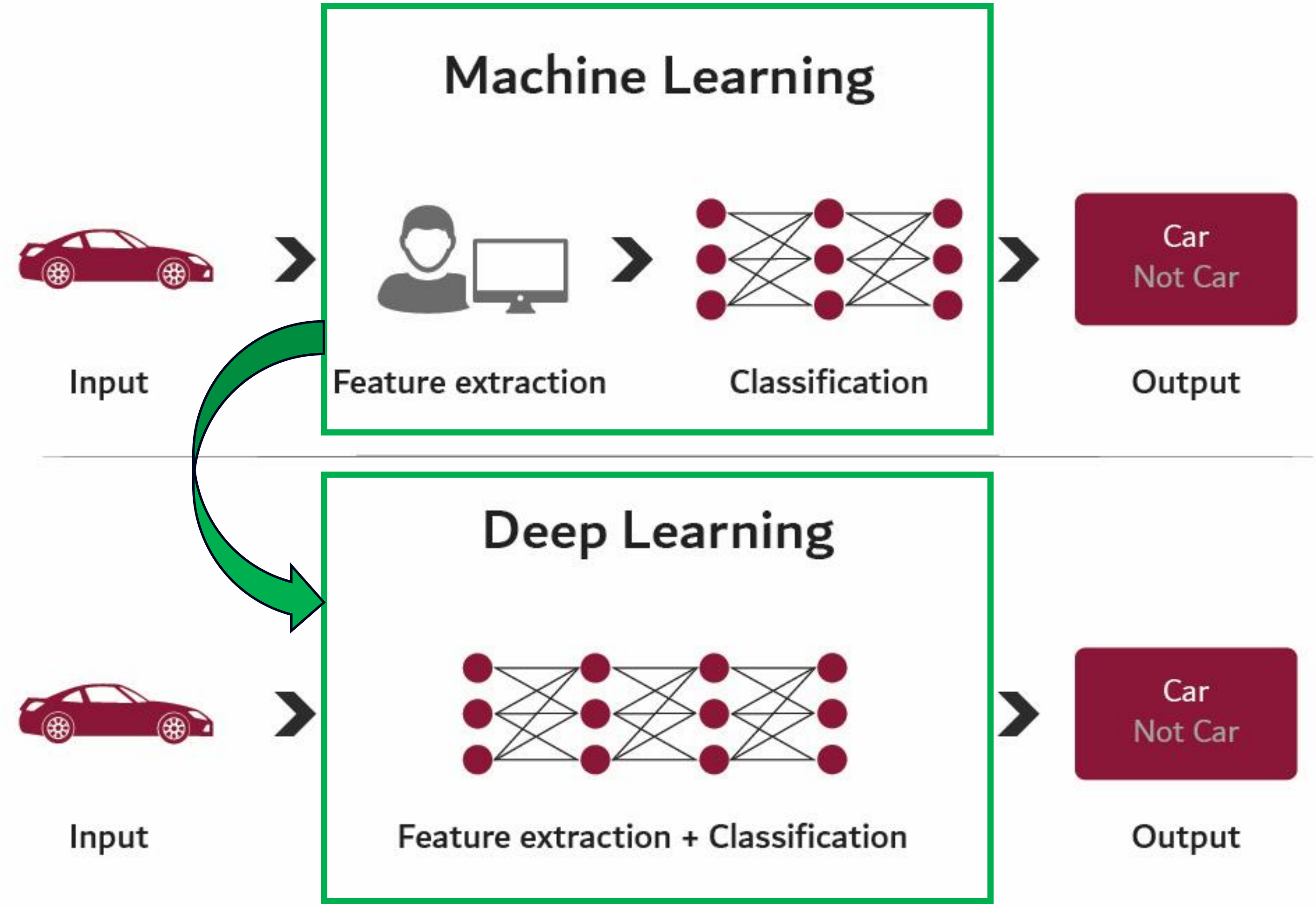
Exploring Deep Neural Networks

Dr. Ahmed Bensaid

Outline

- Machine Learning vs Deep Learning
- Why Deep Learning and Convolutional Neural Networks (CNN)
- Convolutional Neural Networks
- State-Of-Arts CNN models
- Recurrent Neural Networks (RNN)
- Regularizing Deep Neural Networks (DNN)
- Fine Tuning DNN
- Demos
- Conclusion

Classic Machine Learning vs Deep Learning



QATAR UNIVERSITY
COLLEGE OF ENGINEERING

Why Deep Learning and Convolutional Neural Networks (CNN)

What’s wrong with Fully Connected NN?

Data

1D grid: sequential data, e.g. univariate time series, tabular data

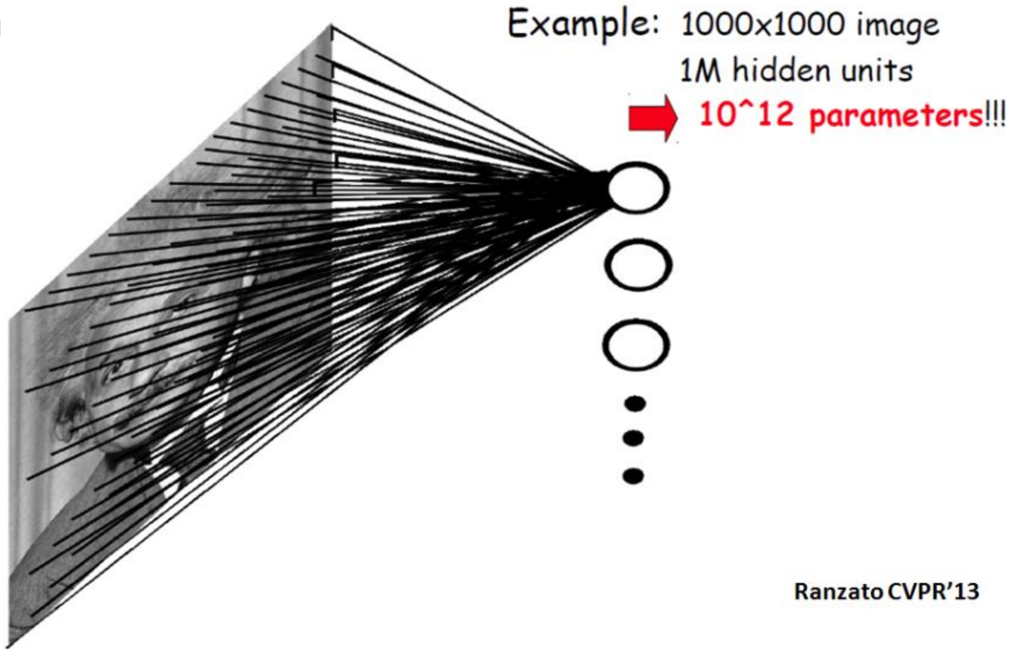
2D grid: natural images

3D grid: video, 3D image volumes (multi/hyper spectral images)

Loss of spacial information

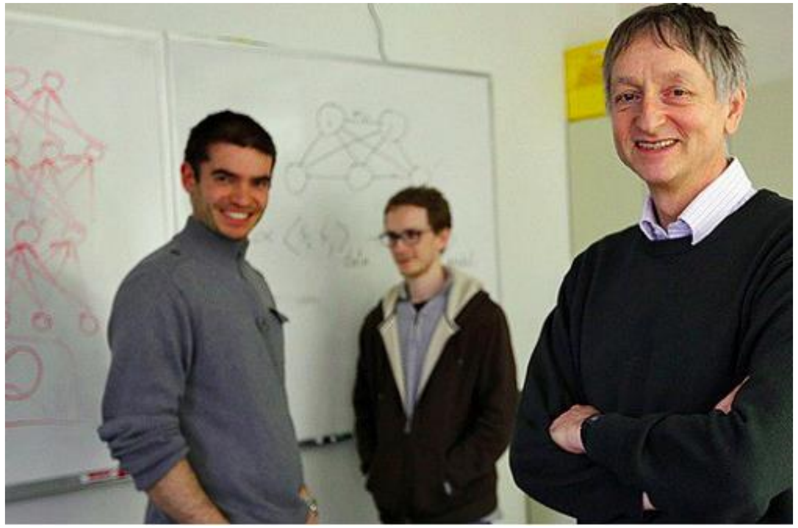
Distant pixels are less correlated

Larger input size ➔ Higher # params



Ranzato CVPR’13

Why deep learning and Convolutional NN?



Deep learning

- Has won numerous pattern recognition competitions
- Does so with minimal feature engineering

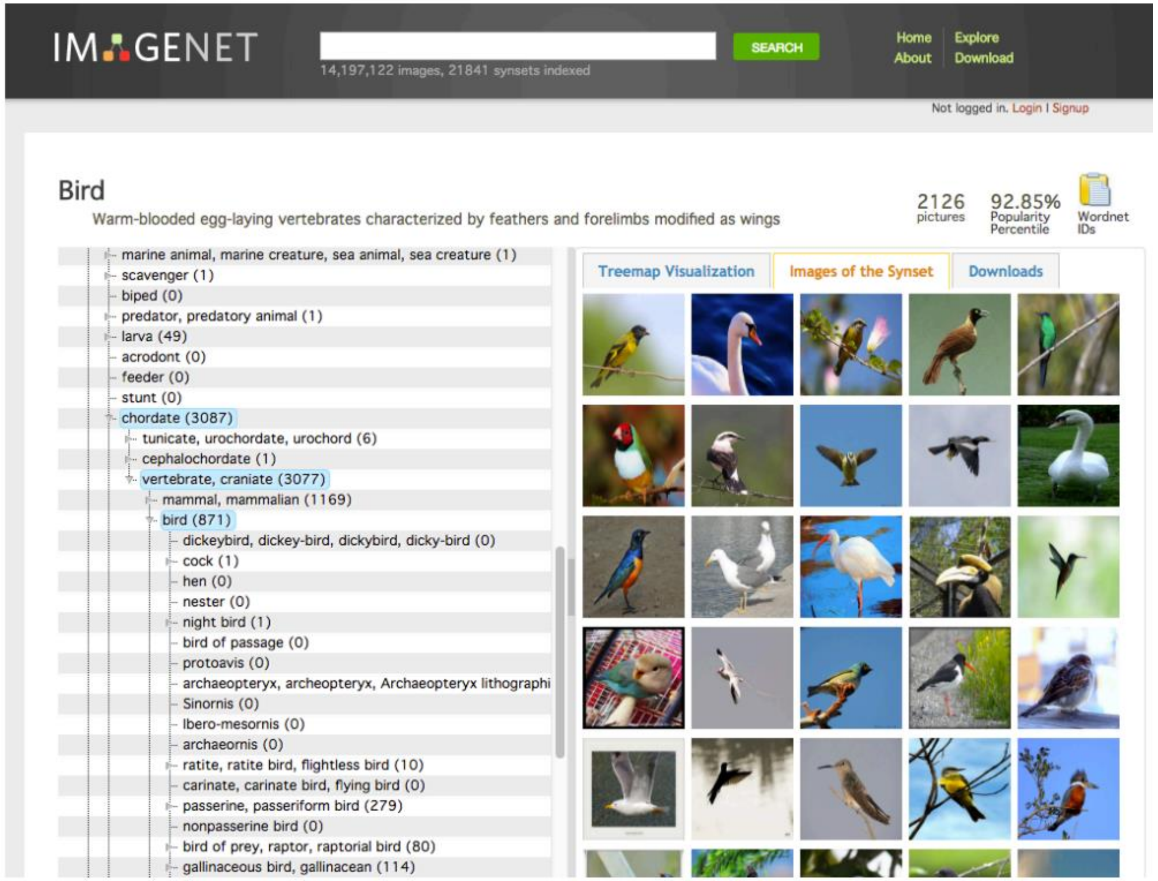
Billions of $ are invested in it

DeepMind: Acquired by Google for $400 million

DNNResearch: Three-person startup (including Geoff Hinton) acquired by Google for unknown price tag

Data are heterogeneous, (image, video, sequential, speech …)

IMAGENET and ILSVRC



14 million annotated images

Over 20.000 categories

ns

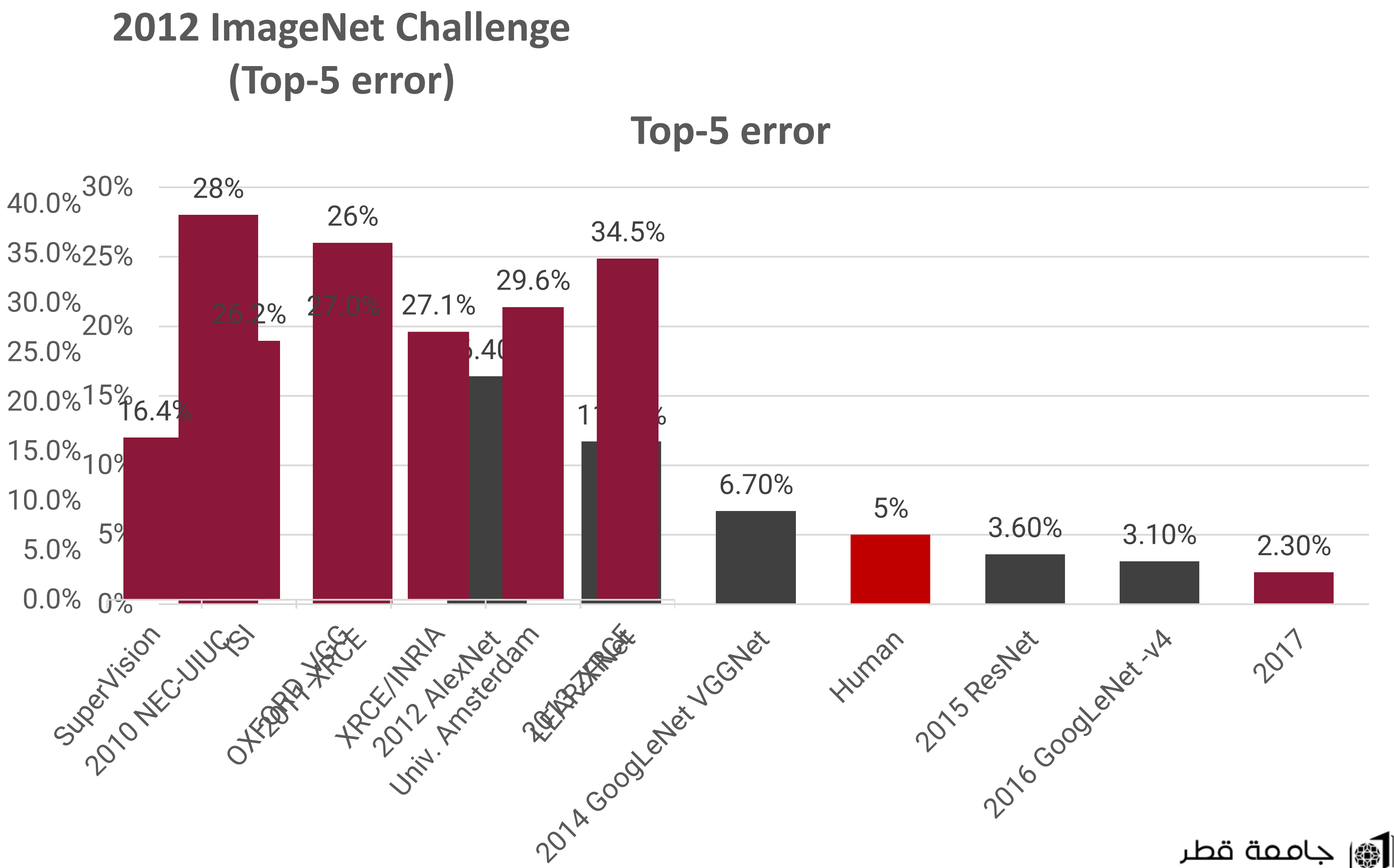
(typic et

tasks or

nd

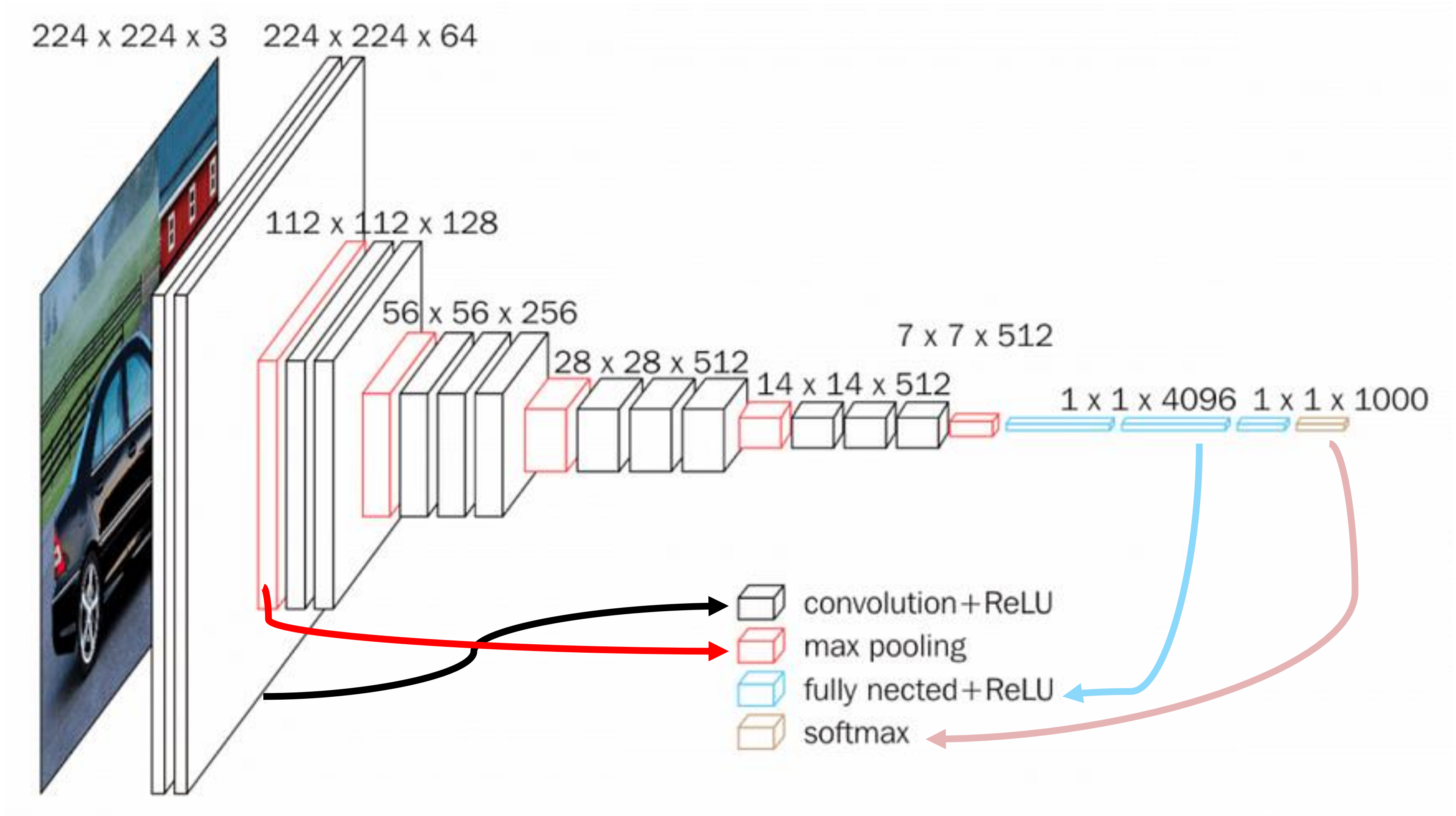
n.

IMAGENET and ILSVRC



Convolutional Neural Networks (CNN)

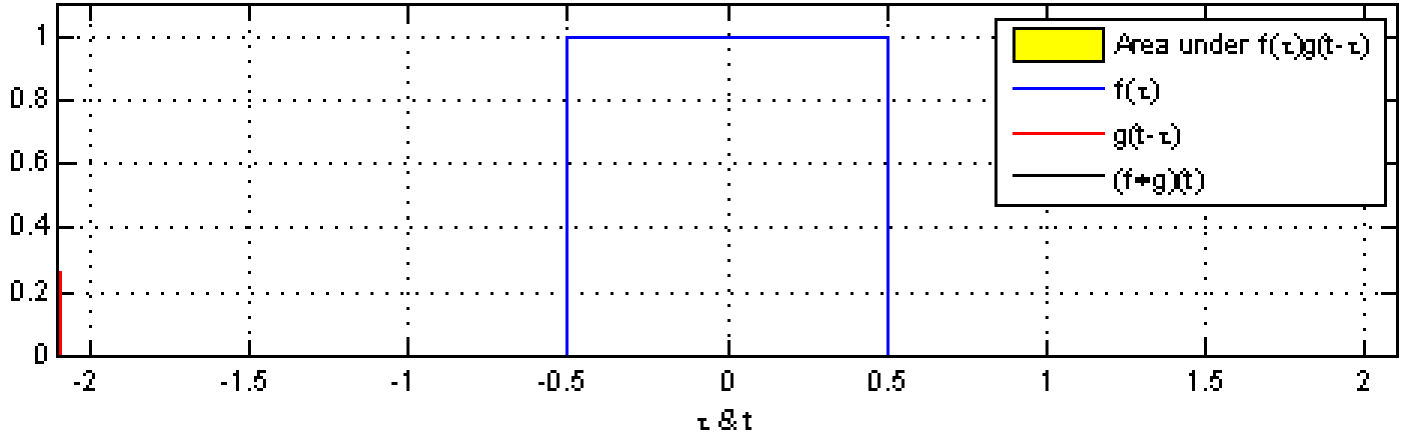
Convolutional Neural Network



Convolution

- To “convolve” means to roll together.
- Is an integral measuring how much two functions overlap as one passes over the other.

$$(f * g)(t) = \int f(\tau)g(t - \tau)d\tau$$



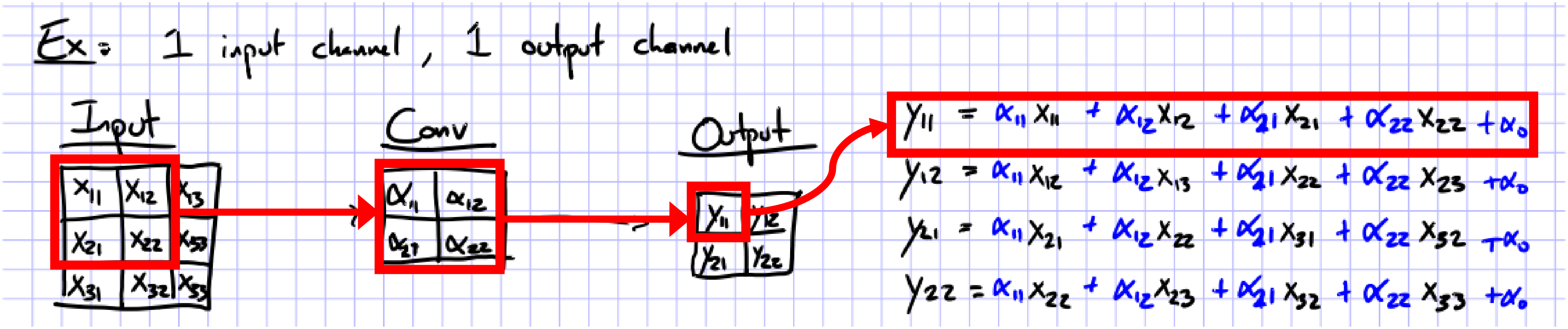
2D Convolution

Basic idea

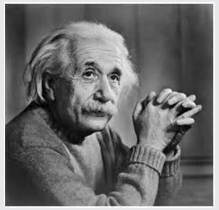
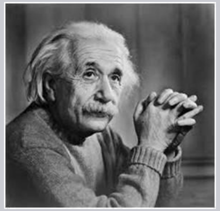
- Pick a 2x2 (for example) matrix (a.k.a kernel) F of weights
- Slide this over an image and compute the “inner product” (similarity) of F and the corresponding field of the image, and replace the pixel in the center of the field with the output of the inner product operation

Key point

- Different convolutions extract different types of low-level “features” from an image
- All that we need to vary to generate these different features is the weights of F



2D Convolutional Kernels

Kernel	Operation	Result (demo 1)
$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	Identity	
$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 1 & 0 \\ 1 & 2 & 1 \end{bmatrix}$	Sobel-Horizontal	 
$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$	Sobel-Vertical	 
$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	Sharpen	 

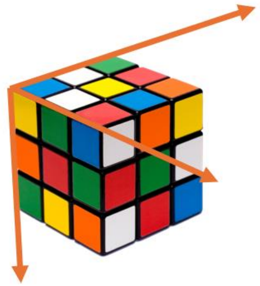
Side note about tensor

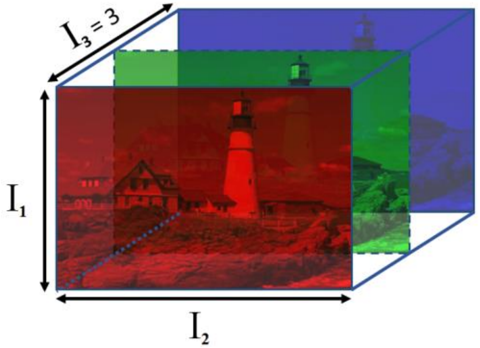
Scalar: $[a]$

Vector: $\begin{bmatrix} a \\ b \\ c \end{bmatrix}$

Matrix: $\begin{bmatrix} a_{11} & \cdots & a_{1N} \\ \vdots & \ddots & \vdots \\ a_{M1} & \cdots & a_{MN} \end{bmatrix}$

3D matrix:





A tensor is a general representation of data in any number of dimension

2D Convolution

Demo 2 (color image/tensor)

Input size: $W_1 \times H_1 \times D_1$

Filters:

K filters of size $F \times F$

Stride S

Zero Padding amount P

Output size: $W_2 \times H_2 \times D_2$

$W_2 = (W_1 - F + 2P)/S + 1$

$H_2 = (H_1 - F + 2P)/S + 1$

$D_2 = K$

Example

Input: $5 \times 5 \times 3$

Filter $3 \times 3 \times 2$

Padding $P = 1$

Stride $S = 2$

$$W_1 = \frac{5 - 3 + 2 \times 1}{2} + 1 = 3 = H_1$$

$D_1 = 2$

➔ Output size: $3 \times 3 \times 2$

CNN: Convolutional Layer

Input Image

0	0	0	0	0	0	0
0	1	1	1	1	1	0
0	1	0	0	1	0	0
0	1	0	1	0	0	0
0	1	1	0	0	0	0
0	1	0	0	0	0	0
0	0	0	0	0	0	0

Learned Convolution

θ_{11}	θ_{12}	θ_{13}
θ_{21}	θ_{22}	θ_{23}
θ_{31}	θ_{32}	θ_{33}

Key Idea: The convolution kernel parameters are learned

Convolved Image

.4	.5	.5	.5	.4
.4	.2	.3	.6	.3
.5	.4	.4	.2	.1
.5	.6	.2	.1	0
.4	.3	.1	0	0

CNN Activation Function

- Real world data is inherently non-linear.
- For instance, the process of recognizing objects in an image involves understanding shapes, textures, and patterns that do not have a linear relationship with the pixel intensities.

Non-linear activation functions enable neural networks to capture these complex, non-linear mappings between inputs and outputs.

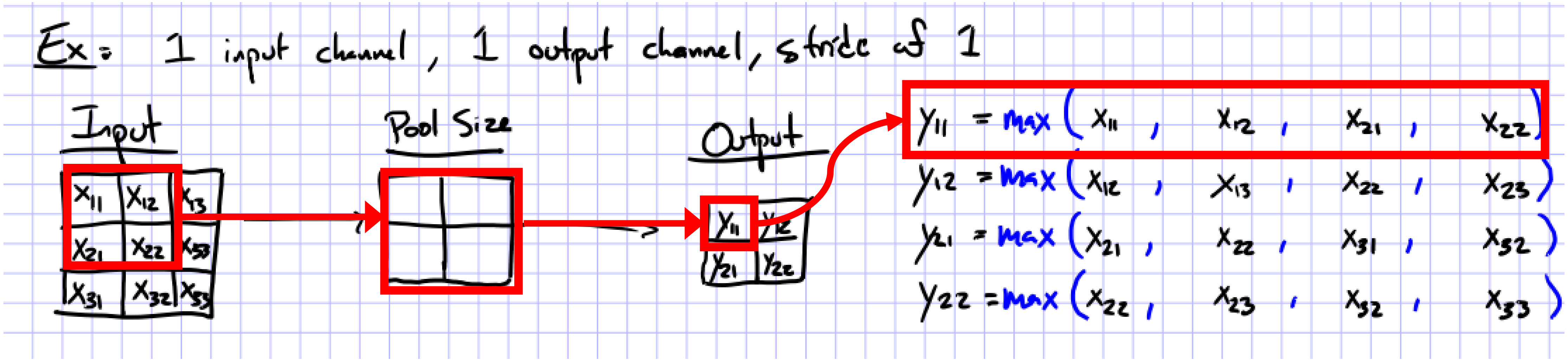
sigmoid	$\boldsymbol{1/(1+e^{-x})}$
Hyperbolic Tangent (tanh)	$\boldsymbol{(e^{x}-e^{-x})/(e^{x}+e^{-x})}$
Rectified Linear Unit (ReLU)	$\boldsymbol{max(0,x)}$
Leaky ReLU	$\begin{cases} x & if\ x>0 \\ \alpha x & otherwise \end{cases}$
Exponential Linear unit (ELU)	$\begin{cases} x & if\ x>0 \\ \alpha(e^{x}-1) & otherwise \end{cases}$
Softmax	$e^{x_j} \Big/ \sum_j e^{x_j}$

CNN Downsampling: Max-Pooling

Used to reduce the input size

Take the max value within the range of the kernel size

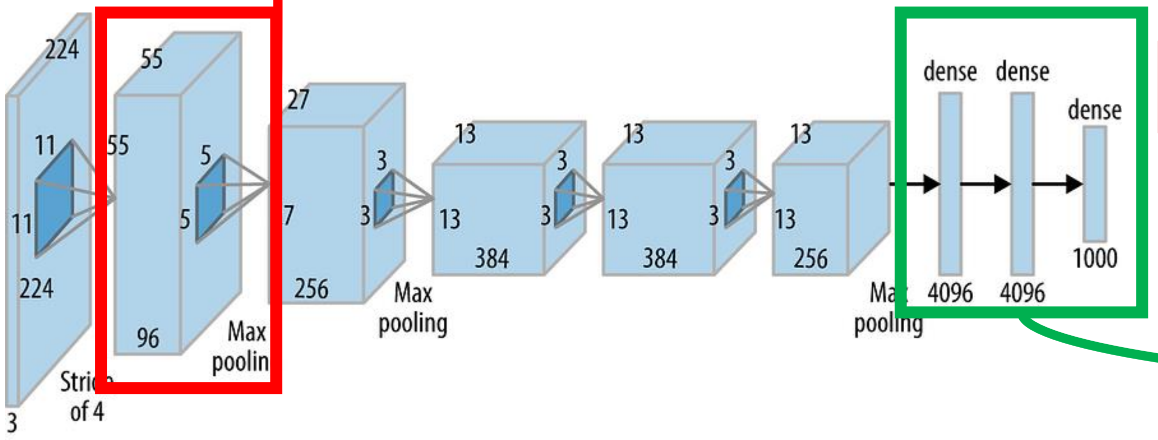
For Stride S = 2:



State-of-arts CNN

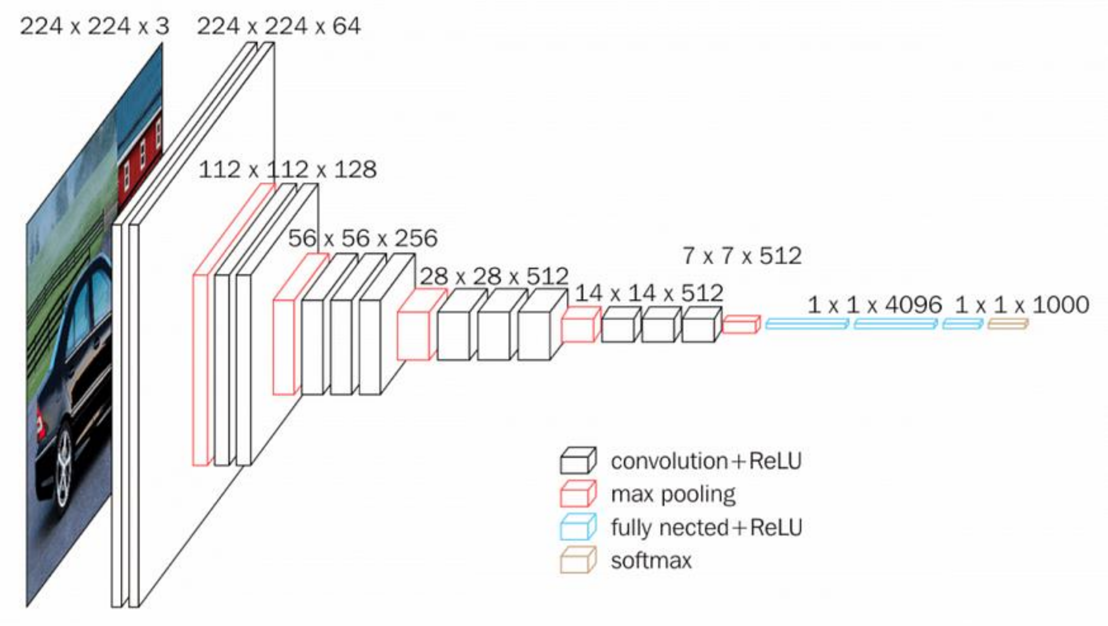
19

SOTA Deep Learning Models: AlexNet



AlexNet Network - Structural Details													
Input			Output			Layer	Stride	Pad	Kernel size		in	out	# of Param
227	227	3	55	55	96	conv1	4	0	11	11	3	96	34944
55	55	96	27	27	96	maxpool1	2	0	3	3	96	96	0
27	27	96	27	27	256	conv2	1	2	5	5	96	256	614656
27	27	256	13	13	256	maxpool2	2	0	3	3	256	256	0
13	13	256	13	13	384	conv3	1	1	3	3	256	384	885120
13	13	384	13	13	384	conv4	1	1	3	3	384	384	1327488
13	13	384	13	13	256	conv5	1	1	3	3	384	256	884992
13	13	256	6	6	256	maxpool5	2	0	3	3	256	256	0
						fc6			1	1	9216	4096	37752832
						fc7			1	1	4096	4096	16781312
						fc8			1	1	4096	1000	4097000
						Total							62,378,344

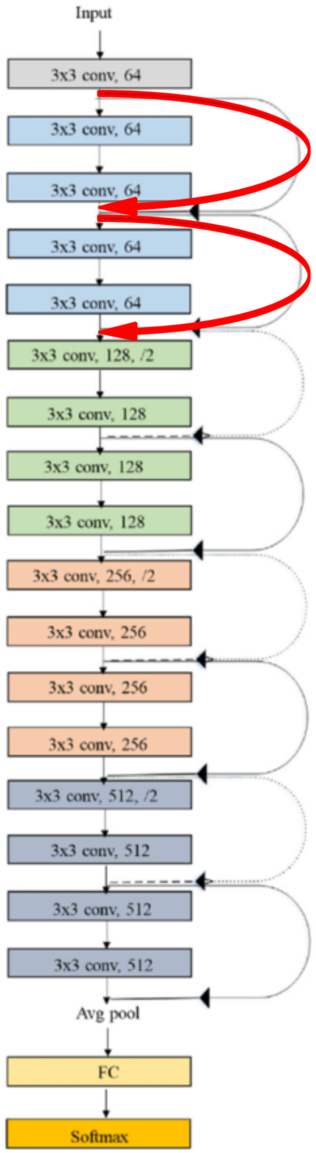
SOTA Deep Learning Models: VGG-16



VGG16 - Structural Details

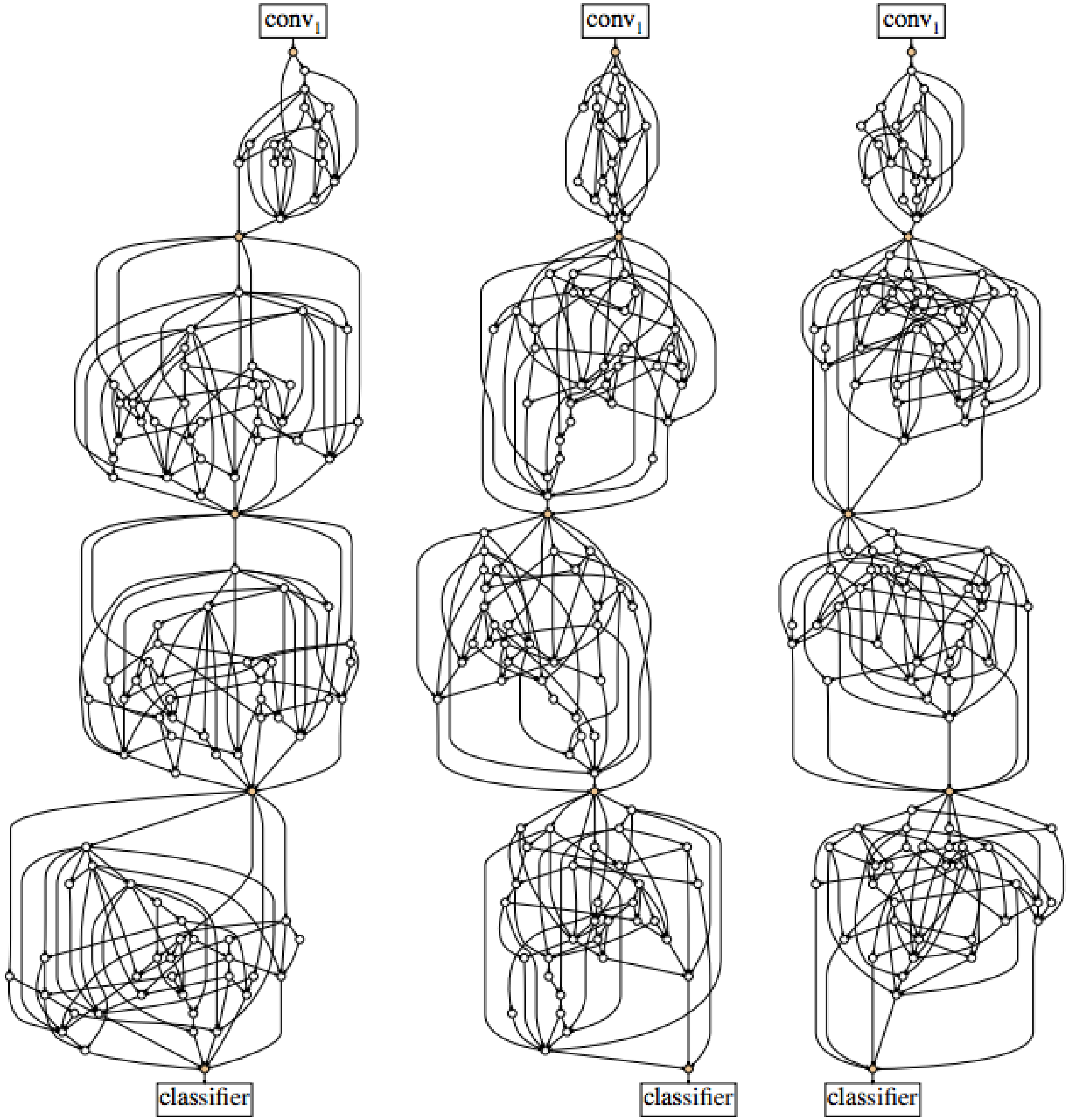
#	Input Image			output			Layer	Stride	Kernel		in	out	Param
1	224	224	3	224	224	64	conv3-64	1	3	3	3	64	1792
2	224	224	64	224	224	64	conv3064	1	3	3	64	64	36928
	224	224	64	112	112	64	maxpool	2	2	2	64	64	0
3	112	112	64	112	112	128	conv3-128	1	3	3	64	128	73856
4	112	112	128	112	112	128	conv3-128	1	3	3	128	128	147584
	112	112	128	56	56	128	maxpool	2	2	2	128	128	65664
5	56	56	128	56	56	256	conv3-256	1	3	3	128	256	295168
6	56	56	256	56	56	256	conv3-256	1	3	3	256	256	590080
7	56	56	256	56	56	256	conv3-256	1	3	3	256	256	590080
	56	56	256	28	28	256	maxpool	2	2	2	256	256	0
8	28	28	256	28	28	512	conv3-512	1	3	3	256	512	1180160
9	28	28	512	28	28	512	conv3-512	1	3	3	512	512	2359808
10	28	28	512	28	28	512	conv3-512	1	3	3	512	512	2359808
	28	28	512	14	14	512	maxpool	2	2	2	512	512	0
11	14	14	512	14	14	512	conv3-512	1	3	3	512	512	2359808
12	14	14	512	14	14	512	conv3-512	1	3	3	512	512	2359808
13	14	14	512	14	14	512	conv3-512	1	3	3	512	512	2359808
	14	14	512	7	7	512	maxpool	2	2	2	512	512	0
14	1	1	25088	1	1	4096	fc		1	1	25088	4096	102764544
15	1	1	4096	1	1	4096	fc		1	1	4096	4096	16781312
16	1	1	4096	1	1	1000	fc		1	1	4096	1000	4097000
							Total						138,423,208

SOTA Deep Learning Models: ResNet-18(34, 50, 101, 152 ..)



ResNet18 - Structural Details															
#	Input Image			output			Layer	Stride	Pad	Kernel		in	out	Param
1	227	227	3	112	112	64	conv1	2	1	7	7	3	64	9472
	112	112	64	56	56	64	maxpool	2	0.5	3	3	64	64	0
2	56	56	64	56	56	64	conv2-1	1	1	3	3	64	64	36928
3	56	56	64	56	56	64	conv2-2	1	1	3	3	64	64	36928
4	56	56	64	56	56	64	conv2-3	1	1	3	3	64	64	36928
5	56	56	64	56	56	64	conv2-4	1	1	3	3	64	64	36928
6	56	56	64	28	28	128	conv3-1	2	0.5	3	3	64	128	73856
7	28	28	128	28	28	128	conv3-2	1	1	3	3	128	128	147584
8	28	28	128	28	28	128	conv3-3	1	1	3	3	128	128	147584
9	28	28	128	28	28	128	conv3-4	1	1	3	3	128	128	147584
10	28	28	128	14	14	256	conv4-1	2	0.5	3	3	128	256	295168
11	14	14	256	14	14	256	conv4-2	1	1	3	3	256	256	590080
12	14	14	256	14	14	256	conv4-3	1	1	3	3	256	256	590080
13	14	14	256	14	14	256	conv4-4	1	1	3	3	256	256	590080
14	14	14	256	7	7	512	conv5-1	2	0.5	3	3	256	512	1180160
15	7	7	512	7	7	512	conv5-2	1	1	3	3	512	512	2359808
16	7	7	512	7	7	512	conv5-3	1	1	3	3	512	512	2359808
17	7	7	512	7	7	512	conv5-4	1	1	3	3	512	512	2359808
	7	7	512	1	1	512	avg pool	7	0	7	7	512	512	0
18	1	1	512	1	1	1000	fc					512	1000	513000
Total														11,511,784

SOTA Deep Learning Models: Neural Architecture Search



Recurrent Neural Networks (RNN)

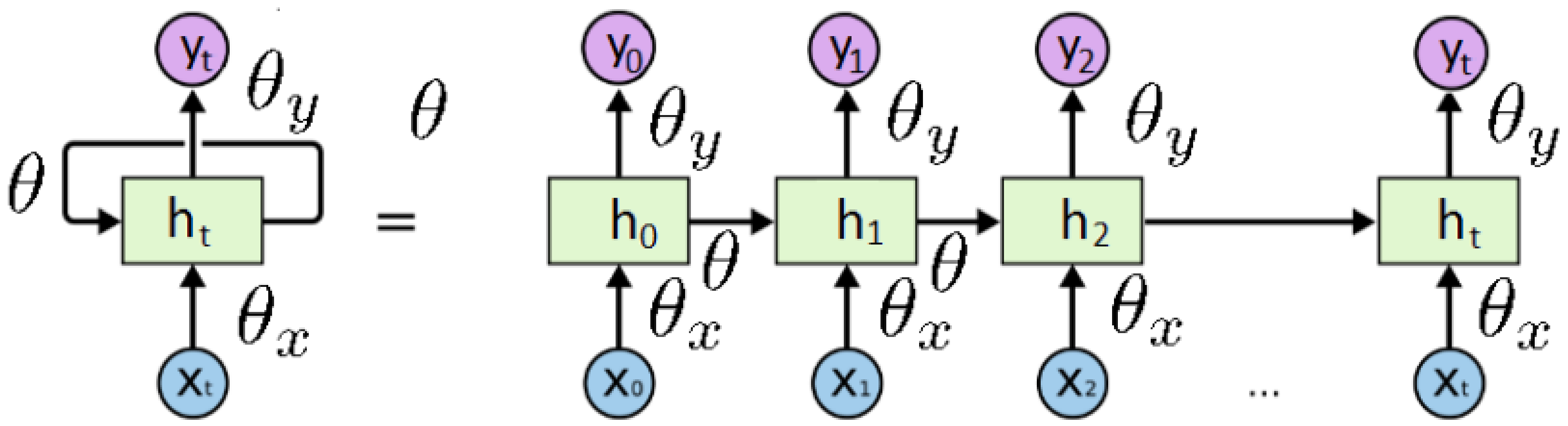
24

Recurrent Neural Networks (RNN)

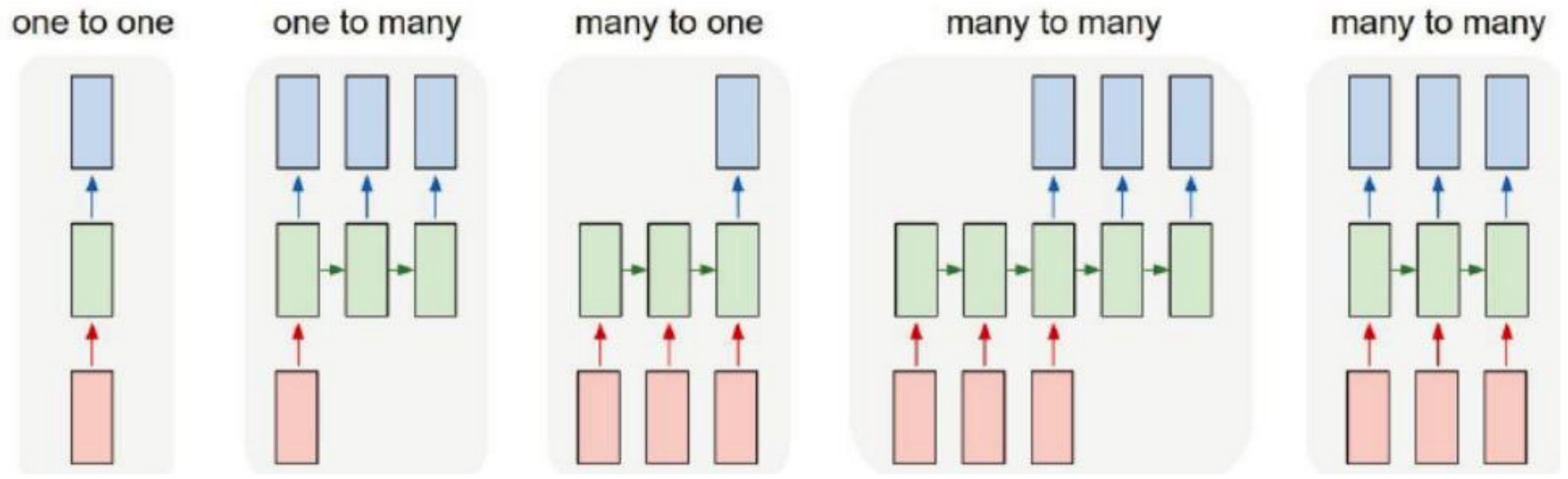
- Motivation:
  - Some data are sequential e.g., time series, text, speech
  - The sequence matters i.e., knowledge about the previous information matters:
  - The clouds are in the …?
    - Sky
  - Issues with simple NN for sequential data:
    - Fixed input/Fixed Output
    - Hard/Impossible to choose a fixed context window

Recurrent Neural Networks (RNN)

- RNNs can be thought of as multiple copies of the same network, each passing a message to a successor.
- The same function and the same set of parameters are used at every time step.
- Are called recurrent because they perform the same task for each input.



Recurrent Neural Networks (RNN)



Vanilla NN: Fixed I/O i.e. image classification	Sequence Output e.g., image captioning: image to sequence of words	Sequence Input e.g., sentiment classification: sequence of words to sentiment	Sequence I/O e.g., machine translation: sequence of words to sequence of words	Synced sequence I/O e.g. video frame classification

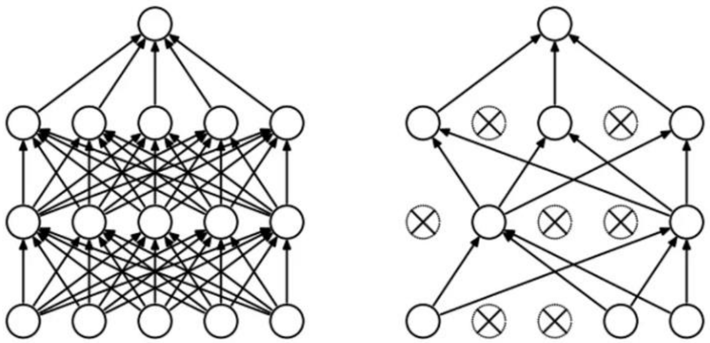
Regularizing Neural Networks

Regularizing deep neural networks

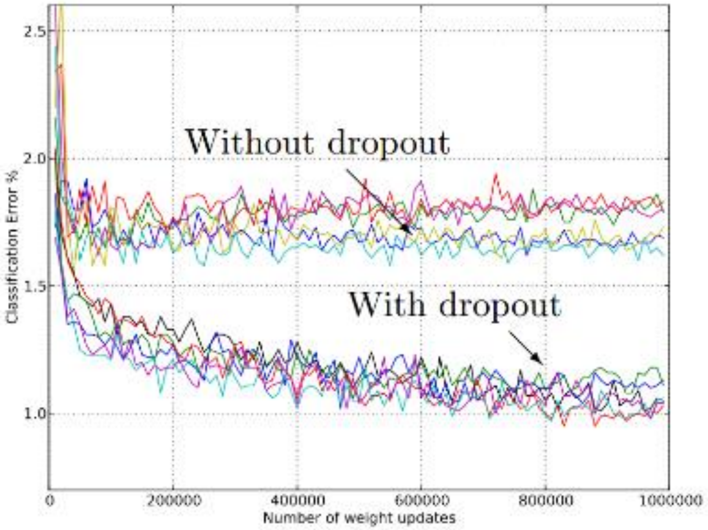
Why?

- Avoid Overfitting
- Noisy data
- Reduce model complexity

Dropout



(a) Standard Neural Net (b) After applying dropout.



Batch Normalization

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1 \ldots m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

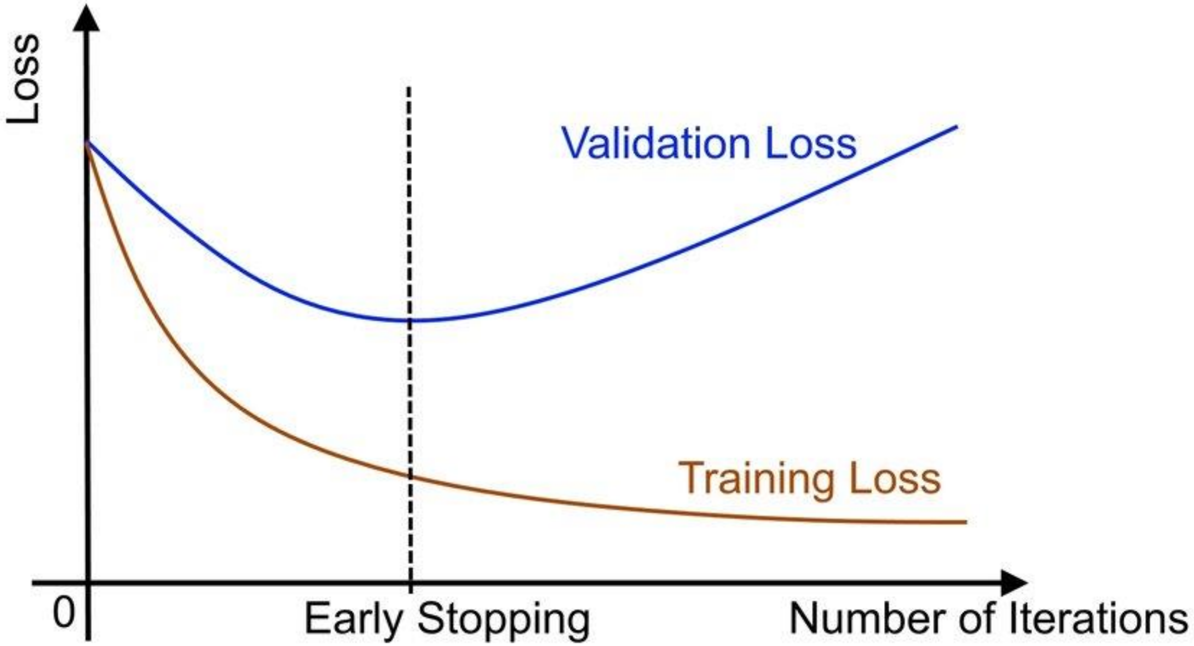
$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

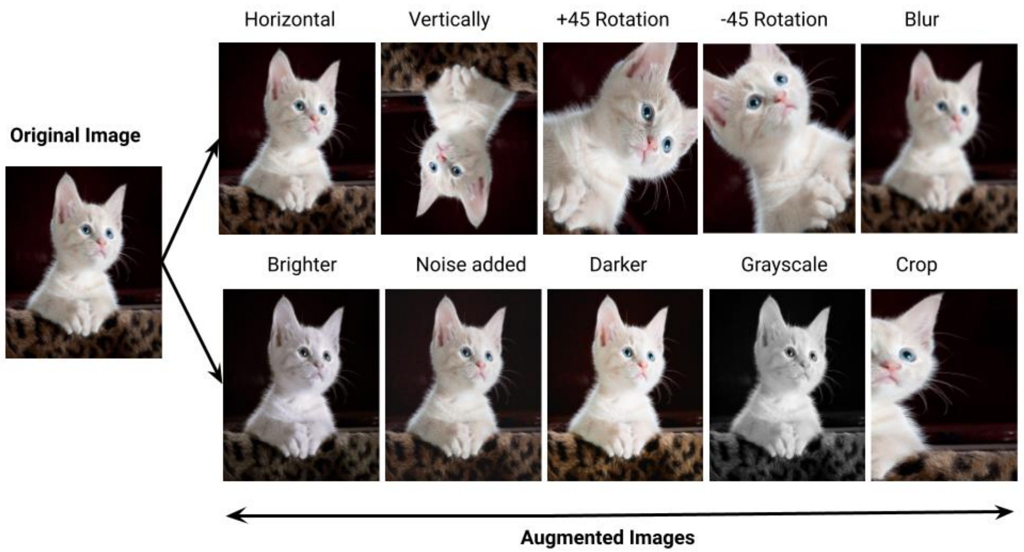
$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Regularizing deep neural networks

Early Stopping



Data Augmentation



Demo

Demo

Fine Tuning

33

Fine Tuning

Motivation:

- Goal: Recognize chair types from images and recommend purchase links.
- Method: Identify 100 common chairs, take 1000 images per chair, train classification model.
- Dataset smaller than ImageNet, risking overfitting of complex models.
- Limited training examples may result in insufficient accuracy for practical use.

➔ Let’s collect more data ➔ Collecting and labeling data can take a lot of time and money.

Or:

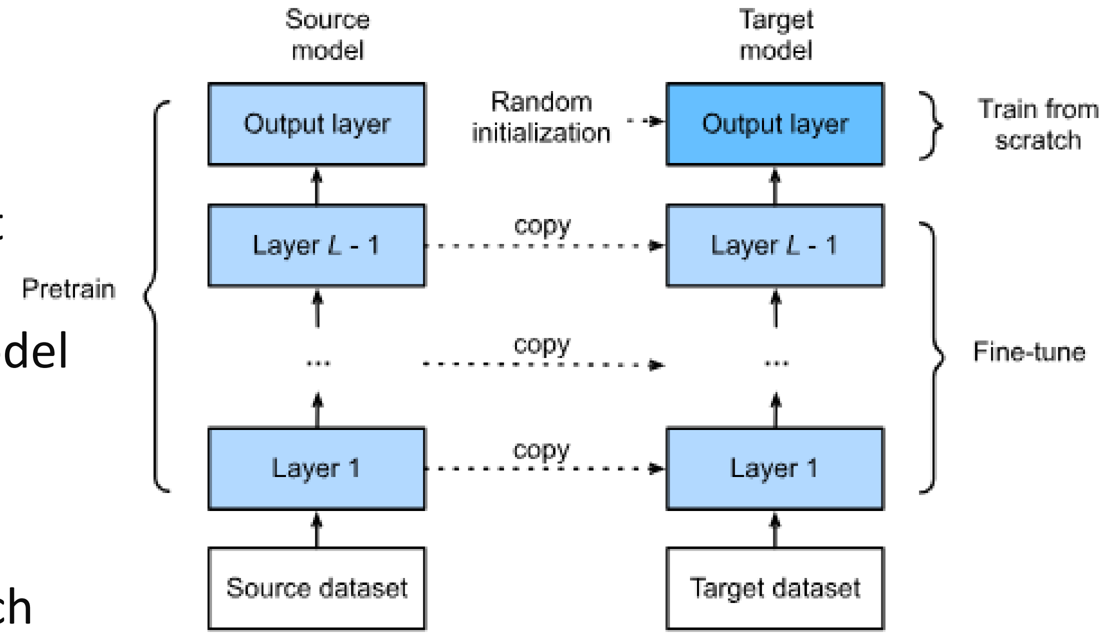
➔ Apply knowledge from source dataset to target dataset.

➔ ImageNet features (edges, textures, shapes, object composition) can help recognize chairs.

➔ Leverages pre-trained models (models already trained) to improve performance on chair recognition task.

Fine Tuning- How?

- Pre-train source model on source dataset (e.g., ImageNet)
- Create target model, copying source model except output layer
- Add randomly initialized output layer to target model based on target dataset categories (how many classes you have)
- Train target model on target dataset, fine-tuning copied layers and training output layer from scratch




```
from torchvision.models import resnet50, ResNet50_Weights

# Old weights with accuracy 76.130%
resnet50(weights=ResNet50_Weights.IMAGENET1K_V1)

# New weights with accuracy 80.858%
resnet50(weights=ResNet50_Weights.IMAGENET1K_V2)

# Best available weights (currently alias for IMAGENET1K_V2)
# Note that these weights may change across versions
resnet50(weights=ResNet50_Weights.DEFAULT)

# Strings are also supported
resnet50(weights="IMAGENET1K_V2")

# No weights - random initialization
resnet50(weights=None)
```

Conclusion: Things to keep in mind

- Classic ML algorithms work better for tabular data (so far)

- Training Deep NN:
    - Data augmentation
    - Dropout
    - Monitor your loss
    - Early stopping

- We didn't even scratch the surface

جامعة قطر
QATAR UNIVERSITY

كلية الهندسة
COLLEGE OF ENGINEERING

Thank you