

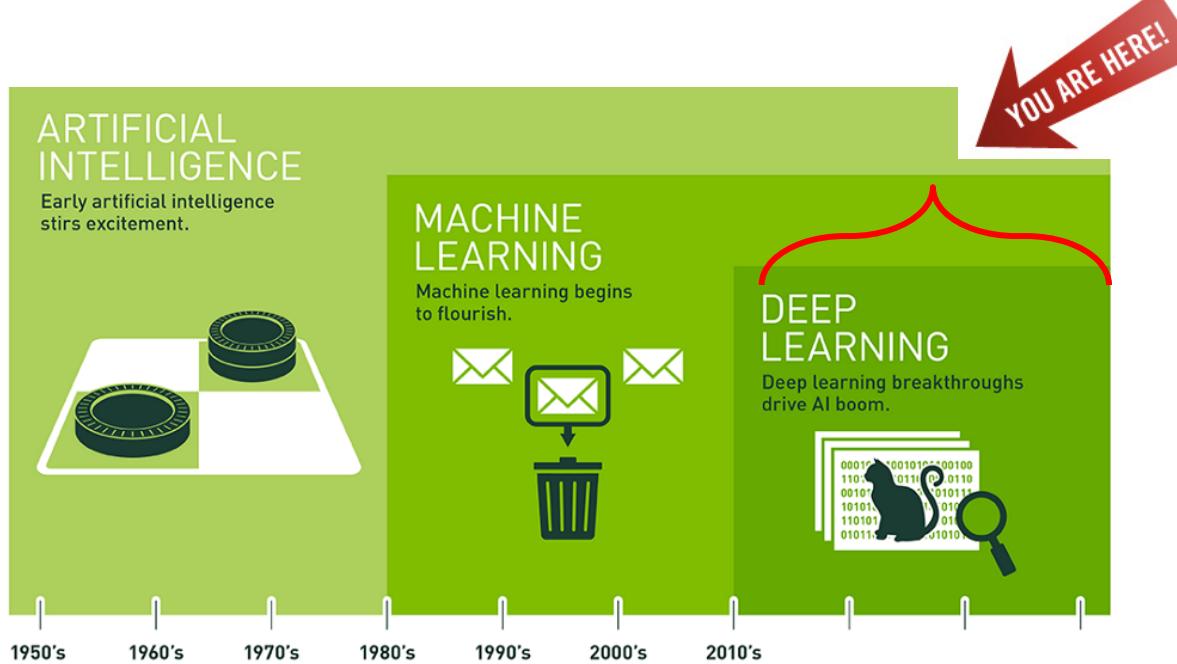
# Artificial Intelligence: Deep Learning, CNNs

many slides from: Y. Bengio, A. Ng and Y. LeCun

# Today

- 
1. Motivation
  2. Feature Learning
  3. Early Training of Deep Neural Networks
  4. CNNs for Image Processing
  5. Conclusion

# History of AI



# Deep Learning in the Academic Press (2012-2015)

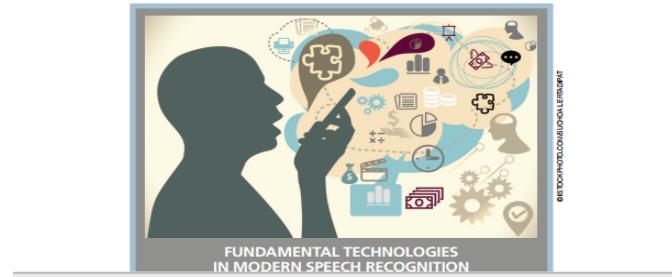
The image shows a screenshot of a Nature journal article. The header reads "nature International weekly journal of science". The main title is "Deep learning" by "Yann LeCun, Yoshua Bengio & Geoffrey Hinton". Below the title are "Affiliations | Corresponding author" links. The publication details are "Nature 521, 436–444 (28 May 2015) | doi:10.1038/nature14539" and "Received 25 February 2015 | Accepted 01 May 2015 | Published online 27 May 2015".

Geoffrey Hinton, Li Deng, Dong Yu, George E. Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N. Sainath, and Brian Kingsbury



## Deep Neural Networks for Acoustic Modeling in Speech Recognition

The shared views of four research groups



# Deep Learning in the News (2013)



## 10 BREAKTHROUGH TECHNOLOGIES 2013

Introduction

The 10 Technologies

Past Years

### Deep Learning

With massive amounts of computational power, machines can now recognize objects and translate speech in real time. Artificial intelligence is finally getting smart.

### Temporary Social Media

Messages that quickly self-destruct could enhance the privacy of online communications and make people freer to be spontaneous.

### Prenatal DNA Sequencing

Reading the DNA of fetuses will be the next frontier of the genomic revolution. But do you really want to know about the genetic problems or musical aptitude of your unborn child?

### Additive Manufacturing

Skeptical about 3-D printing? GE, the world's largest manufacturer, is on the verge of using the technology to make jet parts.

### Baxter: The Blue-Collar Robot

Rodney Brooks's newest creation is easy to interact with, but the complex innovations behind the robot show just how hard it is to get along with people.

### Memory Implants

A maverick neuroscientist believes he has deciphered the code by which the brain forms long-term memories. Next: testing a prosthetic implant for people suffering from long-term memory loss.

### Smart Watches

The designers of the Pebble watch realized that a mobile phone is more useful if you don't have to take it out of your pocket.

### Ultra-Efficient Solar Power

Doubling the efficiency of a solar cell would completely change the economics of renewable energy. Nanotechnology just might make it possible.

### Big Data from Cheap Phones

Collecting and analyzing information from simple cell phones can provide surprising insights into how people move about and behave – and even help us understand the spread of diseases.

### Supergrids

A new high-power circuit breaker could finally make highly efficient DC power grids practical.

# Deep Learning in the News (2012-2014)

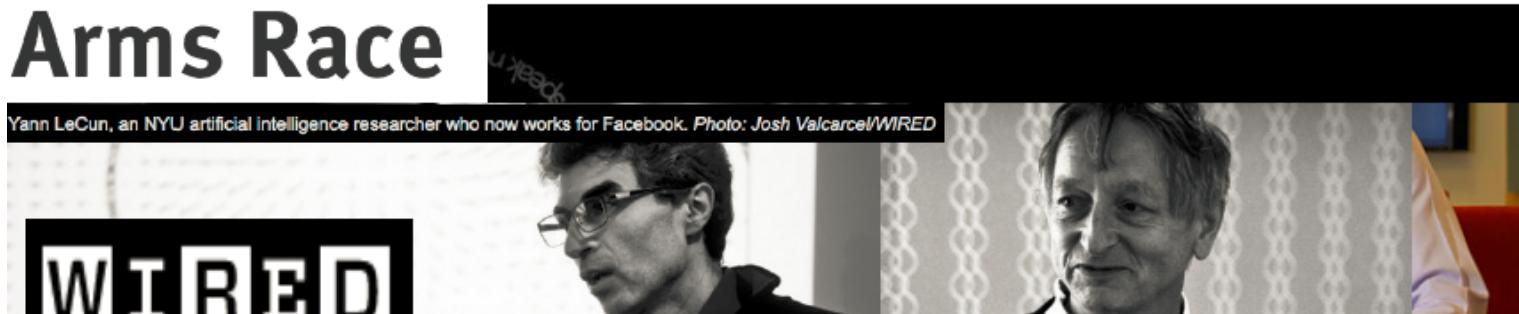


EXCLUSIVE



## Facebook, Google in 'Deep Learning' Arms Race

Yann LeCun, an NYU artificial intelligence researcher who now works for Facebook. Photo: Josh Valcarcel/WIRED



## Google Beat Facebook for DeepMind Google Acquires Artificial Intelligence Startup DeepMind For More Than \$500M

Posted Jan 26, 2014 by Catherine Shu (@catherineshu)

# Major Breakthroughs

## ■ Speech Recognition & Machine Translation (2010+)

Skype to get 'real-time' translator



Analysts say the translation feature could have wide ranging applications

*Skype Translator*

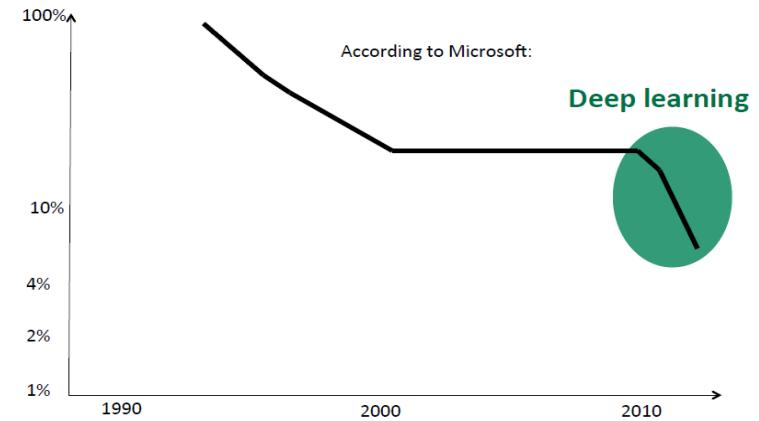


*Google Translate*



*Google now*

2010-2012: Breakthrough in speech recognition → in Androids by 2012



- Image Recognition & Computer Vision (2012+)
- Natural Language Processing (2014+)
- ...

# Major Breakthroughs

- Speech Recognition & Machine Translation (2010+)
- **Image Recognition & Computer Vision (2012+)**



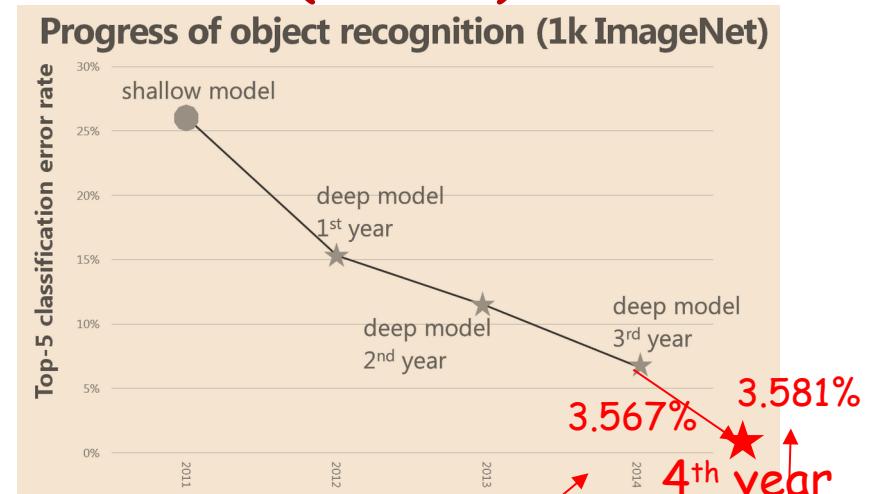
*Object recognition*



*Self driving cars*



Google Photos



- Natural Language Processing (2014+)
- ...

# Major Breakthroughs

- Speech Recognition & Machine Translation (2010+)
- Image Recognition & Computer Vision (2012+)
- **Natural Language Processing (2014+)**

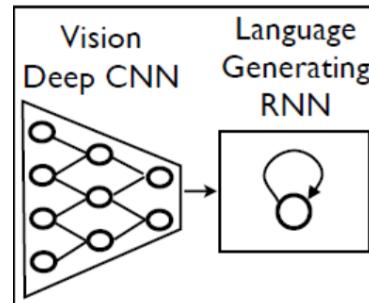
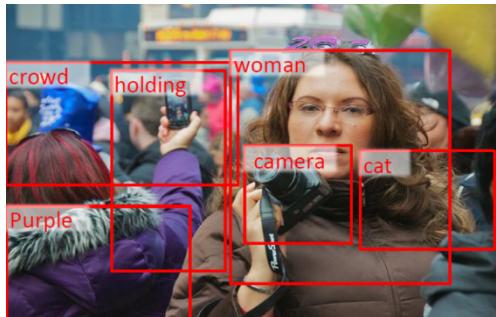
Joe went to the kitchen. Fred went to the kitchen. Joe picked up the milk.  
Joe travelled to the office. Joe left the milk. Joe went to the bathroom.

Where is the milk now? **A: office**

Where is Joe? **A: bathroom**

Where was Joe before the office? **A: kitchen** [http://blog.csdn.net/qfnu\\_cjt\\_wl](http://blog.csdn.net/qfnu_cjt_wl)

*Question Answering*



*Image Captioning (deep vision + deep NLP)*

Machine-generated (but turker preferred)

a bicycle is parked next to a river

Human-annotated (but turker not preferred)

a bike sits parked next to a body of water

# Image Captioning: Better than humans?



A

a woman in a kitchen preparing food

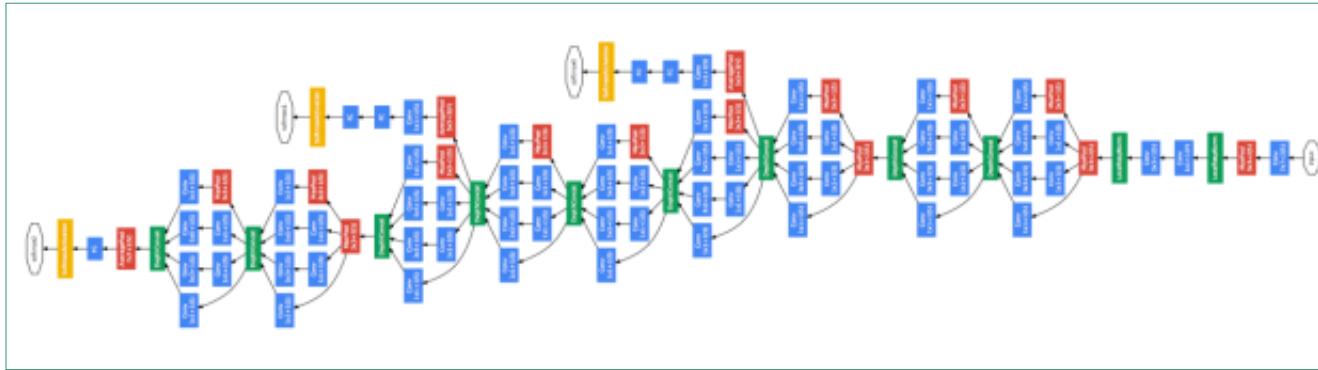
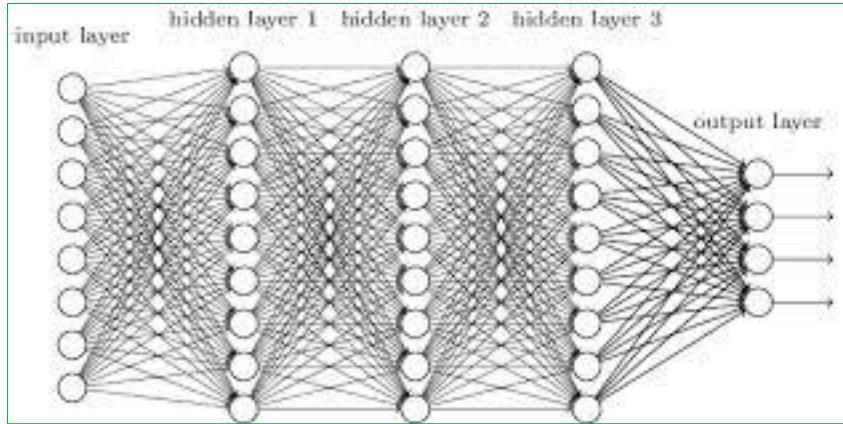
B

woman working on counter near kitchen sink preparing a meal

# Today

1. Motivation
2. Feature Learning 
3. Early Training of Deep Neural Networks
4. CNNs for Image Processing
5. Deep Learning for NLP
6. Conclusion

# A Deep Neural Net



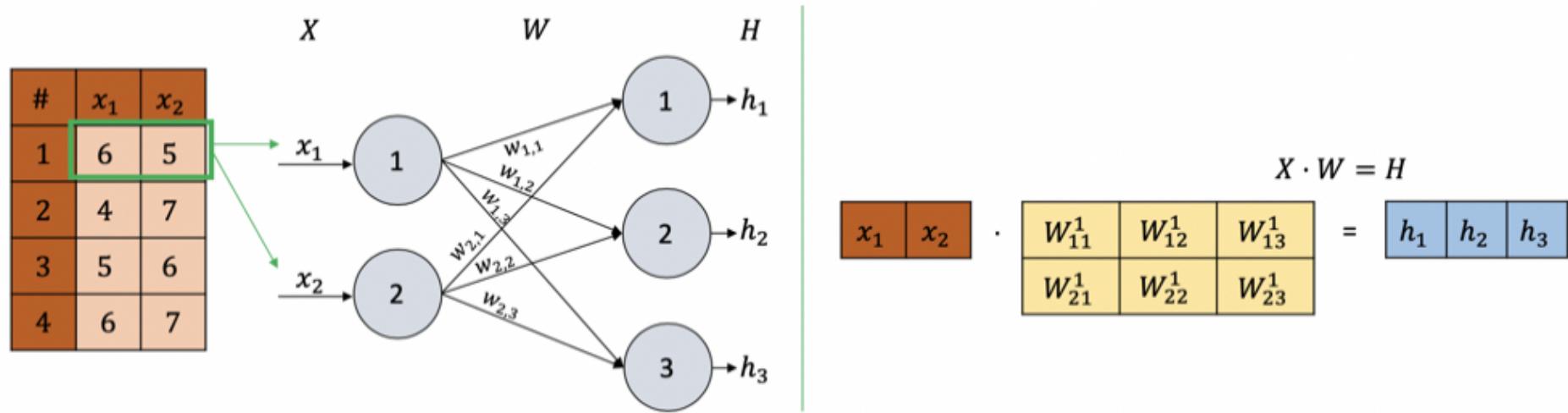
The Google "Inception" deep neural network architecture for image recognition  
(27 layers)

<https://gph.ec.quoracdn.net/main-qimg-970d2b5f57b6b5cd13dc11f5371166b2-c>

<https://devblogs.nvidia.com/parallelforall/mocha-jl-deep-learning-julia/>

# Matrix Notation

We now represent inputs  $x$ , weights  $W$  and the bias weights  $b$  as matrices:



Input to hidden:

1. Compute net activation  $\text{net}_h = x \cdot W_{ih} + b_{ih}$
2. Compute activation function (e.g., sigmod):  $h = S(\text{net}_h)$

Hidden to output:

$$o = S(h \cdot W_{ho} + b_{ho})$$

→ Worksheet #6 ("Matrix Notation")

# Initial Drawbacks

1. Standard backpropagation with sigmoid activation function does not scale well with multiple layers
  - Weight of early layers change too slowly (no learning)
2. Overfitting
  - Large network -> lots of parameters -> increased capacity to "learn by heart"
3. Multilayered ANNs need lots of labeled data
  - Most data is not labeled :(

# Initial Drawbacks (1)

1. Standard gradient-based backpropagation does not scale well with multiple layers...

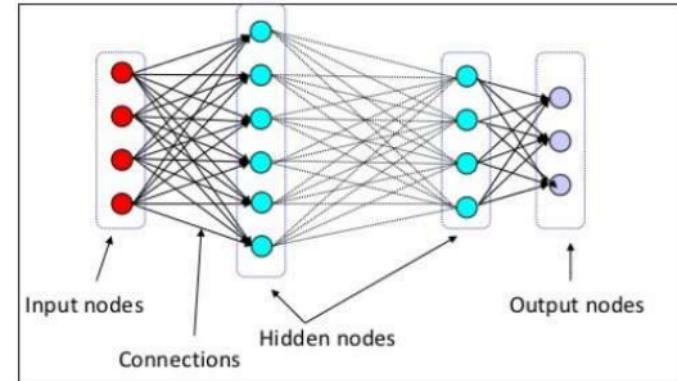
When we multiply the gradients many times (for each layer), it can lead to ...

- a) Vanishing gradient problem:

- ❑ gradients shrink exponentially with the number of layers
- ❑ so weight updates get smaller and smaller
- ❑ and weights of early layers change very slowly and network learns very very slowly

- b) Exploding gradient problem:

- ❑ multiplying gradients could also make them grow exponentially.
- ❑ so weight updates get larger and larger
- ❑ and the weights can become so large as to overflow and result in NaN values



$$\begin{aligned}\delta_h &= g'(x_h) \times Err_h \\ &= O_h (1 - O_h) \times \sum_k (w_{hk} \delta_k)\end{aligned}$$

$$\delta_6 = O_6 (1 - O_6) \times \sum (w_{6,7} \delta_7)$$

$$\delta_5 = O_5 (1 - O_5) \times \sum (w_{5,6} \delta_6)$$

$$\delta_4 = O_4 (1 - O_4) \times \sum (w_{4,5} \delta_5)$$

$$\delta_3 = O_3 (1 - O_3) \times \sum (w_{3,4} \delta_4)$$

...

# Initial Drawbacks (1)

To help, we can :

- a) Use other activation functions...
- b) Do "gradient clipping" (i.e. set bounds on the gradients)

Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$
Rectified Linear Unit (ReLU) [2]		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parametric Rectified Linear Unit (PReLU) [2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU) [3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$

# Initial Drawbacks (2)

## 2. Overfitting

- Large network → lots of parameters → increased capacity to "learn by heart"

- *Solutions:*

- *Regularization:*

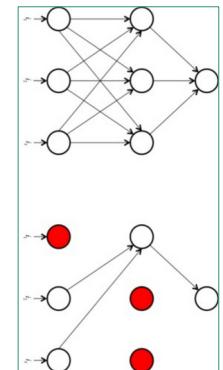
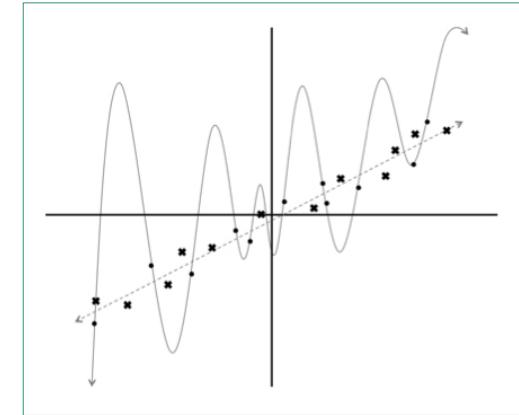
- modify the error function that we minimize to penalize large weights.

$$\frac{1}{2} \sum_{i=0}^n \frac{(T_i - O_i)^2}{n} + \lambda f(w)$$

- where  $f(w)$  grows larger as the weights grow larger and  $\lambda$  is the regularization strength

- *Dropout:*

- keep a neuron active with some probability  $p$  or setting it to zero otherwise.
  - prevents the network from becoming too dependent on any one neuron.



# Initial Drawbacks (3)

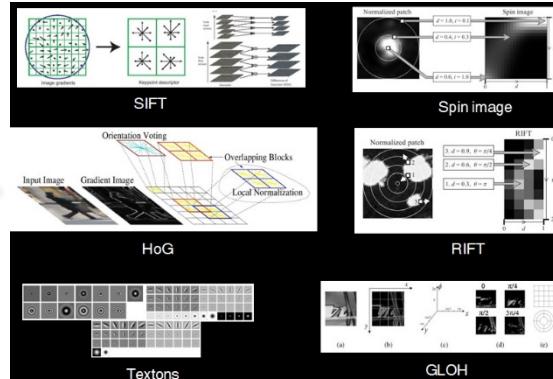
## 3. Multilayered ANNs need lots of labeled data

- Solution: “pre-train” the network with features found automatically using unsupervised data
- i.e. Automatic feature learning... (see next few slides)

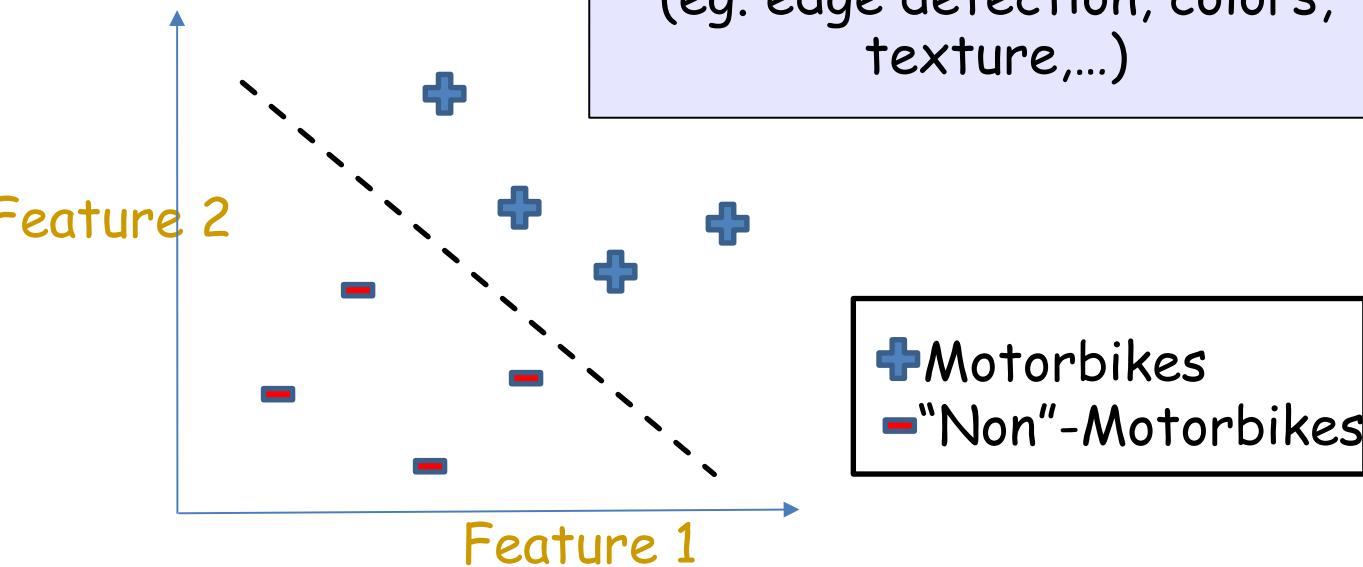
# Classic ML



Input



Learning algorithm

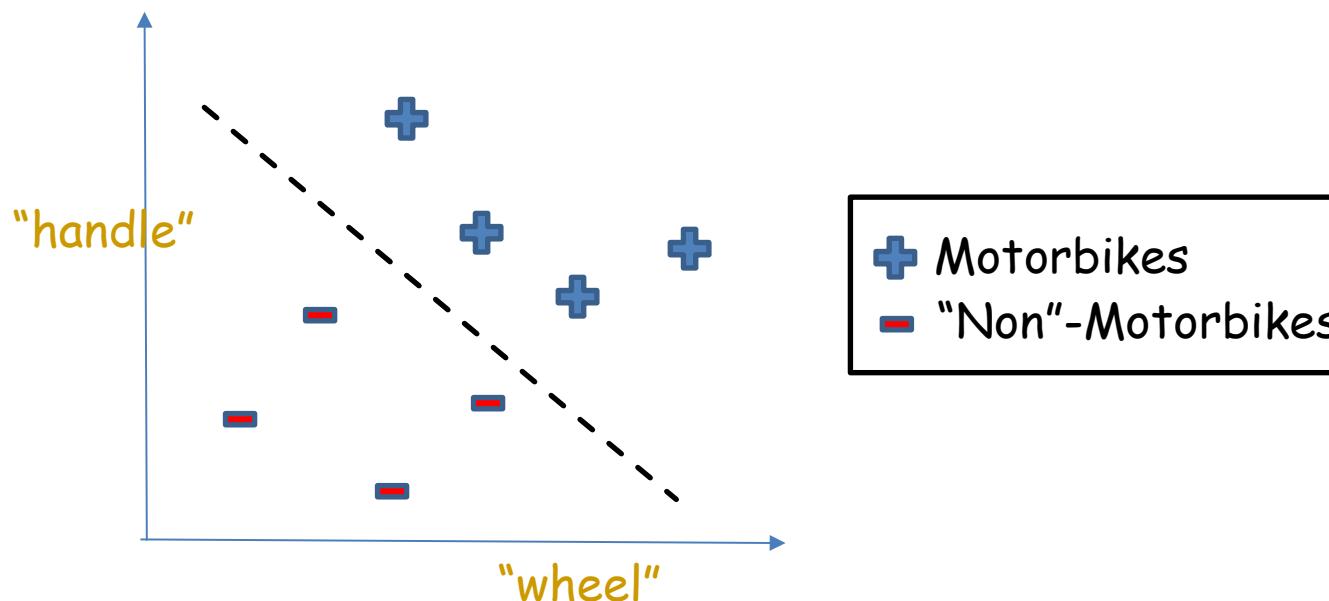
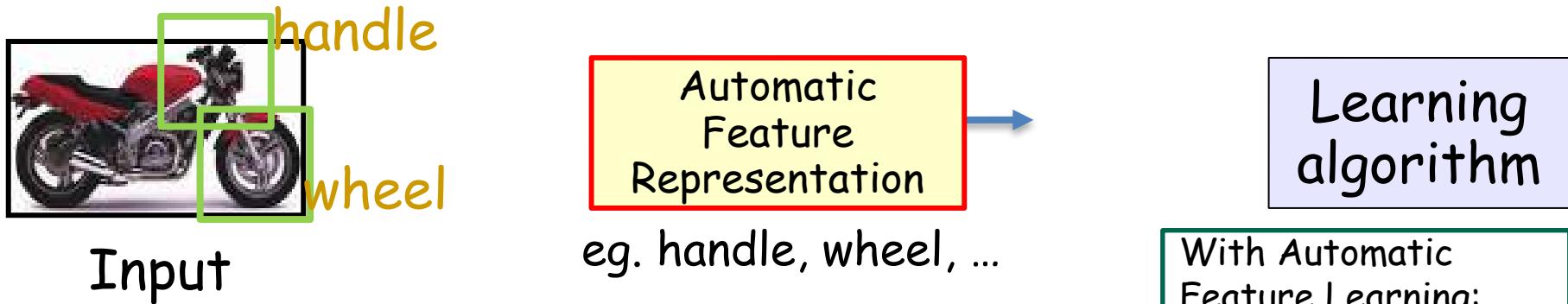


+ Motorbikes  
- "Non"-Motorbikes

Classic ML,  
requires labeled  
data and hand-  
crafted features

1. Needs expert knowledge
2. Time-consuming and expensive
3. Does not generalize to other domains

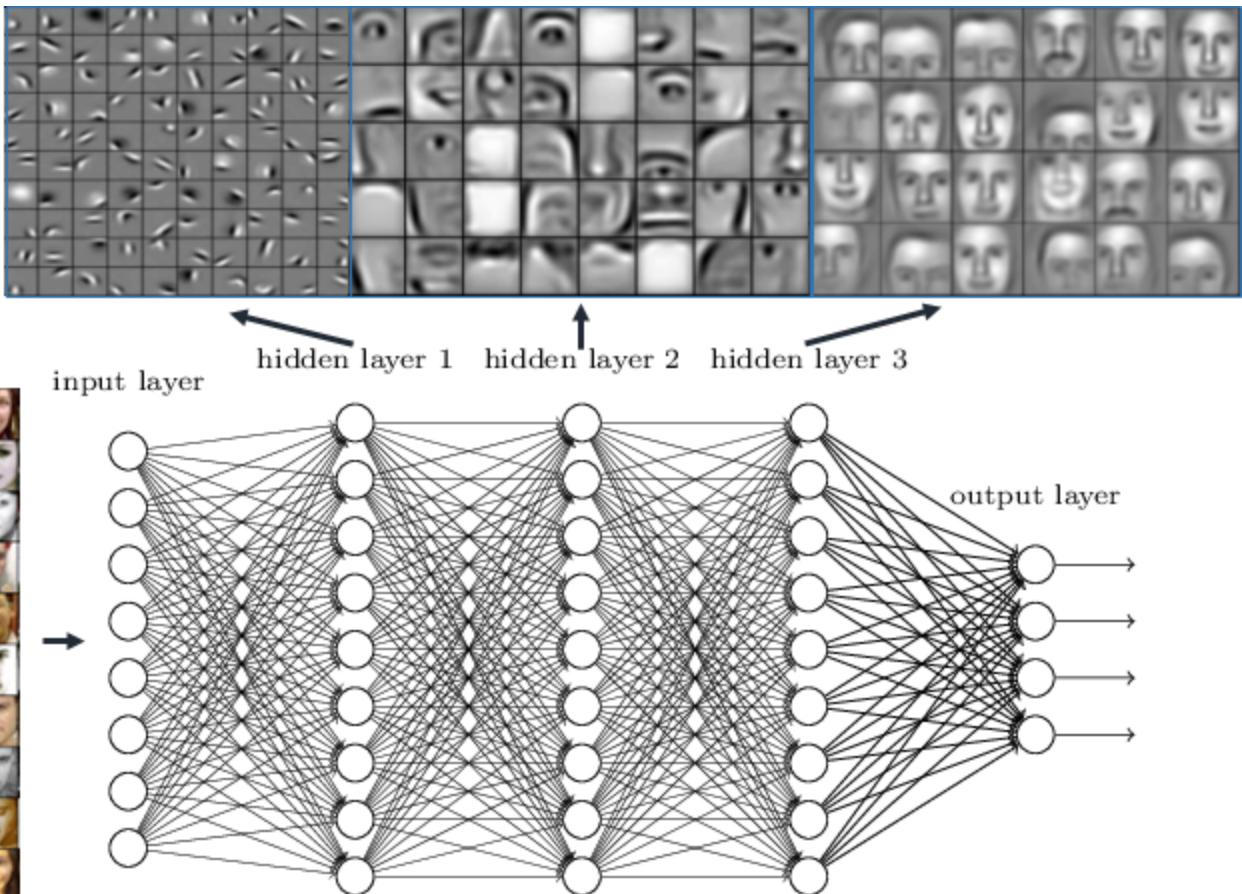
# Automatic Feature Learning



- With Automatic Feature Learning:
1. We feed the network the raw data (not feature-curated)
  2. The features are learned by the network
  3. Features learned can be re-used in similar tasks.

# Automatic Feature Learning

Deep neural networks learn hierarchical feature representations



# Automatic Feature Learning

*Deep Learning = Machine learning algorithms based on learning multiple levels of representation / abstraction.*

- Y. Bengio

- Each layer learns more abstract features that are then combined / composed into higher-level features automatically
- Like the human brain ...
  - has many layers of neurons which act as feature detectors
  - detecting more and more abstract features as you go up
- E.g. to classify an image of a cat:
  - Bottom Layers: Edge detectors, curves, corners straight lines
  - Middle Layers: Fur patterns, eyes, ears
  - Higher Layers: Body, head, legs
  - Top Layer: Cat or Dog



# Automatic Feature Learning

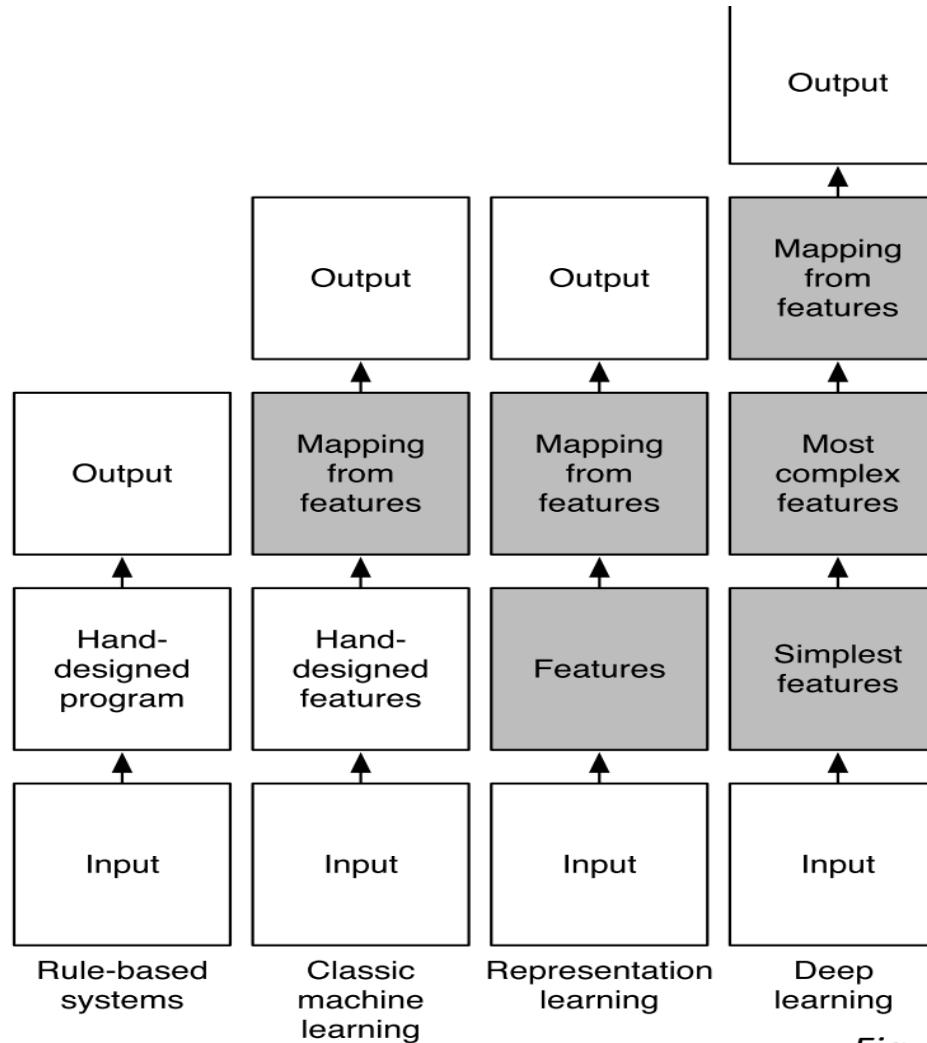
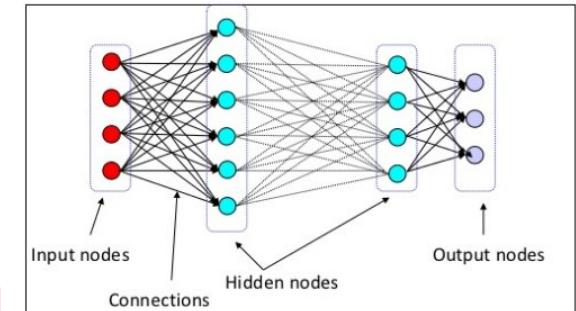
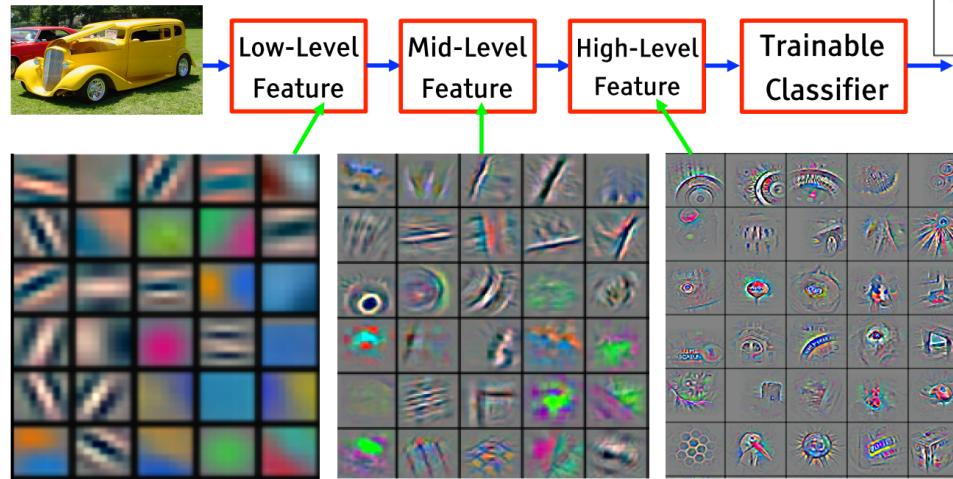


Fig: I. Goodfellow

# What Types of Features?

- For image recognition
  - pixel → edge → texton → motif → part → object

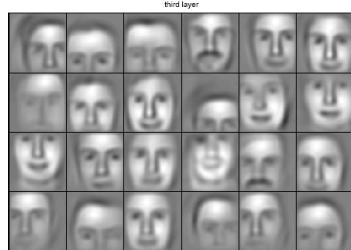


- For NLP
  - character → word → constituents → clause → sentence → discourse
- For speech:
  - sample → spectral band → sound → ... phone → phoneme → word

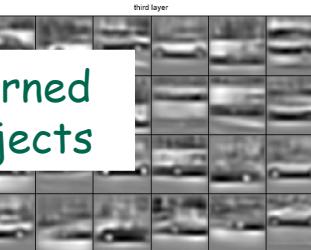
# Eg: Learning Image Features

Examples of learned objects parts from object categories

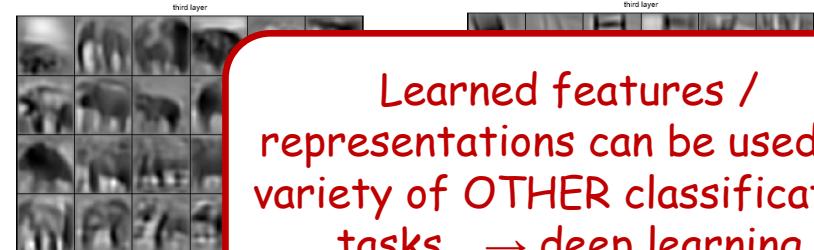
Faces



Cars



Elephants



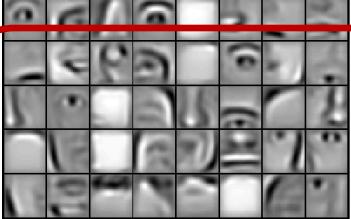
Chairs



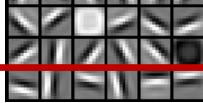
Learned  
objects



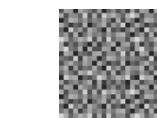
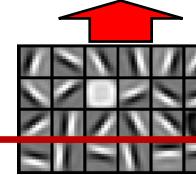
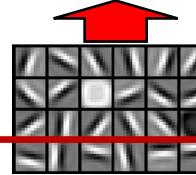
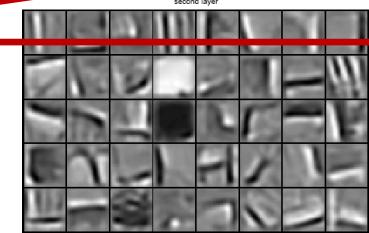
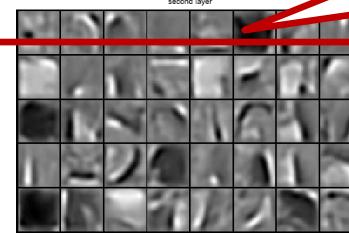
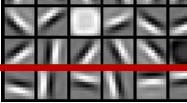
Learned  
object  
parts



Learned  
edges

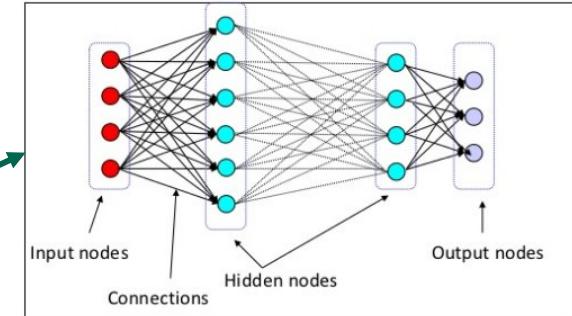


Actual images  
(pixels)



Learned features /  
representations can be used in  
variety of OTHER classification  
tasks... → deep learning

# Advantages of Unsupervised Feature Learning



SPEECH LANGUAGE THERAPY IN THE EDUCATIONAL CONTEXT  
By: Victor Antonio Pardo Rodríguez\*  
Master in Speech Language Pathology and Audiology  
Graduate of the Universidad Nacional de Colombia.  
Email: v.pardo@unal.edu.co  
<http://www.venezolanaingenieros.com>

INTRODUCTION  
Recently, after the Cruttenden (1983) began that "the approach of speech-language pathologists towards knowledge and control of the articulation and representation of the language is mainly based on the study of the individual in communication. That is to say, it is addressed that address each child in a specific way".

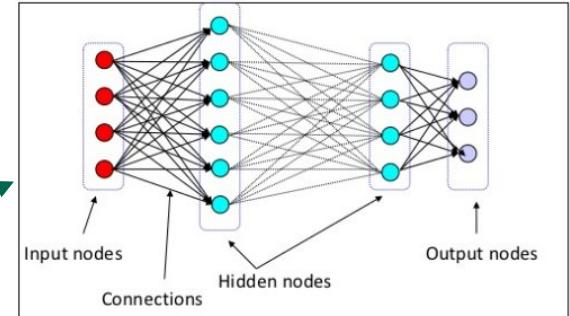
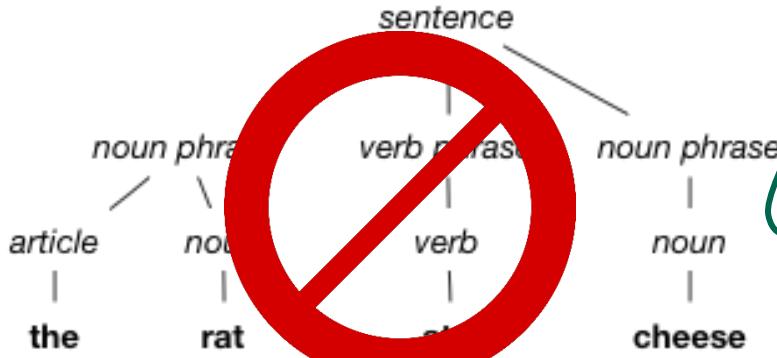
For Briscoe (1988, p. 27), many children have demonstrated socioemotional difficulties, and therefore, the speech-language pathologist must take into account the social context in which the child lives, as well as the family environment, the school environment, the neighborhood, the community, etc., in addition to the individual environment. Thus, in other words, address each child in a specific way, that is to say, it is addressed that address each child in a specific way. This is because the speech-language pathologist must take into account the individual and their environment in the interpretation of results of the interaction.

Rivas et al. (2004, p. 44 and 45) argue that the quality is understood in this context, and that it is the result of the interaction between the child and the environment in which they live, which is to say, the interaction of providing quality education to all in a safe and healthy environment, so that the child can develop his/her potentialities and abilities. In this case, the quality education was measured in different socio groups and areas to measure, we addressed that address each child in a specific way, that is to say, it is addressed that address each child in a specific way. So when talking about quality of education, one should take into account the individual and their environment in the interpretation of results of the interaction.

Castellón (1995), cited by Bustamante Jr. and Gómez (2009, p. 15) says that the notion of communication is a complex concept, but it is also a basic concept, since the social relationship is a communication system. And the latter (2009, p. 16) indicates that the communication is a process of exchange of information between two or more agents. Teaching practices would be given by the relation of instructional situation and communication, that is to say, the communication is a process of exchange of information between two or more agents.

This indicates that a program for learning reading, writing and other academic content must be given by the relation of instructional situation and communication, that is to say, the communication is a process of exchange of information between two or more agents.

\*Corresponding author. E-mail: v.pardo@unal.edu.co  
The authors declare that they have no conflicts of interest with respect to their present work.



# Advantages of Unsupervised Feature Learning

- Much more unlabeled data available than labeled data:
  - Eg. Websites, Books, Videos, Pictures
- Humans learn initially from unlabeled examples
  - Eg. Babies learn to talk/recognize objects without labeled data
- As the features are learned in an unsupervised way from a different and larger dataset, less risk of over-fitting
- **No need for manual feature engineering**
- These features are organized into multiple levels
  - Each level creates new features from combinations of features from the level below
  - Each level is more abstract than the ones below (hierarchy of features)

# Today

1. Motivation
2. Feature Learning
3. Early Training of Deep Neural Networks
4. CNNs for Image Processing
5. Deep Learning for NLP
6. Conclusion

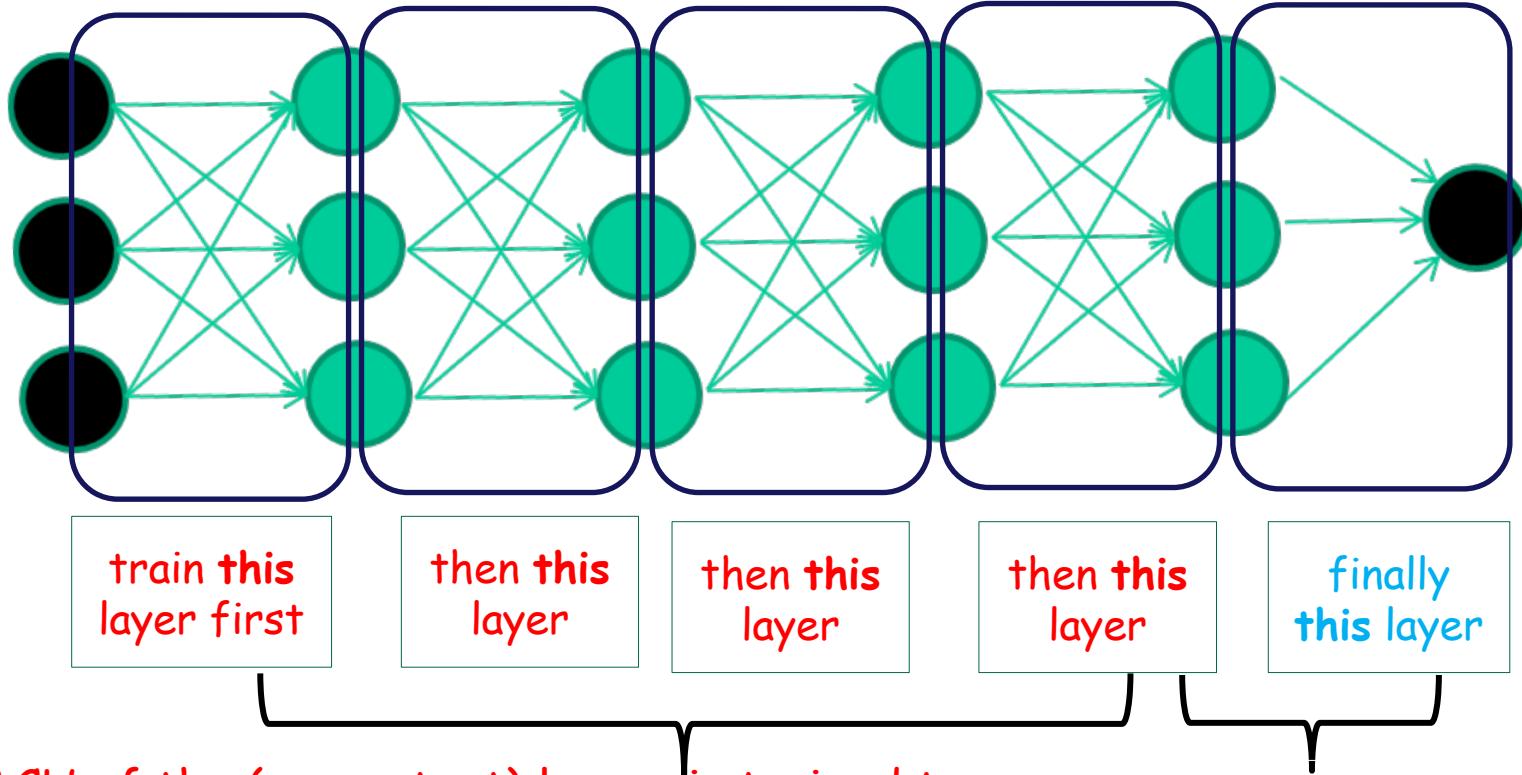


YOU ARE HERE!

# General Architecture of a Deep Network

1. Unsupervised pre-training of neural network using unlabeled data
  - aka. unsupervised learning of features
2. Supervised training with labeled data using features learned from above with a standard classifier
  - Eg. an ANN, SVM, ...

# Training a Deep NN



EACH of the (non-output) layers is trained to learn a representation of the data (eg. autoencoder) aka “pretraining the network with unsupervised data”

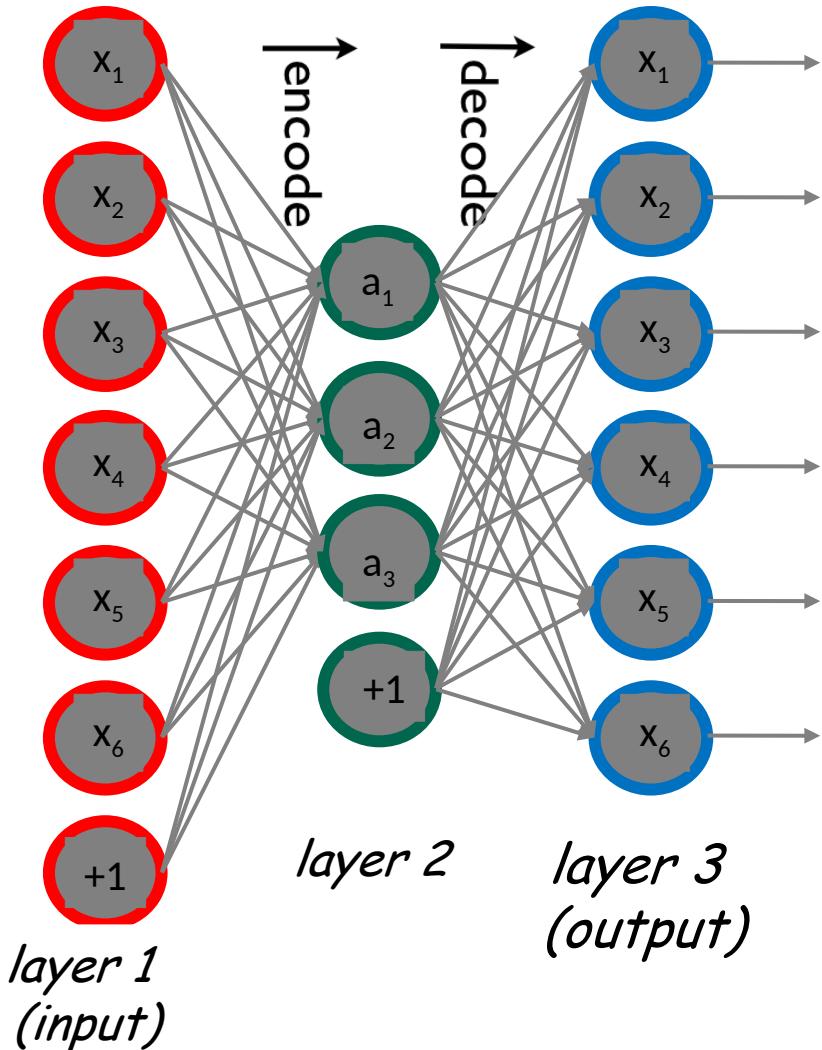
Use pretrained representations to feed a regular ANN with a smaller set of labelled data.

# Autoencoders

- To learn a representation of the data, we can use:
  - Deep Belief Networks
    - Mid 2000's: Geoffrey Hinton trains a deep network by:
      - Stacking Restricted Boltzmann Machines (RBM's) on top of one another - deep belief network
      - Training layer by layer on un-labeled data
      - Then, using back propagation to fine tune weights on labeled data
  - Autoencoders
    - 2006: Yoshua Bengio et al. does something similar using auto-encoders instead of RBM's
- Both are 2 layer neural networks that learn to model their inputs



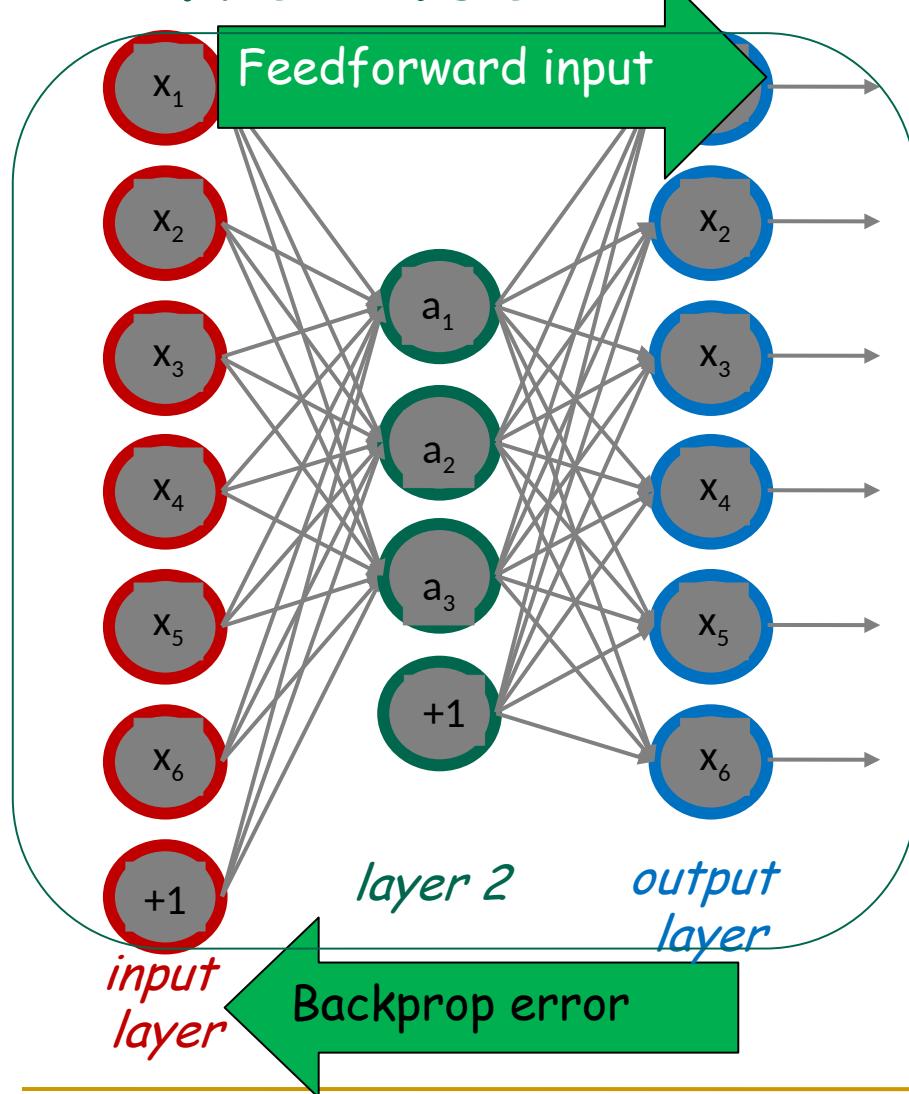
# Autoencoder



- The network is trained to output the input i.e. learn the identity function.
- Trivial... unless, we impose constraints:
  - Nb of units in layer 2 < nb of input units (learn compressed representation)  
OR
  - Constrain layer 2 to be sparse (i.e. many connections are "disabled")

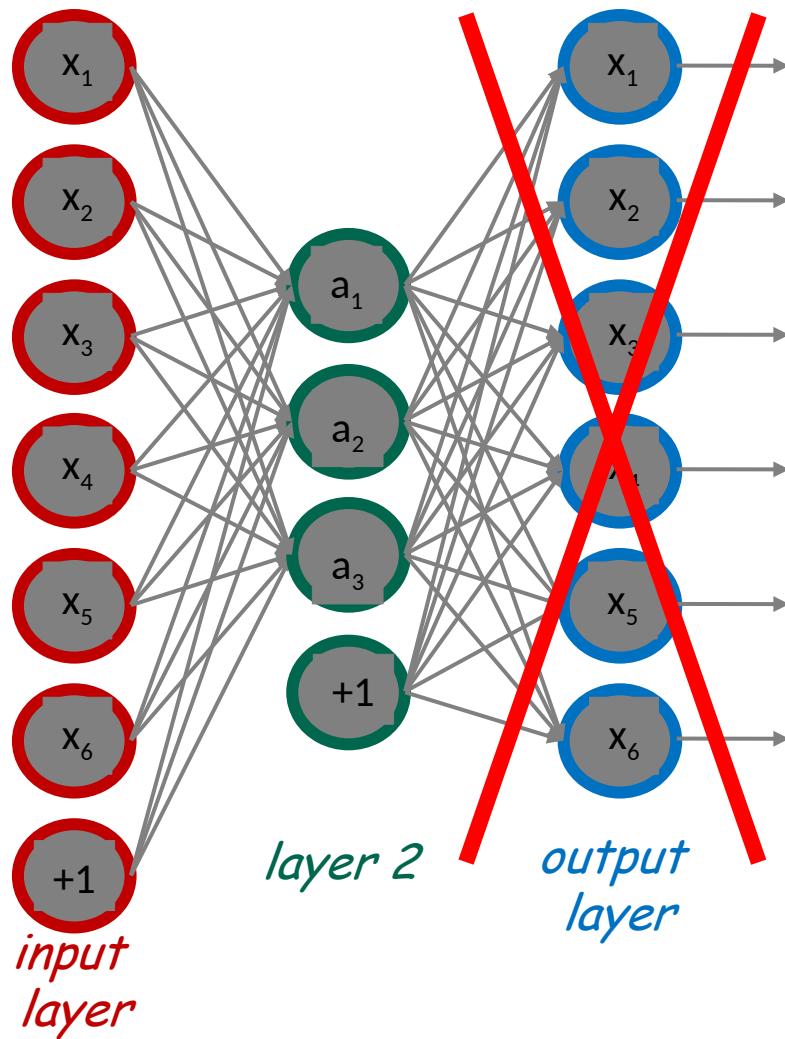
→ Worksheet #6 ("Autoencoder")

# Autoencoder

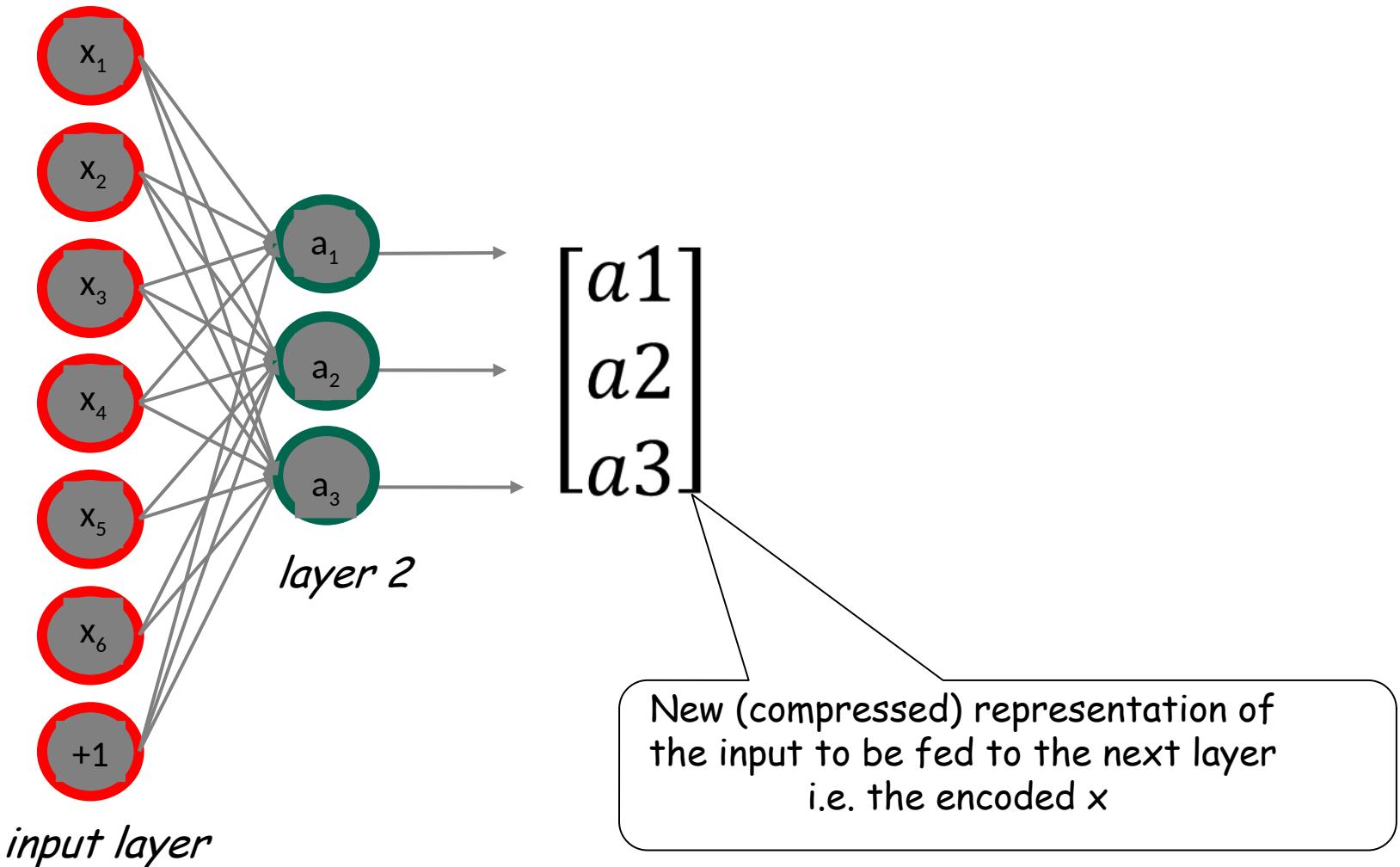


→ Worksheet #6  
("Autoencoder Activation")

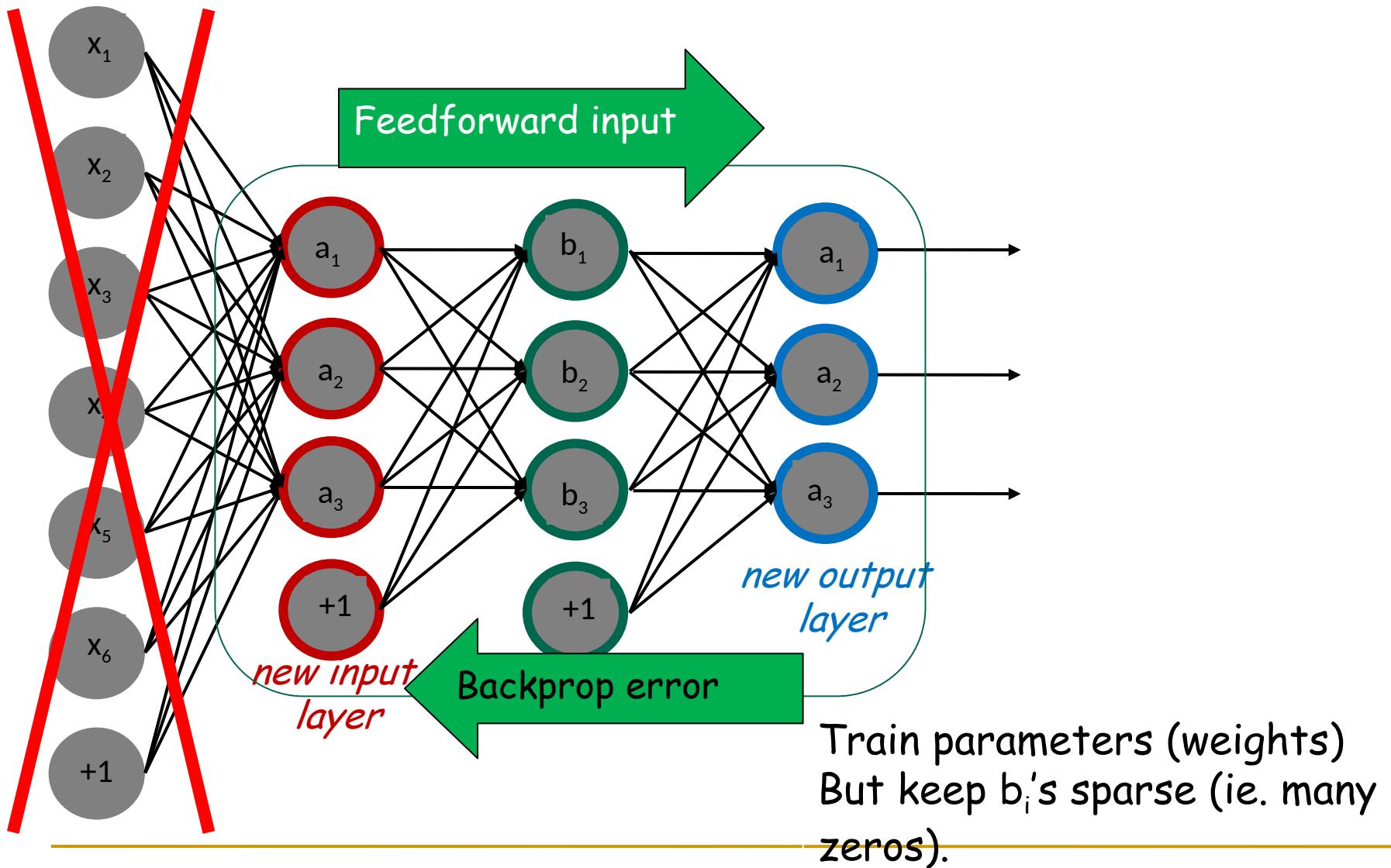
# Autoencoder



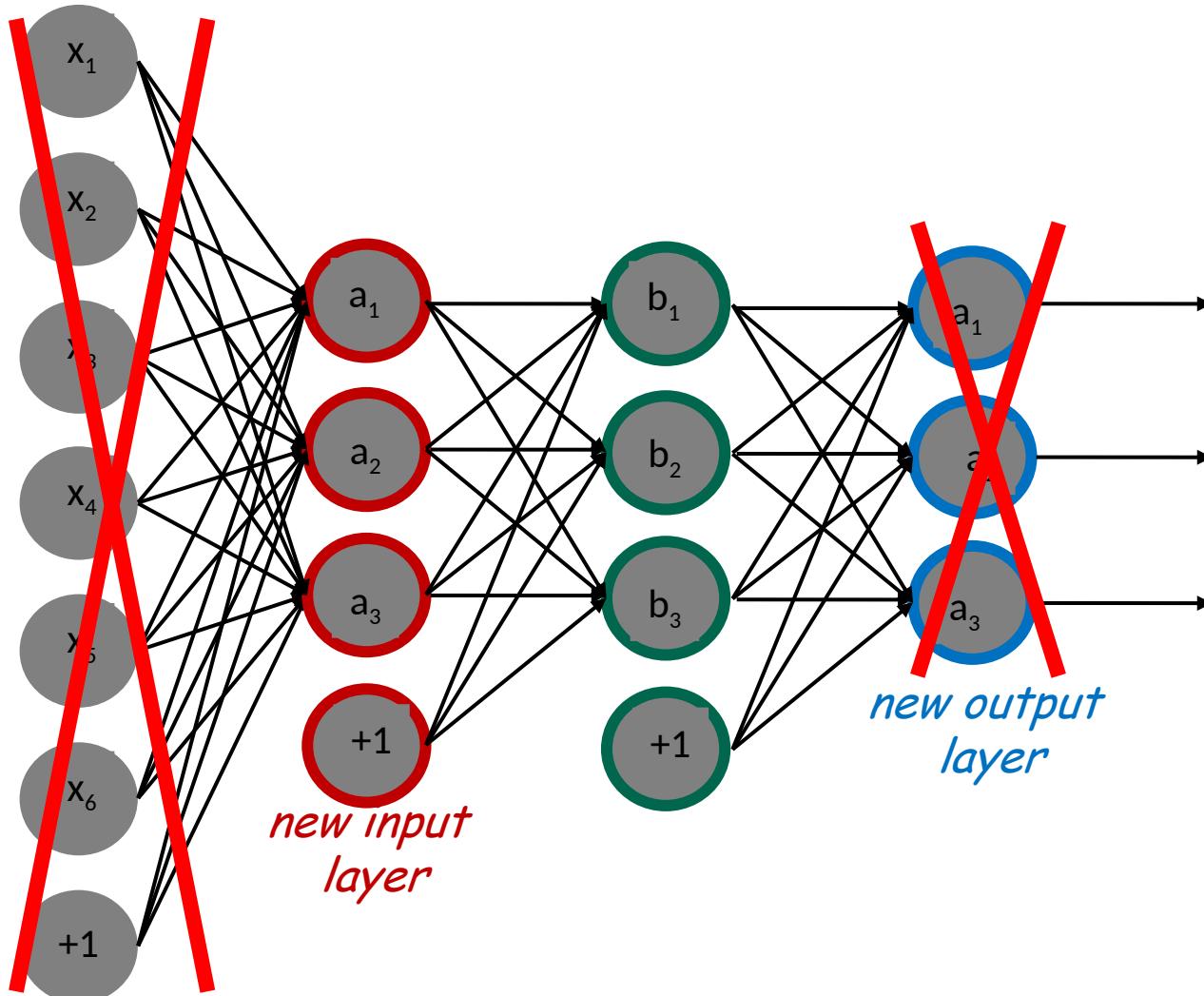
# Autoencoder



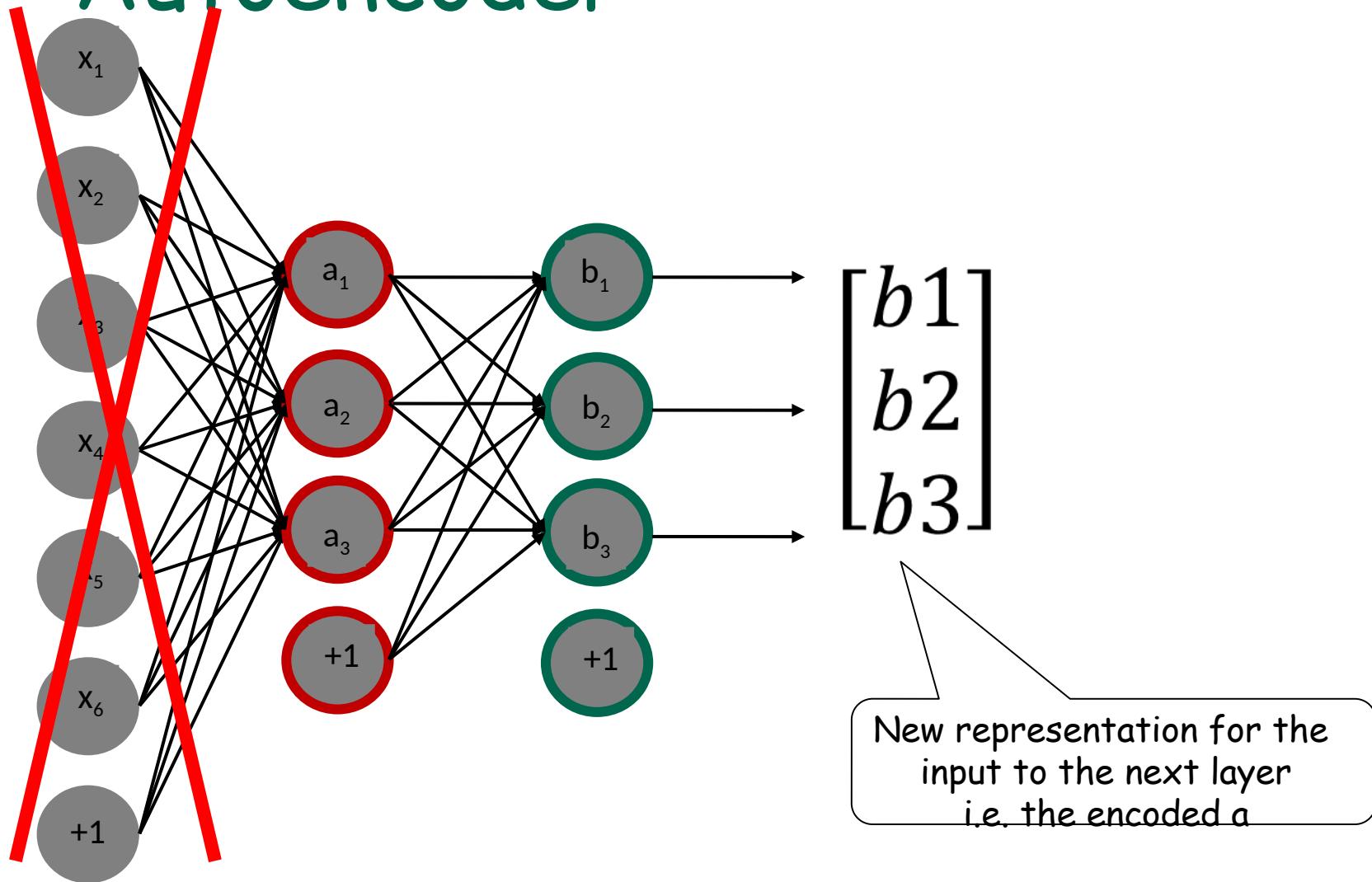
# Autoencoder



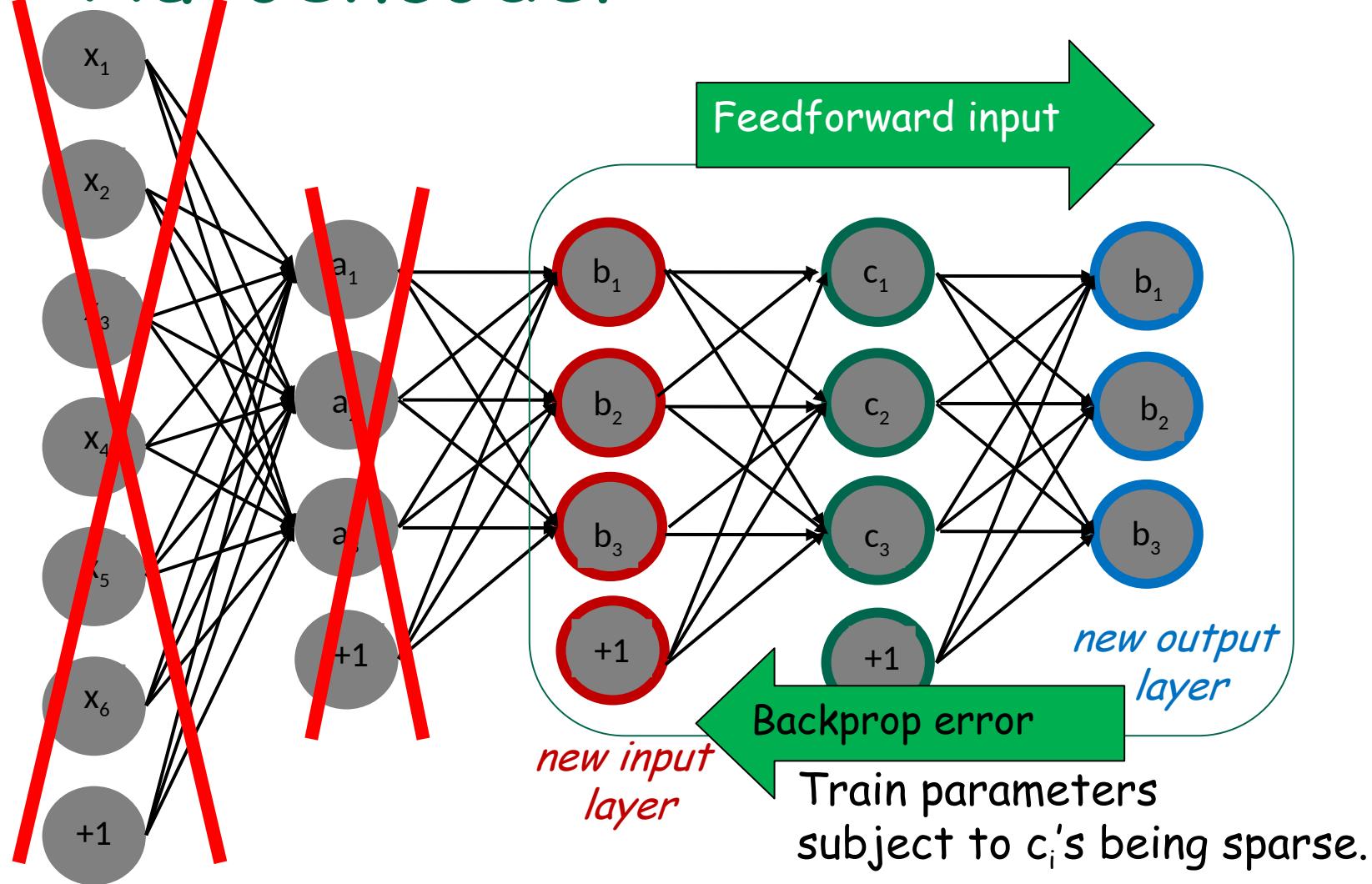
# Autoencoder



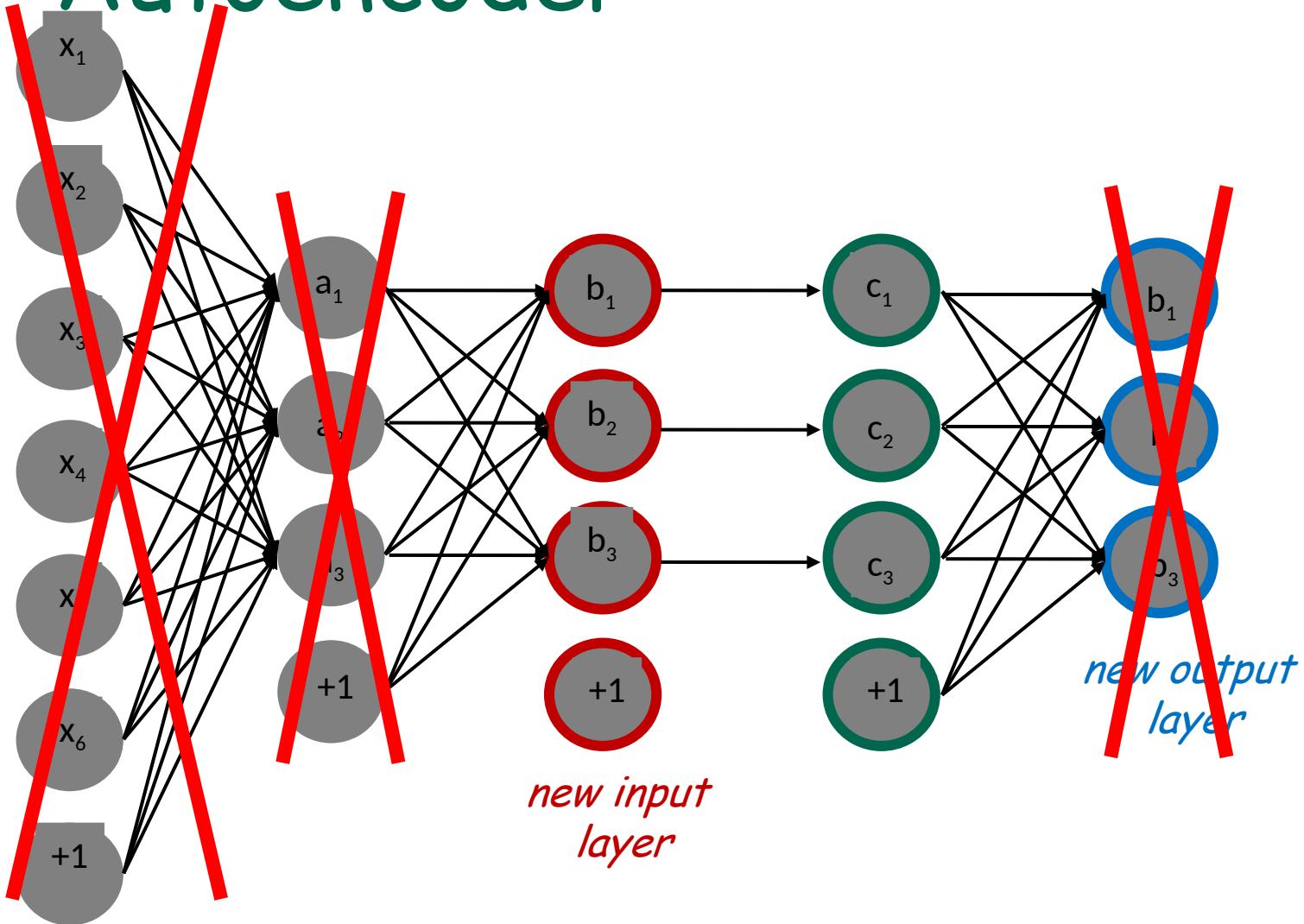
# Autoencoder



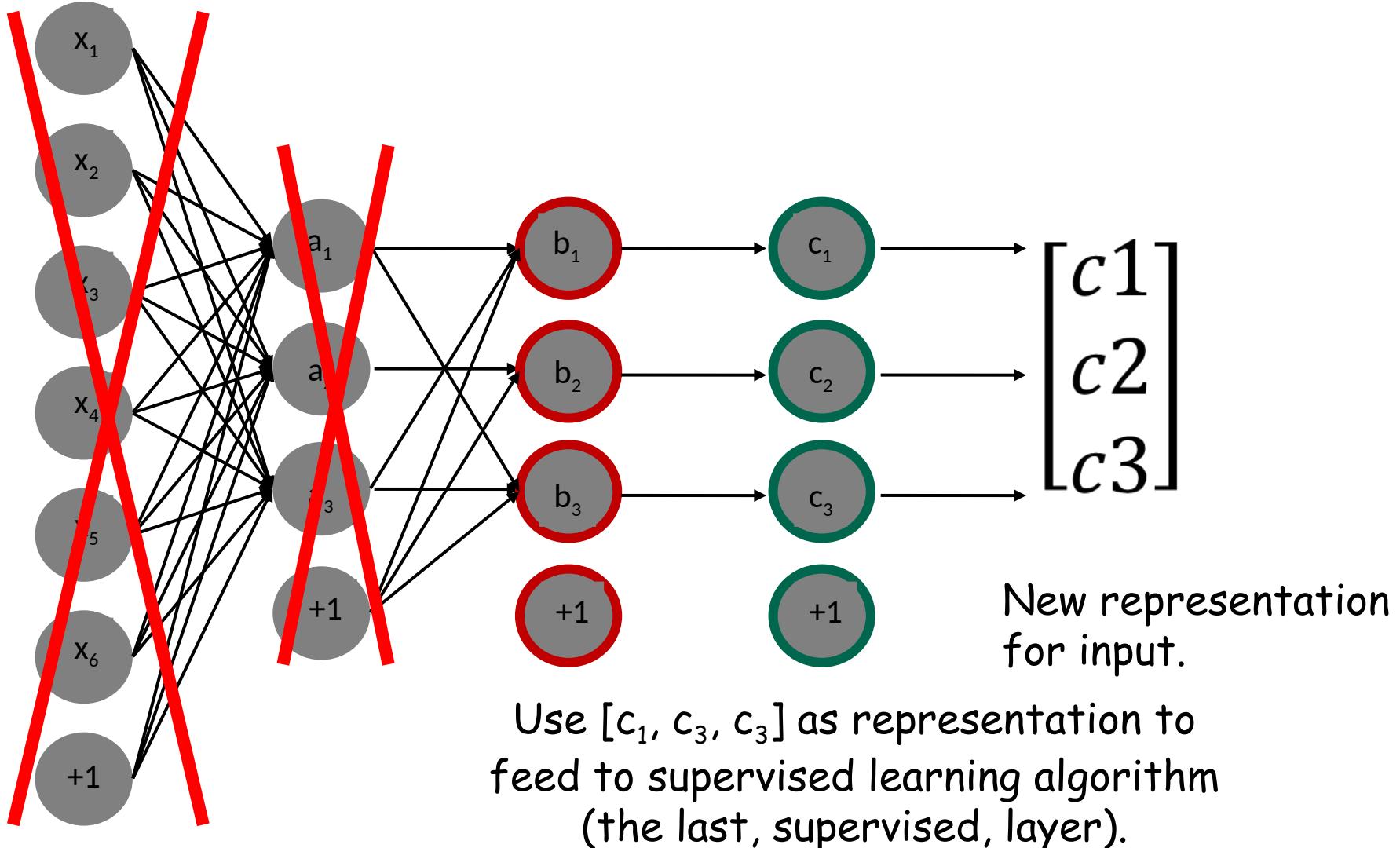
# Autoencoder



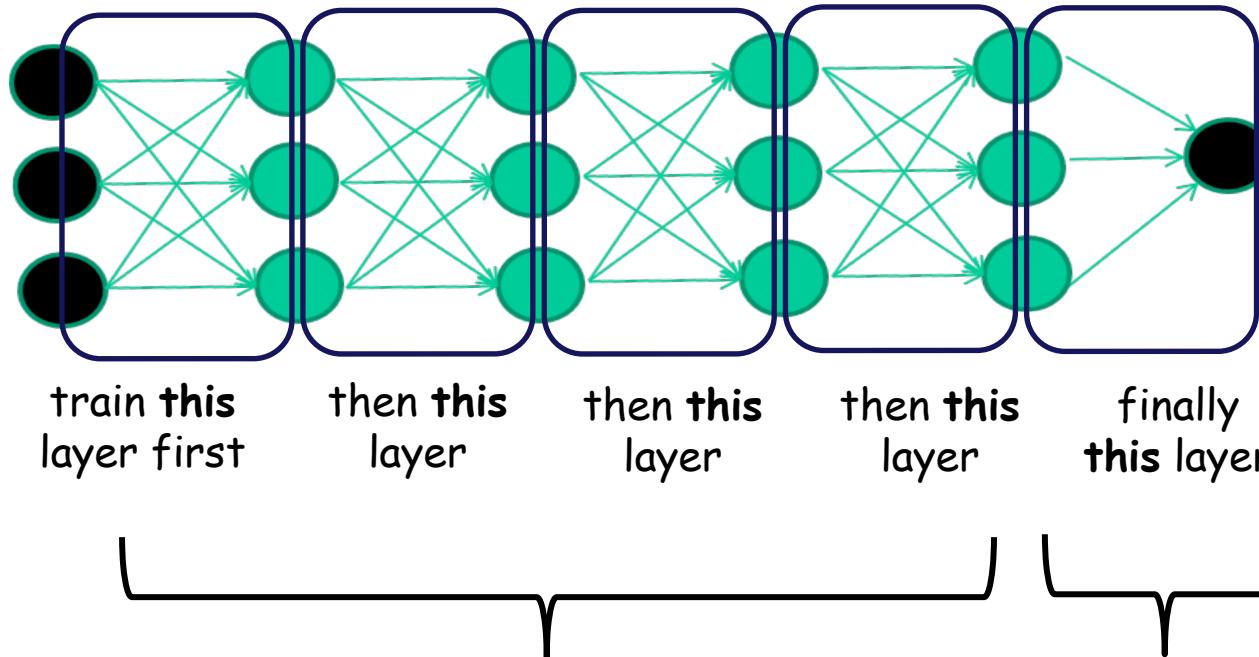
# Autoencoder



# Autoencoder



# Training a Deep NN



Use of **unlabelled** data to “pretrain” the network  
i.e. learn more and more abstract feature representations

Use pretrained representations to feed a regular ANN with a **smaller set of labelled data**.

# Many Types of Neural Networks

A mostly complete chart of

## Neural Networks

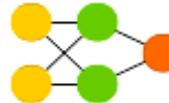
©2016 Fjodor van Veen - asimovinstitute.org

- Backfed Input Cell
- Input Cell
- △ Noisy Input Cell
- Hidden Cell
- Probabilistic Hidden Cell
- △ Spiking Hidden Cell
- Output Cell
- Match Input Output Cell
- Recurrent Cell
- Memory Cell
- △ Different Memory Cell
- Kernel
- Convolution or Pool

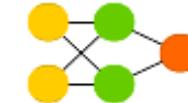
Perceptron (P)



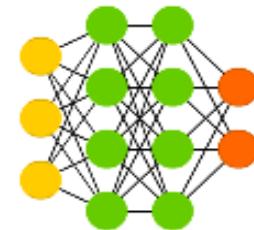
Feed Forward (FF)



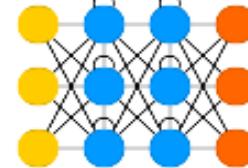
Radial Basis Network (RBF)



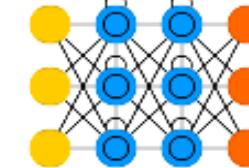
Deep Feed Forward (DFF)



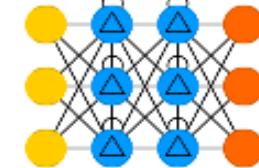
Recurrent Neural Network (RNN)



Long / Short Term Memory (LSTM)



Gated Recurrent Unit (GRU)



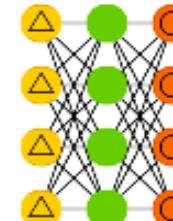
Auto Encoder (AE)



Variational AE (VAE)



Denoising AE (DAE)



Sparse AE (SAE)



# Many Types of Deep Networks (con't)



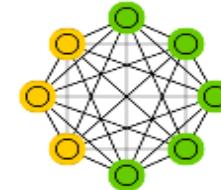
Markov Chain (MC)



Hopfield Network (HN)



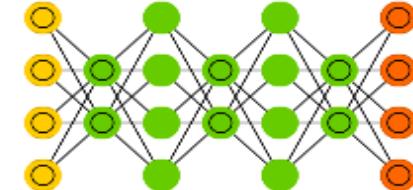
Boltzmann Machine (BM)



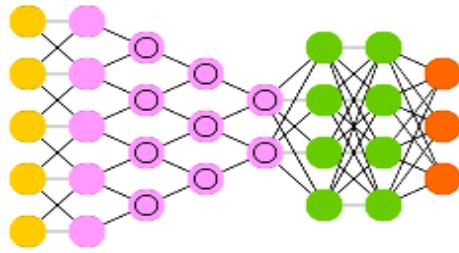
Restricted BM (RBM)



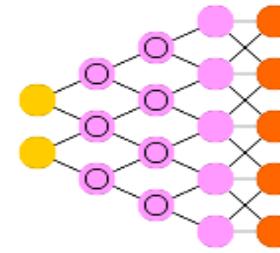
Deep Belief Network (DBN)



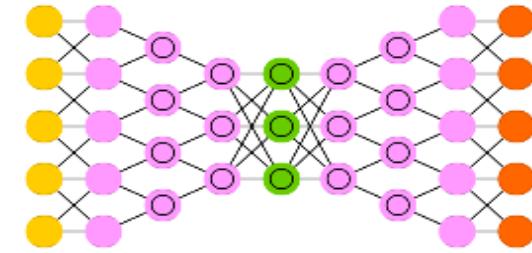
Deep Convolutional Network (DCN)



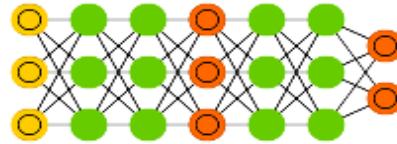
Deconvolutional Network (DN)



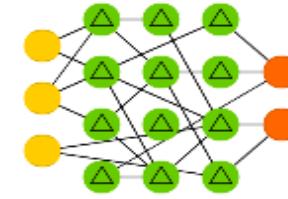
Deep Convolutional Inverse Graphics Network (DCIGN)



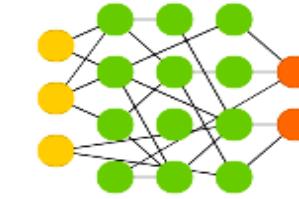
Generative Adversarial Network (GAN)



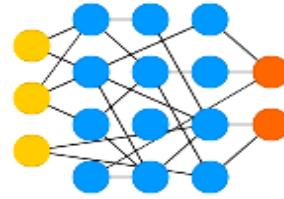
Liquid State Machine (LSM)



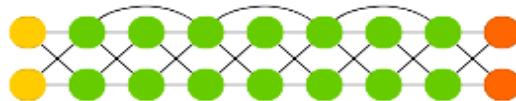
Extreme Learning Machine (ELM)



Echo State Network (ESN)



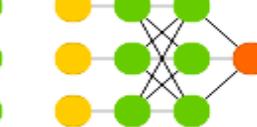
Deep Residual Network (DRN)



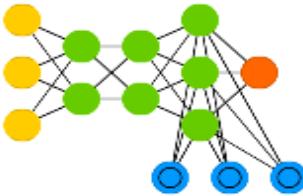
Kohonen Network (KN)



Support Vector Machine (SVM)



Neural Turing Machine (NTM)

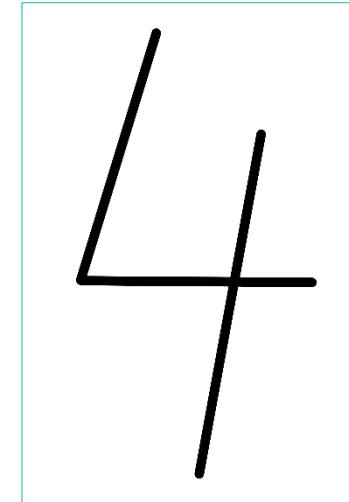
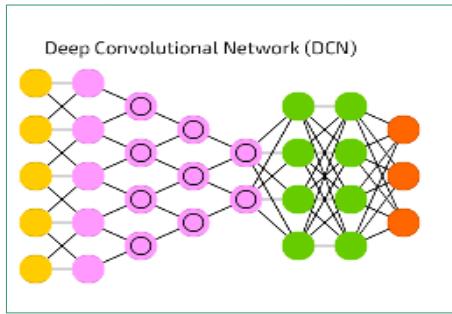


# Today

1. Motivation
  2. Feature Learning
  3. Early Training of Deep Neur
  4. CNNs for Image Processing
  5. Conclusion
- 
- works

# CNNs for Image Processing

CNNs = Convolutional Neural Networks



25	2	1	44
223	7	6	60
196	8	2	148
249	1	3	40
60	7	1	154
59	1	7	213
214	7	3	163
89	182	219	13
74	146	113	72
89	18	244	85
1	4	8	97
3	4	2	121
2	1	2	131
7	6	8	47
3	5	5	126
7	6	8	121
5	3	1	237

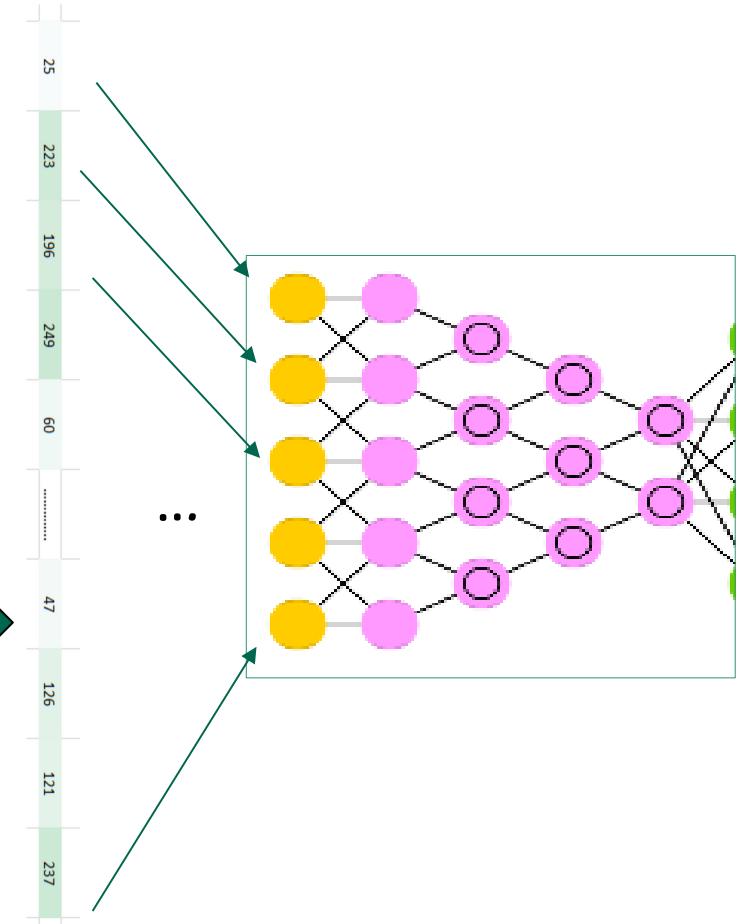
Image of a 4 in grey scale  
Value = 0-> white .... 255->black

# CNNs for Image Processing

- Standard input of the image in the ANN:

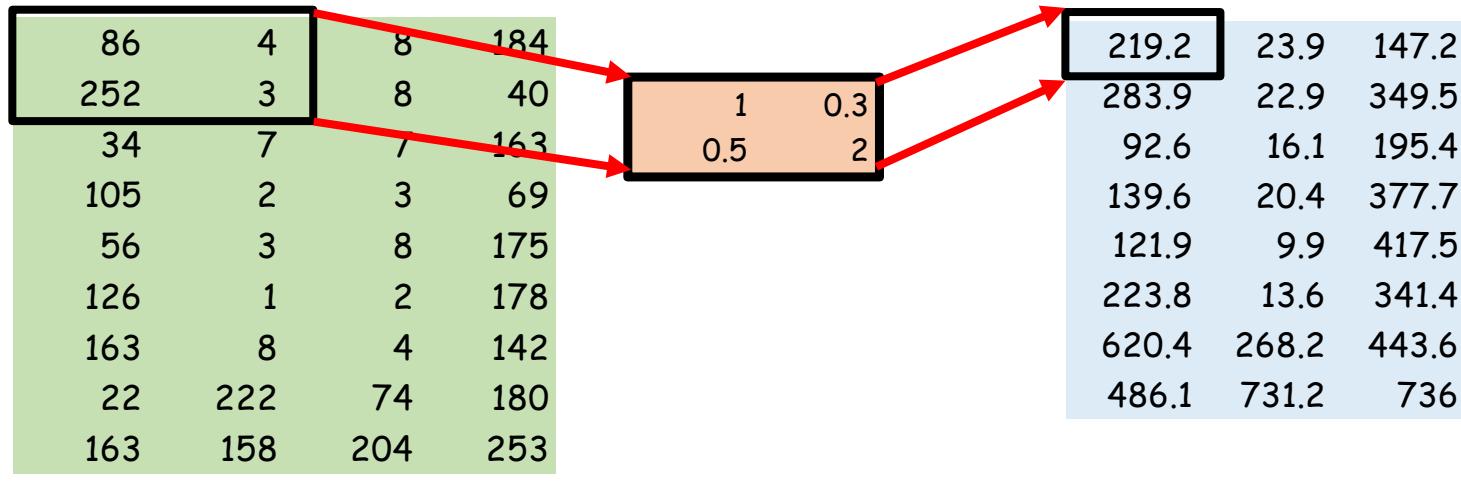
- 1 pixel = 1 input.
- Eg. color image of  $200 \times 200 \times 3$  channels (RGB)
  - > in a fully connected ANN, a neuron of the input layer has  $200 \times 200 \times 3 = 120,000$  weights
  - > huge number of parameters, can easily overfit
- We linearize the image ==> We lose spatial information

25	2	1	44
223	7	6	60
196	8	2	148
249	1	3	40
60	7	1	154
59	1	7	213
214	7	3	163
89	182	219	13
74	146	113	72
89	18	244	85
1	4	8	97
3	4	2	121
2	1	2	131
7	6	8	47
3	5	5	126
7	6	8	121
5	3	1	237



# Convolutional Layer

- Use a filter (aka kernel) that “convolves” on the image
- Filter = small weight matrix to learn



$I$  (original image)

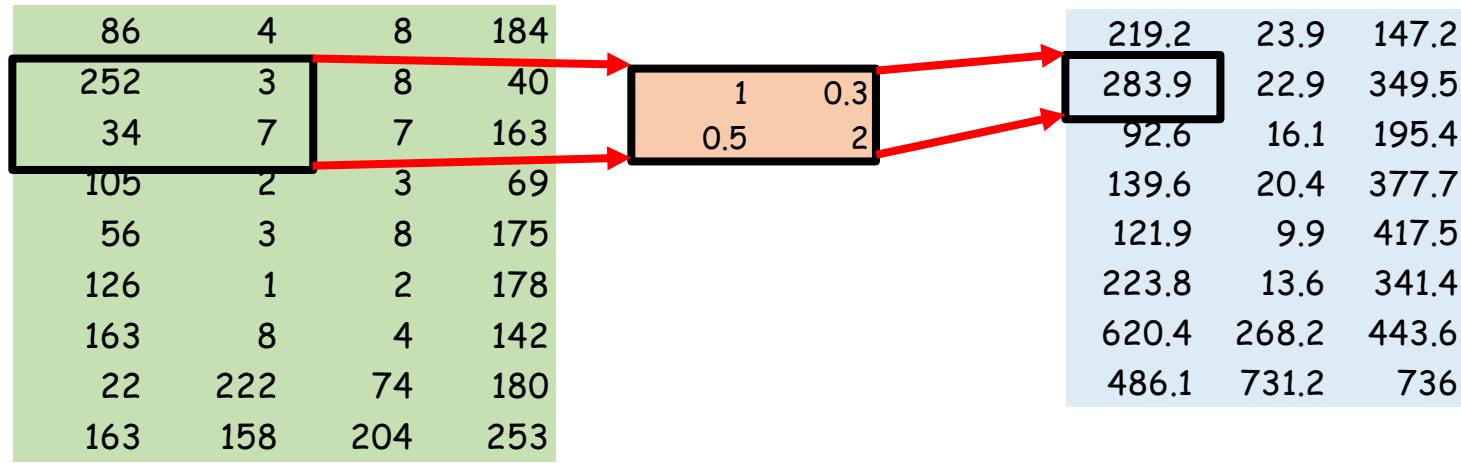
$W$  (filter)

$C$  (activation map)

$$\begin{aligned}C_{11} &= (I_{11} \times W_{11}) + (I_{12} \times W_{12}) + (I_{21} \times W_{21}) + (I_{22} \times W_{22}) \\&= 86 \times 1 + 4 \times 0.3 + 252 \times 0.5 + 3 \times 2 = 219.2\end{aligned}$$

# Convolutional Layer

- Use a filter (aka kernel) that “convolves” on the image
- Filter = small weight matrix to learn



$I$  (original image)

$W$  (filter)

$C$  (activation map)

$$\begin{aligned}C_{12} &= (I_{12} \times W_{11}) + (I_{13} \times W_{12}) + (I_{22} \times W_{21}) + (I_{23} \times W_{22}) \\&= 252 \times 1 + 3 \times 0.3 + 34 \times 0.5 + 7 \times 2 = 283.9\end{aligned}$$

# Convolutional Layer

- Use a filter (aka kernel) that “convolves” on the image
- Filter = small weight matrix to learn

86	4	8	184
252	3	8	40
34	7	7	163
105	2	3	69
56	3	8	175
126	1	2	178
163	8	4	142
22	222	74	180
163	158	204	253

1	0.3
0.5	2

219.2	23.9	147.2
283.9	22.9	349.5
92.6	16.1	195.4
139.6	20.4	377.7
121.9	9.9	417.5
223.8	13.6	341.4
620.4	268.2	443.6
486.1	731.2	736

$I$ (original image)

$W$ (filter)

$C$ (activation map)

$$\begin{aligned}C_{13} &= (I_{13} \times W_{11}) + (I_{14} \times W_{12}) + (I_{23} \times W_{21}) + (I_{24} \times W_{22}) \\&= 34 \times 1 + 7 \times 0.3 + 105 \times 0.5 + 2 \times 2 = 92.6\end{aligned}$$

# Convolutional Layer

- Use a filter (aka kernel) that “convolves” on the image
- Filter = small weight matrix to learn

86	4	8	184
252	3	8	40
34	7	7	163
105	2	3	69
56	3	8	175
126	1	2	178
163	8	4	142
22	222	74	180
163	158	204	253

$I$ (original image)

1	0.3
0.5	2

$W$ (filter)

219.2	23.9	147.2
283.9	22.9	349.5
92.6	16.1	195.4
139.6	20.4	377.7
121.9	9.9	417.5
223.8	13.6	341.4
620.4	268.2	443.6
486.1	731.1	736

$C$ (activation map)

$$\begin{aligned}C_{38} &= (I_{37} \times W_{11}) + (I_{47} \times W_{12}) + (I_{38} \times W_{21}) + (I_{37} \times W_{22}) \\&= 74 \times 1 + 180 \times 0.3 + 204 \times 0.5 + 253 \times 2 = 736\end{aligned}$$

# Learn the Filters

18	54	51	239	244	188
55	121	75	78	95	88
35	24	204	113	109	221
3	154	104	235	25	130
15	253	225	159	78	233
68	85	180	214	245	0

$I$  ( $6 \times 6$ )

1	0	1
0	1	0
1	0	1

$W$  ( $3 \times 3$ )

429	505	686	856
261	792	412	640
633	653	851	751
608	913	713	657

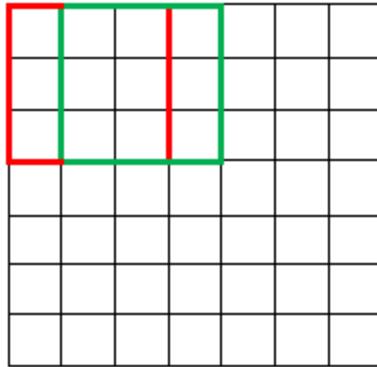
$C$  ( $4 \times 4$ )

- The weight matrix (filter/kernel) behaves like a filter
- The network learns the values of the filter(s) that activate when they “see” some visual feature that is useful to identify the object (the final classification)
  - Ex. a horizontal line, a blotch of some color, a circle...

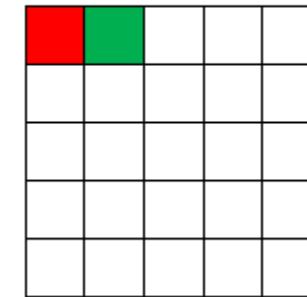
# Convolution Hyper-parameters

1. Stride
2. Padding

# Stride

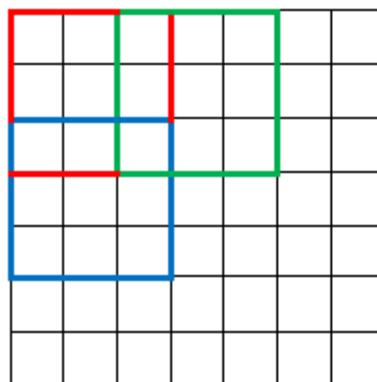


$(7 \times 7)$

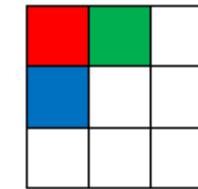


$W (3 \times 3)$  with stride = 1

$C (5 \times 5)$



$(7 \times 7)$



$W (3 \times 3)$  with stride = 2

$C (3 \times 3)$

# Padding

9	0	0	1
1	0	1	0
0	1	1	2
2	1	0	1

0	0	0
0	1	0
0	0	0

0	1
1	1

9 not picked up ;-(

filter should pick up high values surrounded by low values

0	0	0	0	0	0
0	9	0	0	1	0
0	1	0	1	0	0
0	0	1	1	2	0
0	2	1	0	1	0
0	0	0	0	0	0

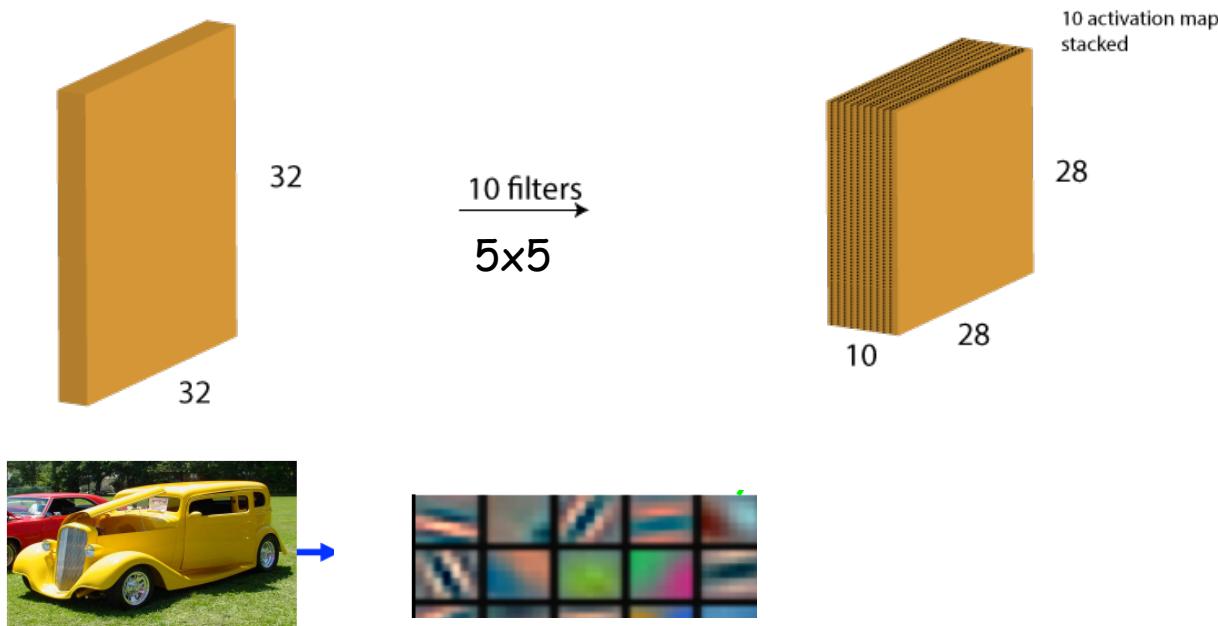
0	0	0
0	1	0
0	0	0

9	0	0	1
1	0	1	0
0	1	1	2
2	1	0	1

9 picked up :-)

# Learn Several Filters

- So we create 1 activation map per filter



# Pooling Layer

- Used to:
  - To reduce the size of the activation maps
  - So that we reduce the number of parameters of the network and hence avoid overfitting.
- Several strategies:
  - Max pooling

A diagram illustrating max pooling. On the left is a 4x4 input matrix with values: Row 1: 429, 505, 686, 856; Row 2: 261, 792, 412, 640; Row 3: 633, 653, 851, 751; Row 4: 608, 913, 713, 657. The top-left cell (429) is highlighted in yellow. An arrow points to the right, leading to a 2x2 output matrix: Row 1: 792, 856; Row 2: 913, 851. The top-left cell (792) is highlighted in yellow.

429	505	686	856
261	792	412	640
633	653	851	751
608	913	713	657

A diagram illustrating average pooling. On the left is a 4x4 input matrix with values: Row 1: 429, 505, 686, 856; Row 2: 261, 792, 412, 640; Row 3: 633, 653, 851, 751; Row 4: 608, 913, 713, 657. The top-left cell (429) is highlighted in yellow. An arrow points to the right, leading to a 2x2 output matrix: Row 1: 496.8, 648.5; Row 2: 701.8, 743. The top-left cell (496.8) is highlighted in yellow.

429	505	686	856
261	792	412	640
633	653	851	751
608	913	713	657

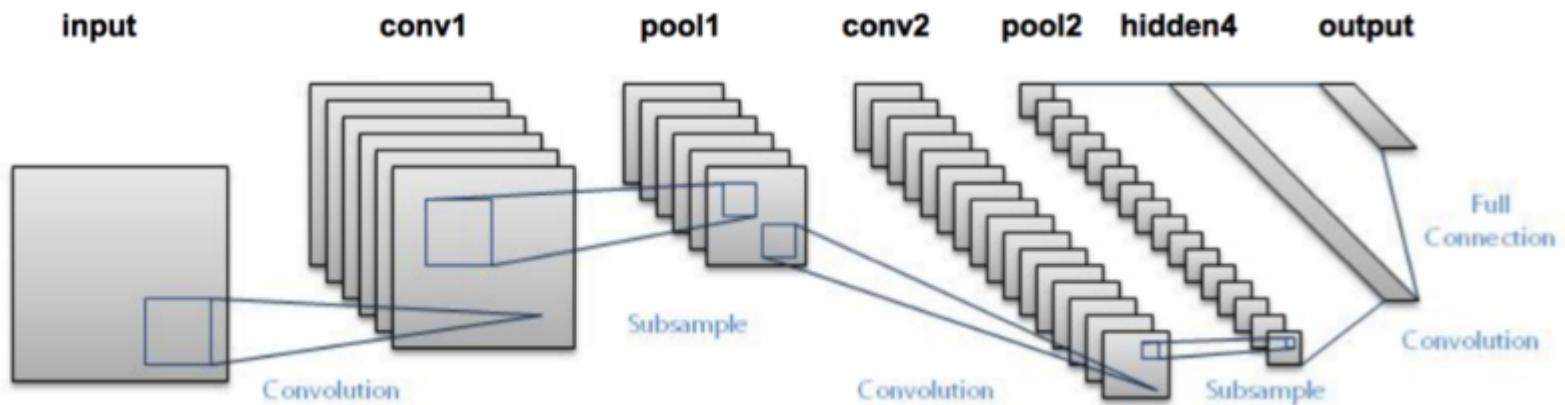
- Average pooling

- ...

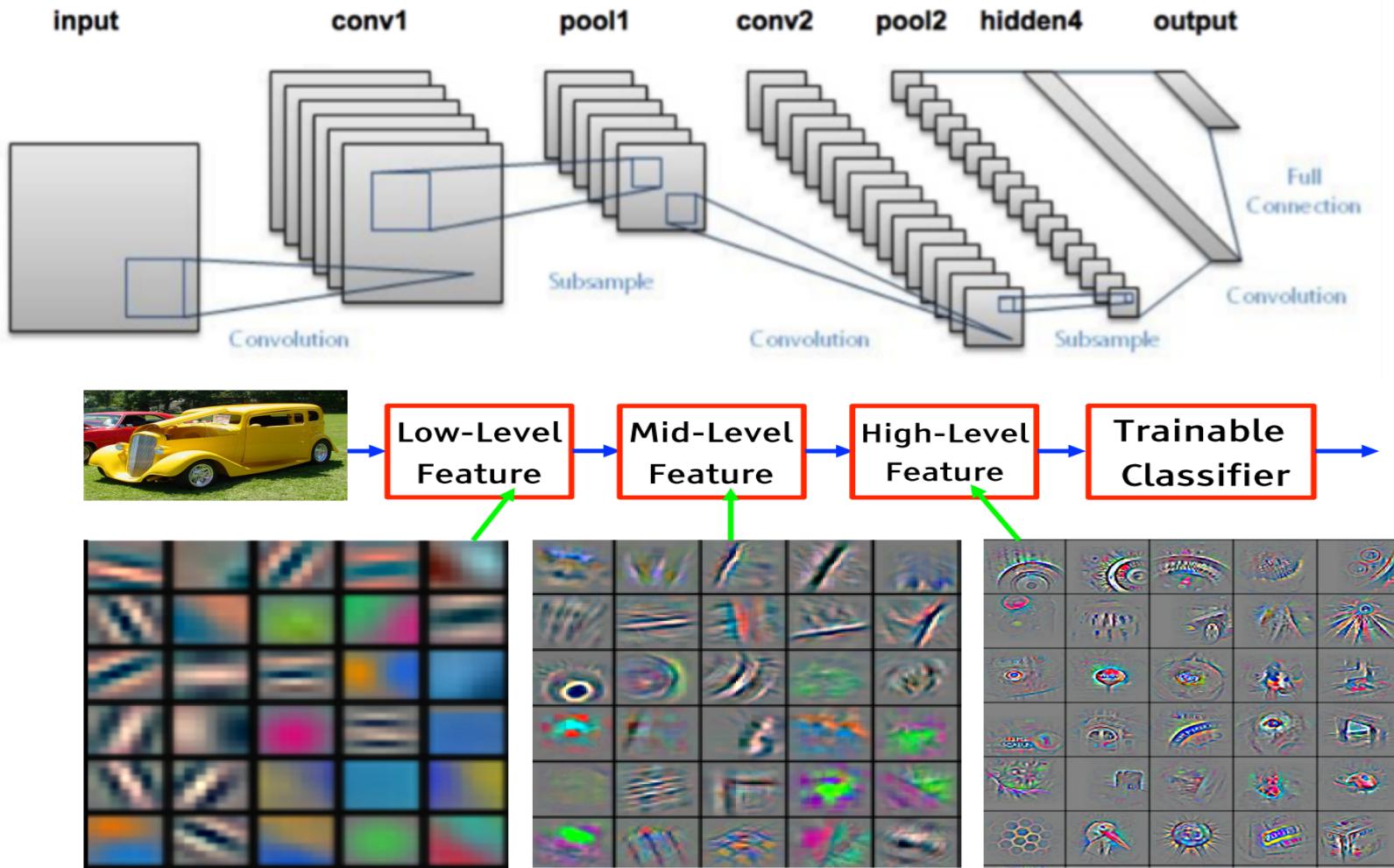
→ Worksheet #6 ("Pooling Layer")

# Architecture of a CNN

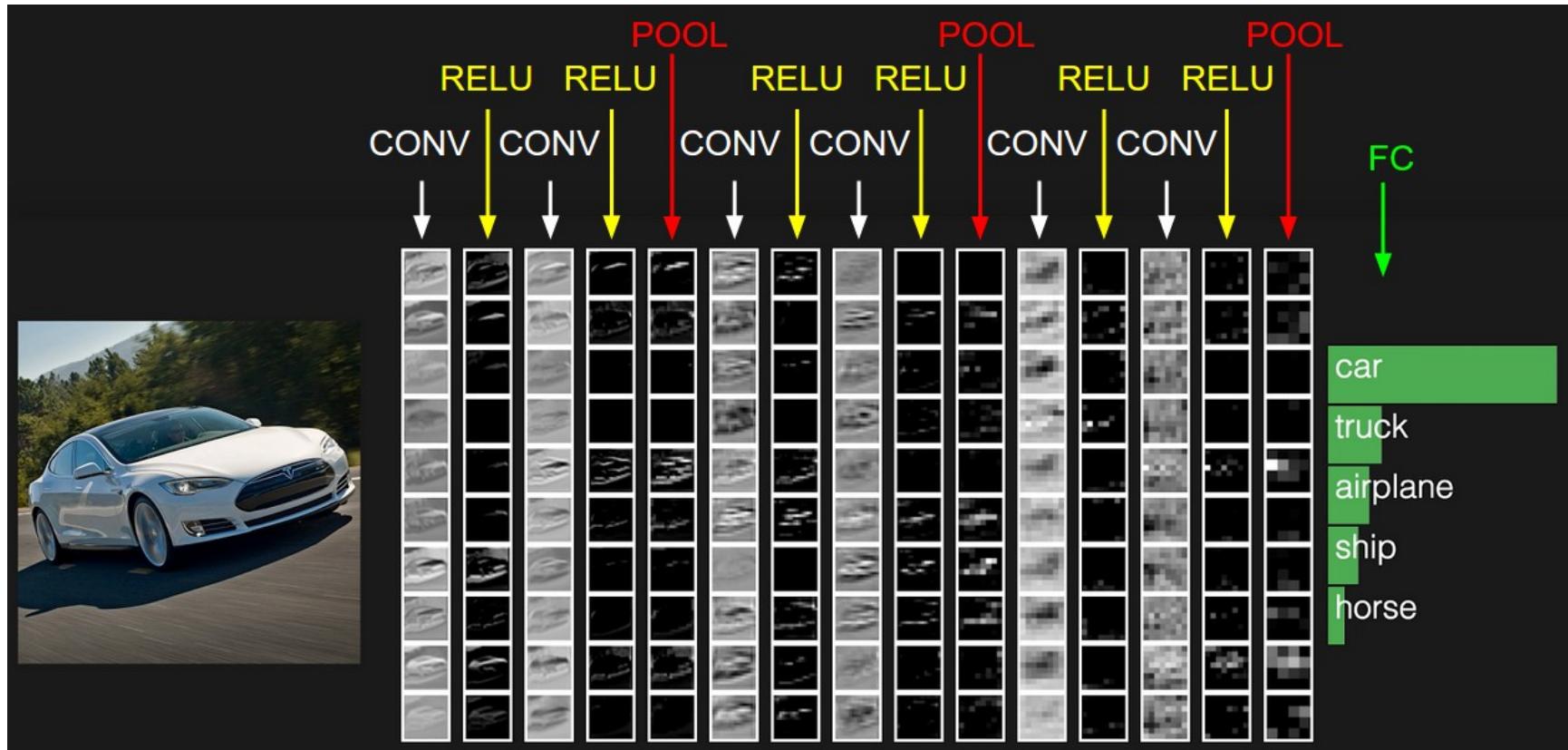
- Stack:
  - Convolutional Layers
  - Pooling Layers
- Finish off with:
  - A fully connected layer at the end for the final classification



# Learning a Hierarchy of Features



# Example of a CNN



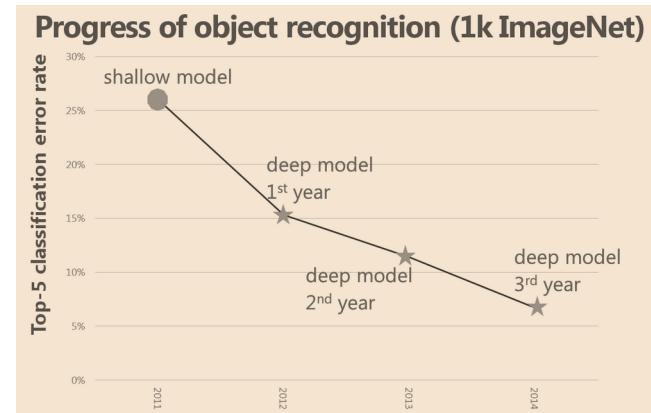
# Successful CNN Networks

- LeNet

- LeNet
- First successful applications of CNNs
- Developed by Yann LeCun in the 1990's
- used to read zip codes, digits, etc.

- AlexNet

- AlexNet
- First work that popularized CNNs for computer vision
- developed by Alex Krizhevsky, Ilya Sutskever and Geoff Hinton (U. of Toronto)
- In 2012 significantly outperformed all teams at the ImageNet ILSVRC challenge



# Why now?

## 1. Basic science

- Backpropagation did not work / overfitting...
- *now:* developed method for training, better activation functions, better architectures....
- Need for lots training data...
- *now:* we have massive amounts + unsupervised pre-training

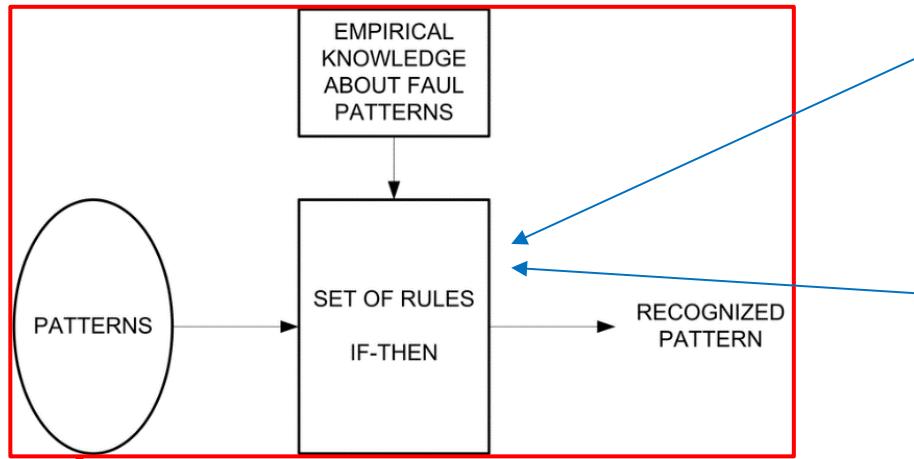
## 2. GPU computing

- Neural networks take very very long to train... (days, weeks)
- *now:* use of GPU's which are optimized for very fast matrix multiplication

## 3. Open Access to resources

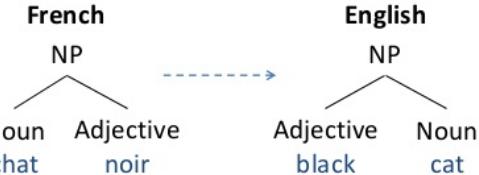
- *now:* Access to DL methods, code and frameworks
- *now:* Fast turnaround from idea to implementation

# History of AI



Rules written by experts (e.g. linguistics, medical doctors,...)

Rules hand-written by linguists

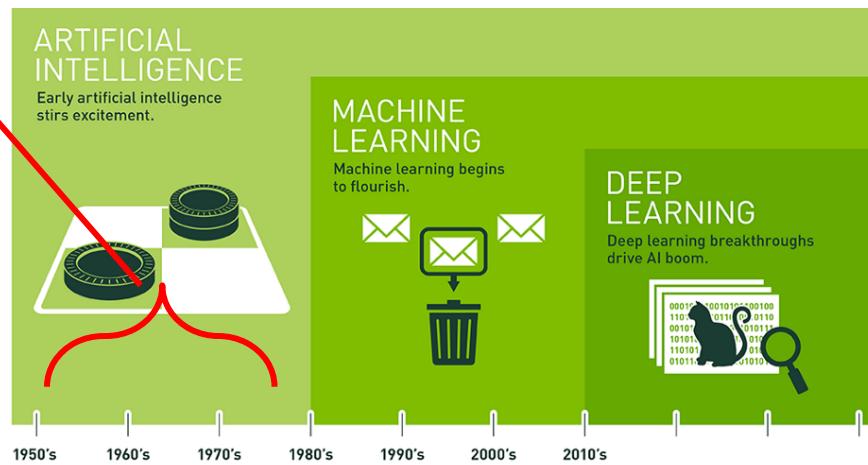


**IF**

- "yes" is equal to uniform\_layer\_flow
- THETA is greater than 45.0
- THETA is less than or equal to 90.0
- C4 is greater than (Lm/(0.8\*(Hs-H0)))
- C6 is greater than (Lb/(0.8\*(Hs-H0)))
- C9 is less than or equal to (Lt/(0.8^(Hs-H0)))

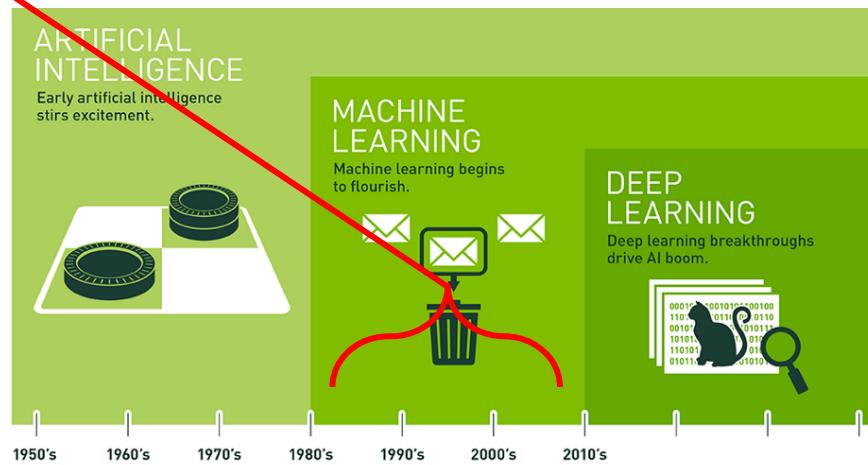
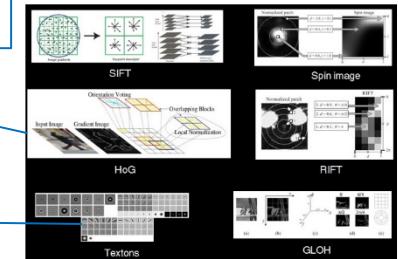
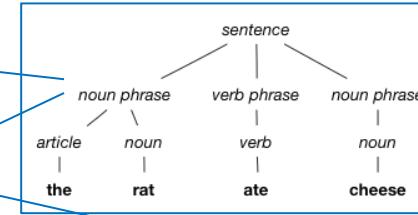
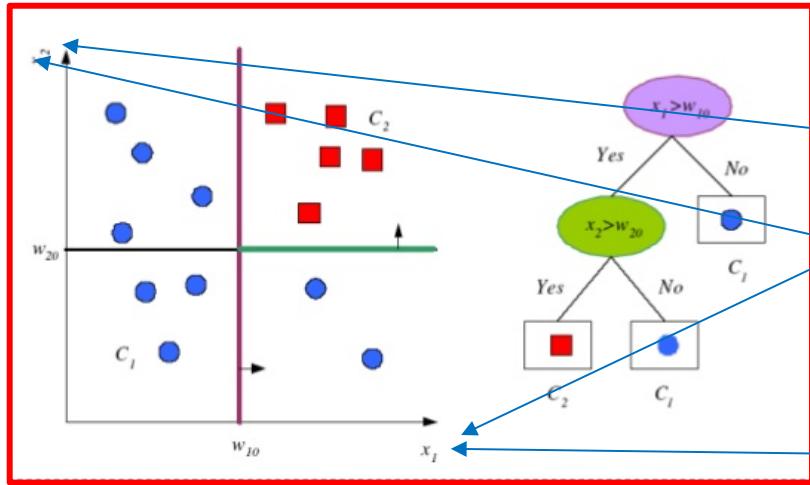
**Then**

- flow\_type\_ok is confirmed
- "V2" is assigned to flow\_type
- "No" is assigned to wake\_attachment
- Find coanda\_attachment\_value

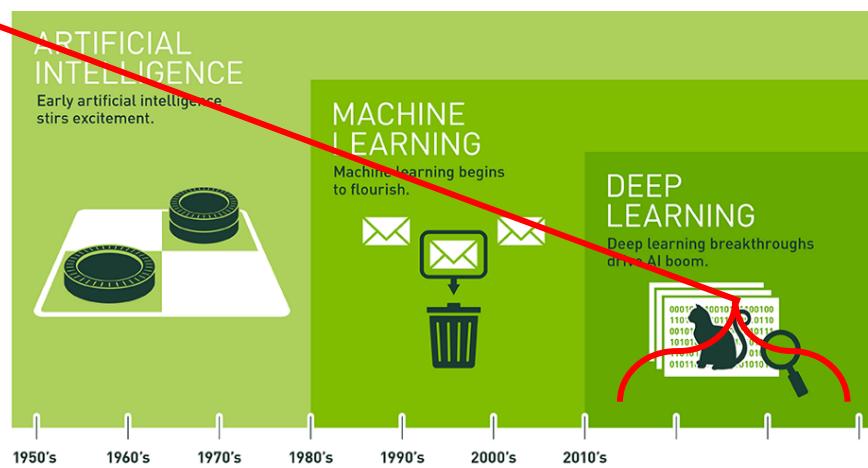
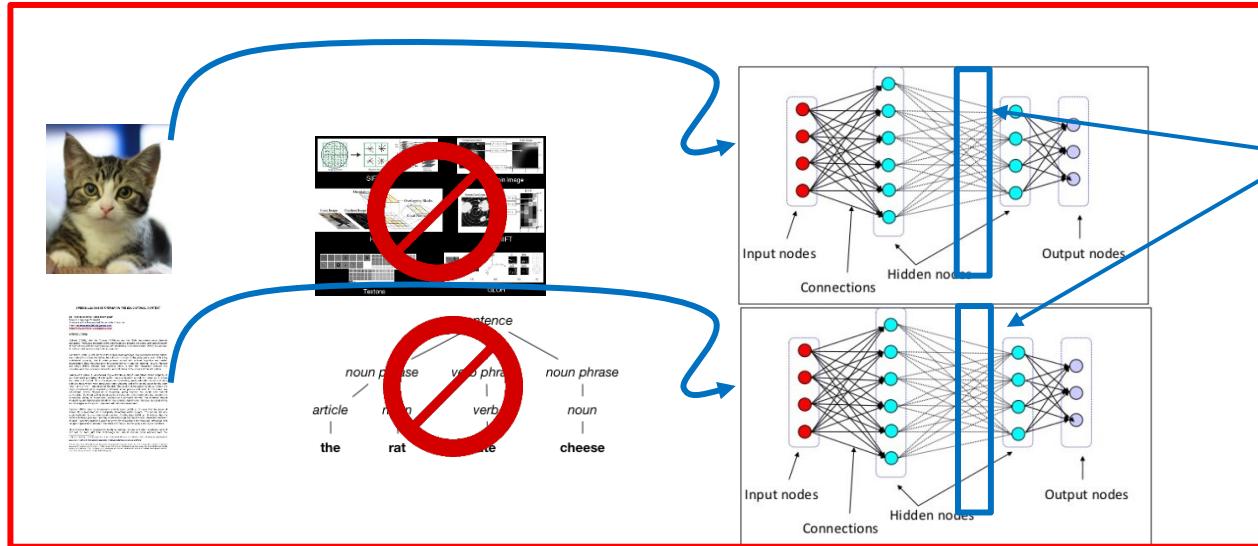


# History of AI

Rules learns via the data ;-)  
But: features identified by the experts  
(eg. linguistics, medical doctors,...)



# History of AI



<https://www.linkedin.com/pulse/goedels-incompleteness-theorem-emergence-ai-eberhard-schoeneburg/>

# Conclusion

- Deep Learning is thriving !
  - *vision*
  - *image processing*
  - *speech recognition*
  - *natural language processing*
  - ...
- Canada is a world leader in Deep Learning
  1. Montreal: (Bengio et al.) [MILA](#)
  2. Toronto: (Hinton et al.) [Vector Institute](#)
  3. Edmonton: (Sutton et al.) [AMII](#)

YOU ARE HERE!

THE END!