Introduction to Deep Learning

Acknowledgments

This presentation is heavily based on:

- http://cs.nyu.edu/~fergus/pmwiki/pmwiki.php
- http://deeplearning.net/reading-list/tutorials/
- http://deeplearning.net/tutorial/lenet.html
- http://ufldl.stanford.edu/wiki/index.php/UFLDL_Tutorial

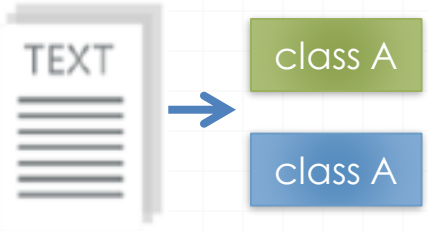
… and many other

Types of Learning

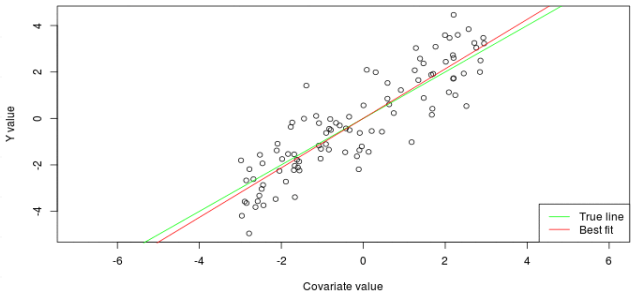
Supervised: Learning with a **labeled training** set
Example: email *classification* with already labeled emails

Unsupervised: Discover **patterns** in **unlabeled** data
Example: *cluster* similar documents based on text

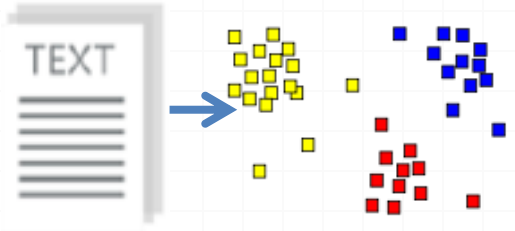
Reinforcement learning: learn to **act** based on **feedback/reward**
Example: learn to play Go, reward: *win or lose*



Classification



Regression

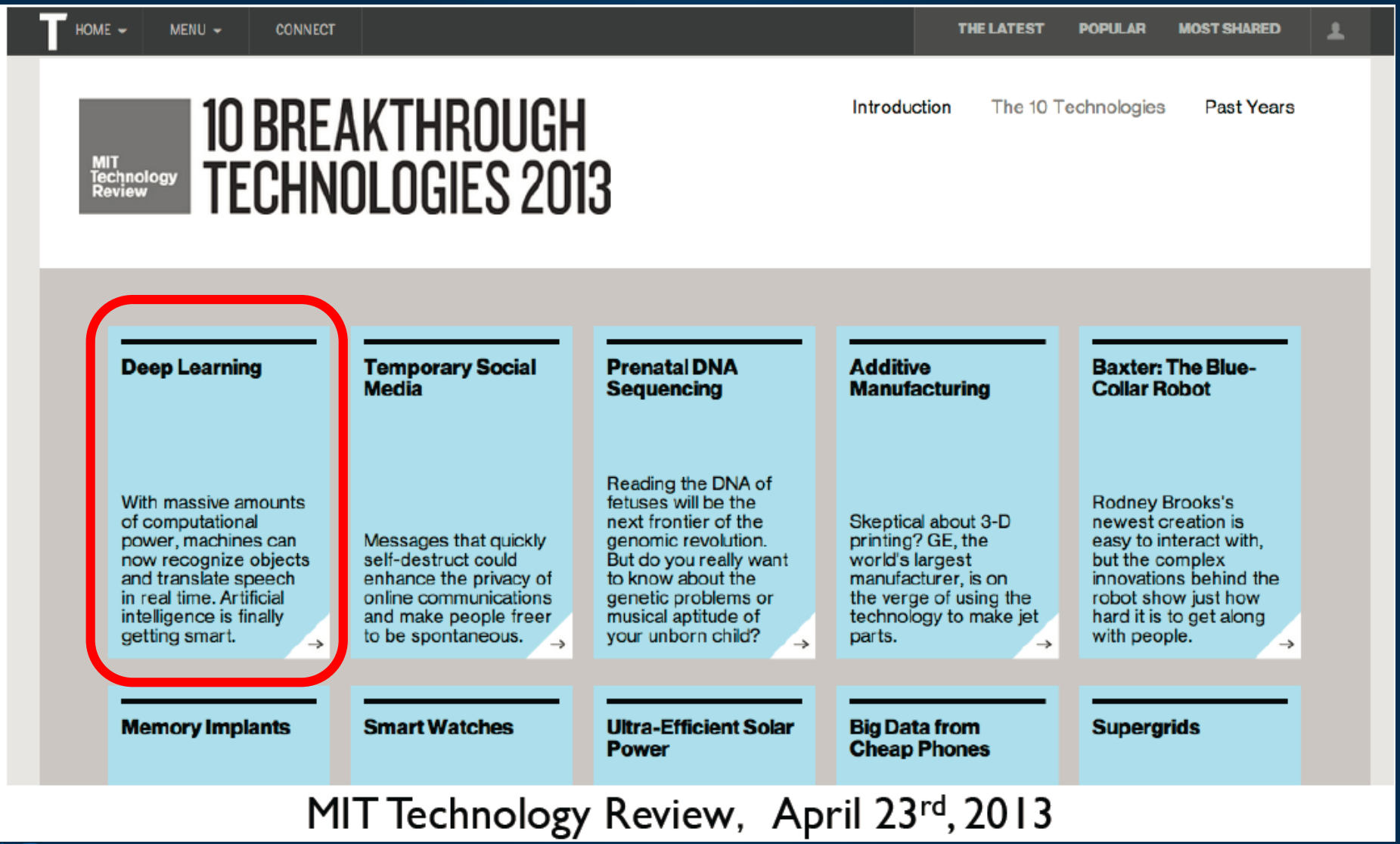


Clustering

Anomaly Detection
Sequence labeling
…

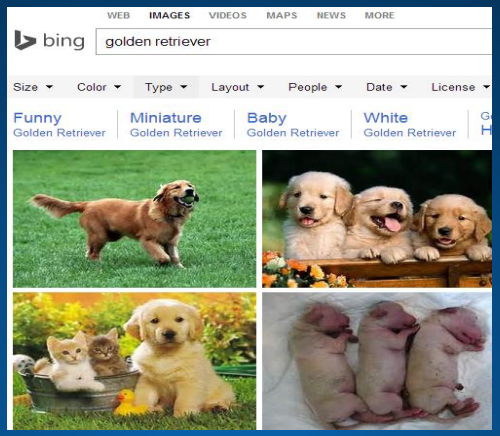
http://mbjoseph.github.io/2013/11/27/measure.html

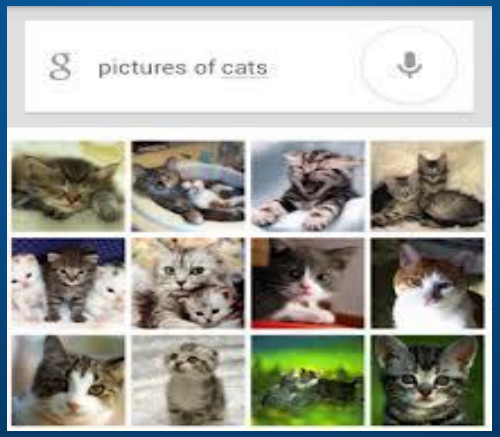
Buzz…



MIT Technology Review, April 23rd, 2013

Deep Learning – from Research to Technology





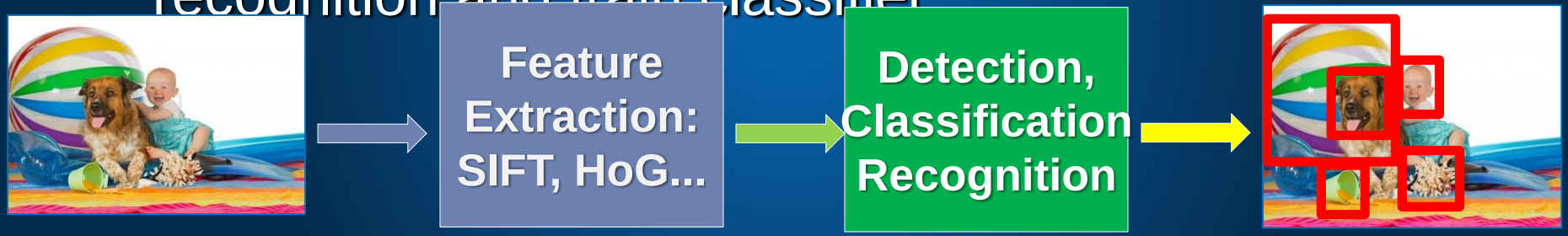
Deep Learning - breakthrough in visual and speech recognition

Classical Computer Vision Pipeline

Classical Computer Vision Pipeline.

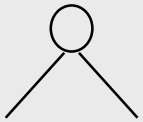
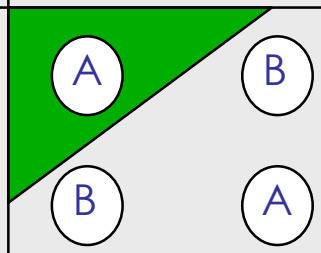
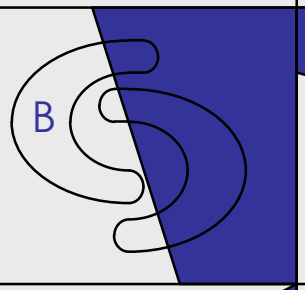
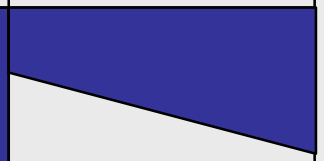
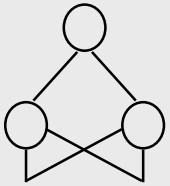
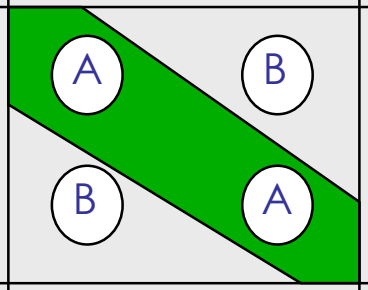
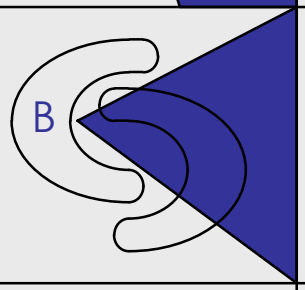
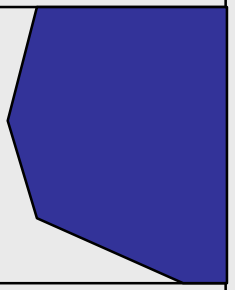
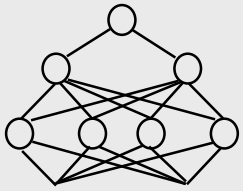
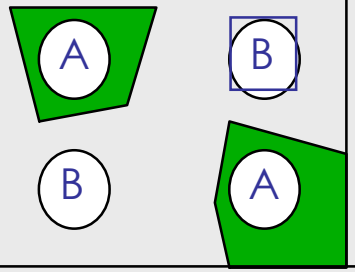
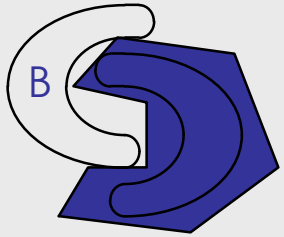
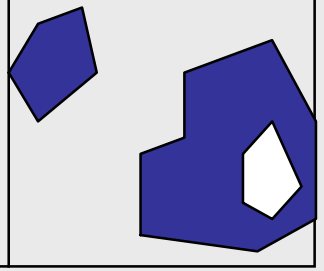
CV experts

1. Select / develop features: SURF, HoG, SIFT, RIFT, …
2. Add on top of this Machine Learning for multi-class recognition and train classifier



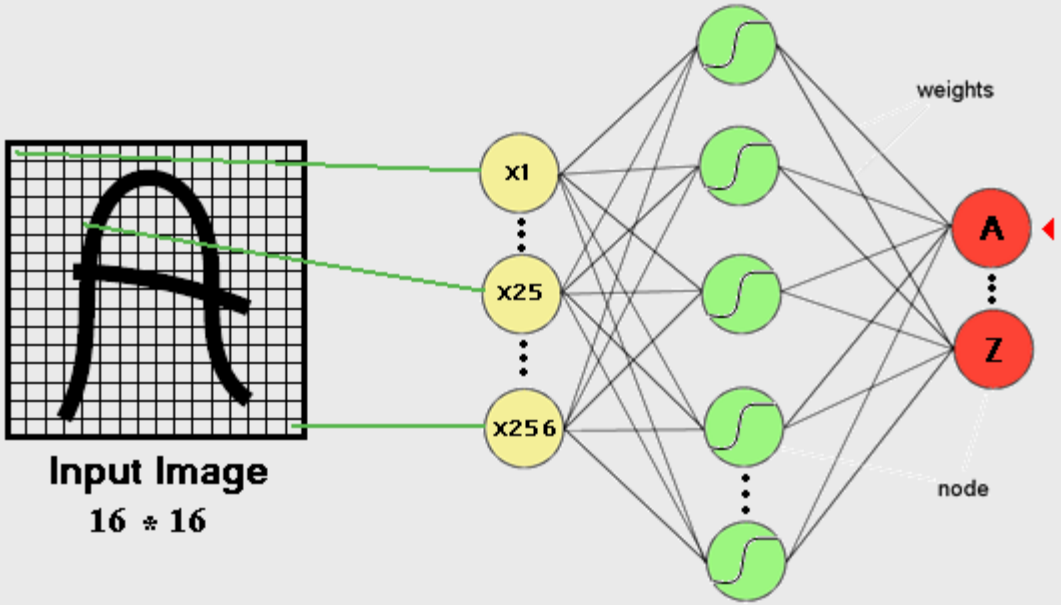
Classical CV feature definition is domain-specific and time-consuming

Behavior of multilayer neural networks

Structure	*Types of Decision Regions*	*Exclusive-OR Problem*	*Classes with Meshed regions*	*Most General Region Shapes*
Single-Layer 	*Half Plane Bounded By Hyper plane*			
Two-Layer 	*Convex Open Or Closed Regions*			
Three-Layer 	Arbitrary (Complexity Limited by No. of Nodes)			

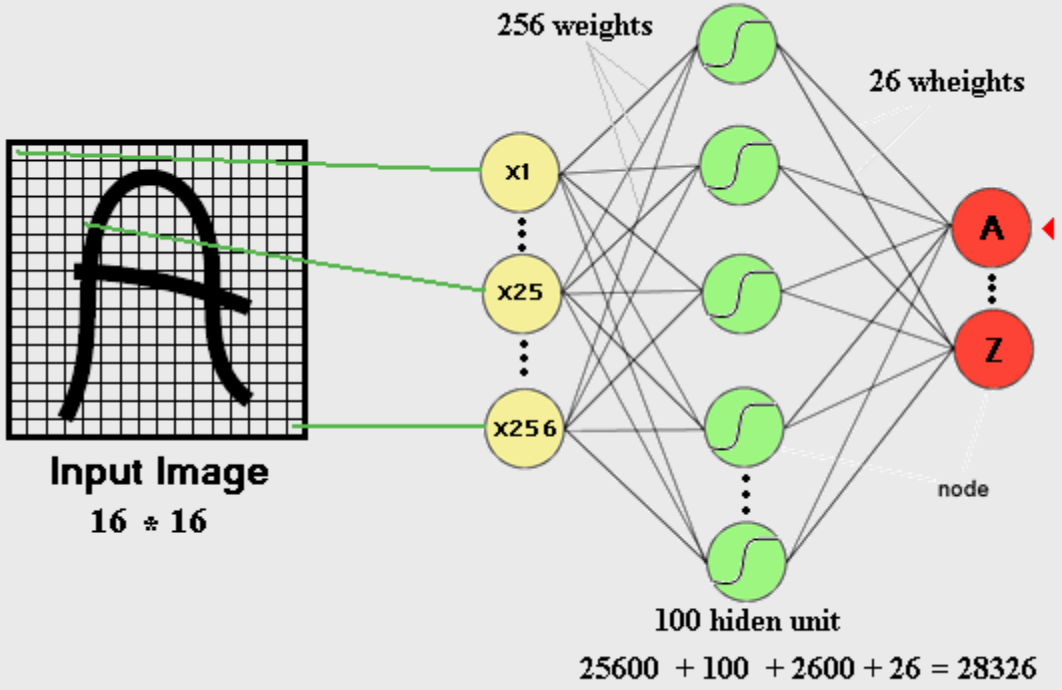
Multi-layer perceptron and image processing

- One or more hidden layers
- Sigmoid activations functions



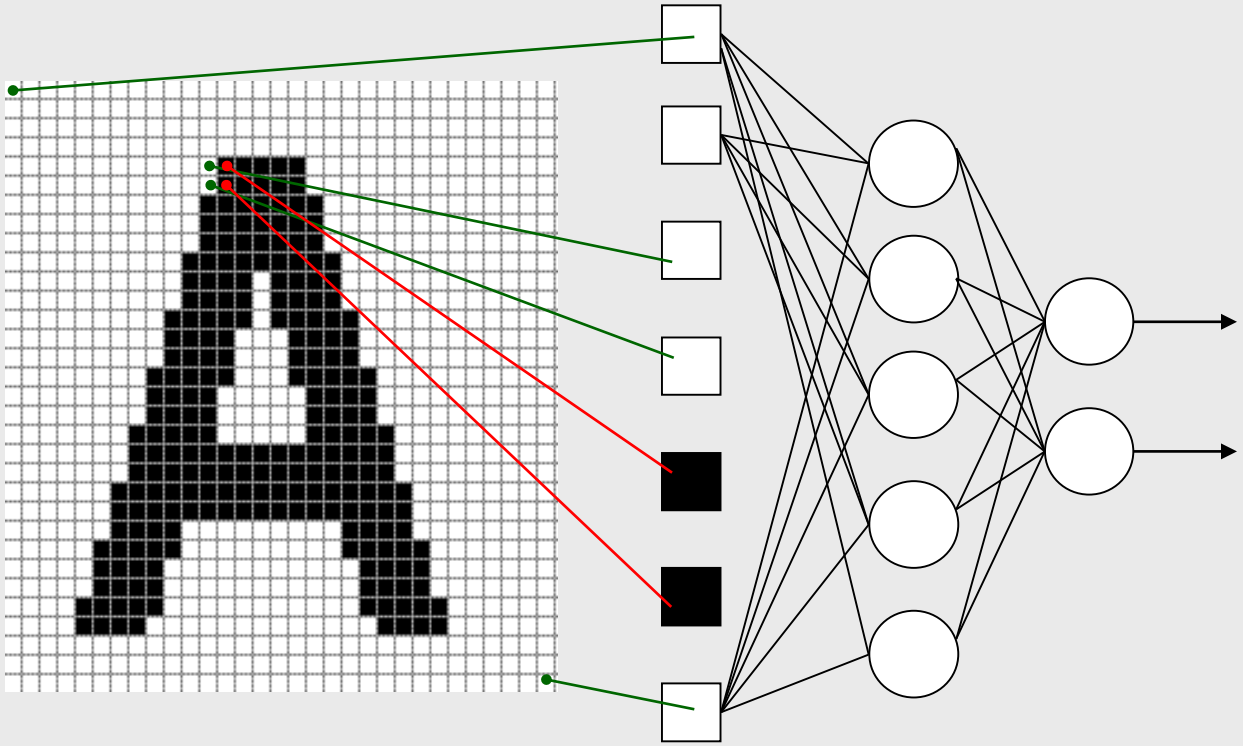
Drawbacks of previous neural networks

- the number of trainable parameters becomes extremely large



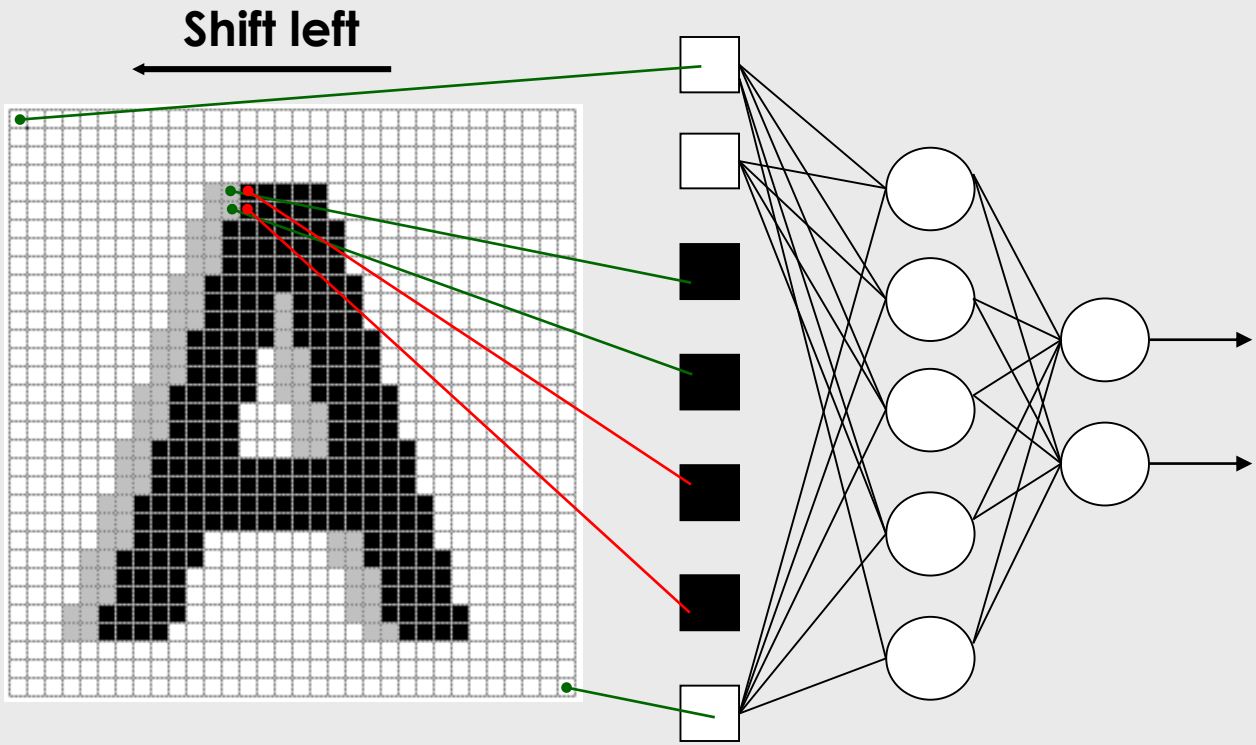
Drawbacks of previous neural networks

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

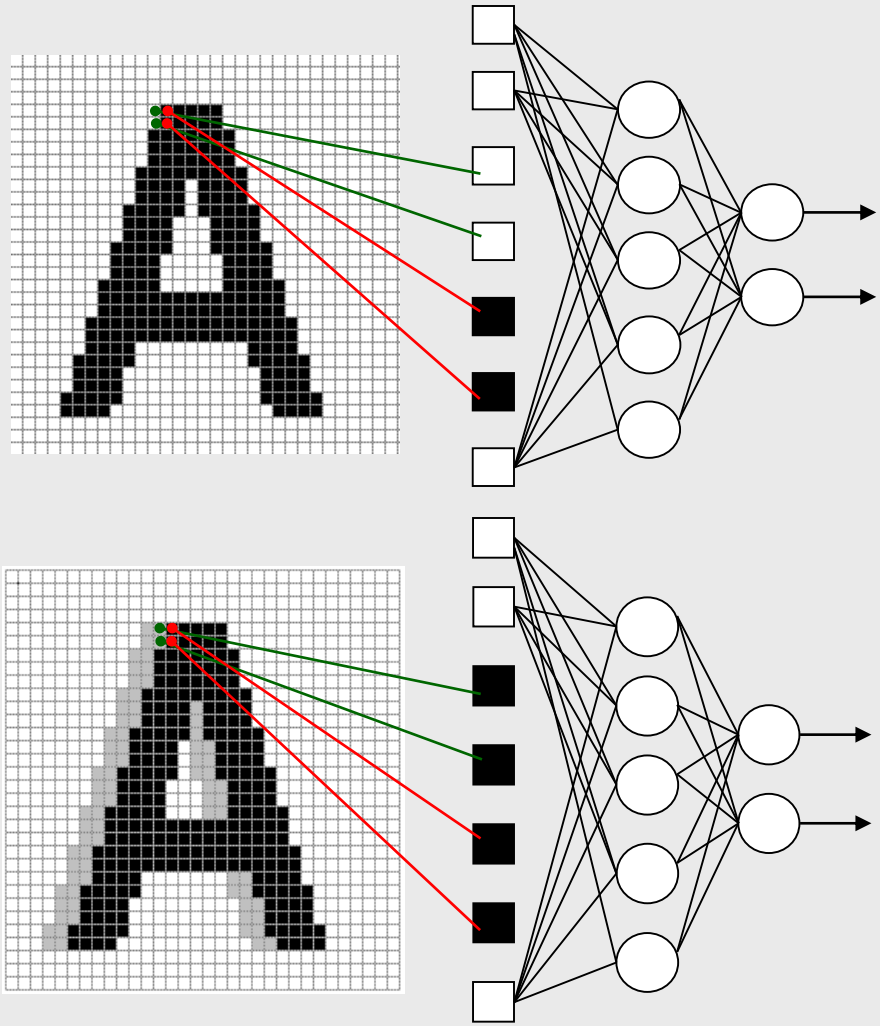


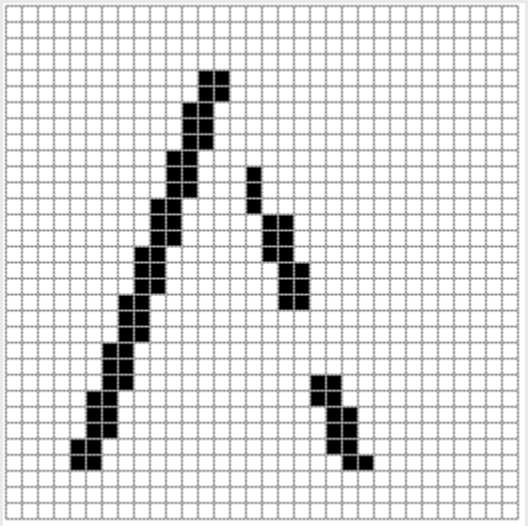
Drawbacks of previous neural networks

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



Drawbacks of previous neural networks





154 input chang
e from 2 shift lef
t

77 : black to whit
e

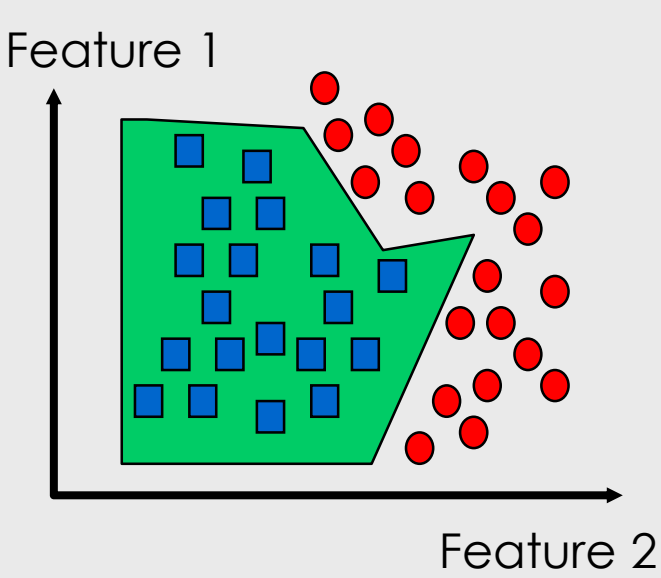
77 : white to bla

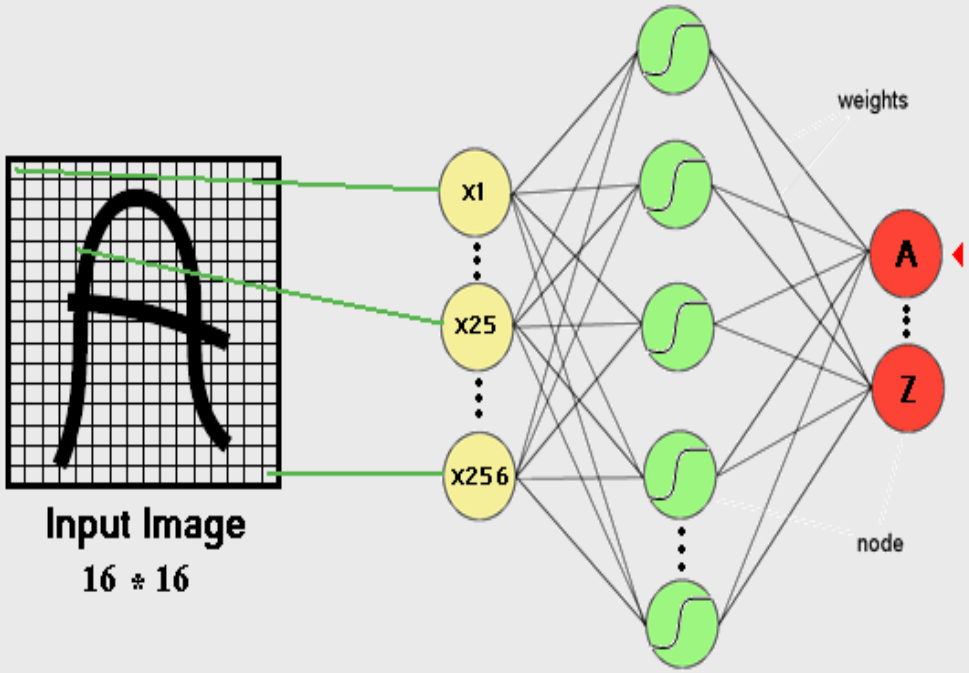
Drawbacks of previous neural networks

- scaling, and other forms of distortion

Drawbacks of previous neural networks

- the topology of the input data is completely ignored
- work with raw data.

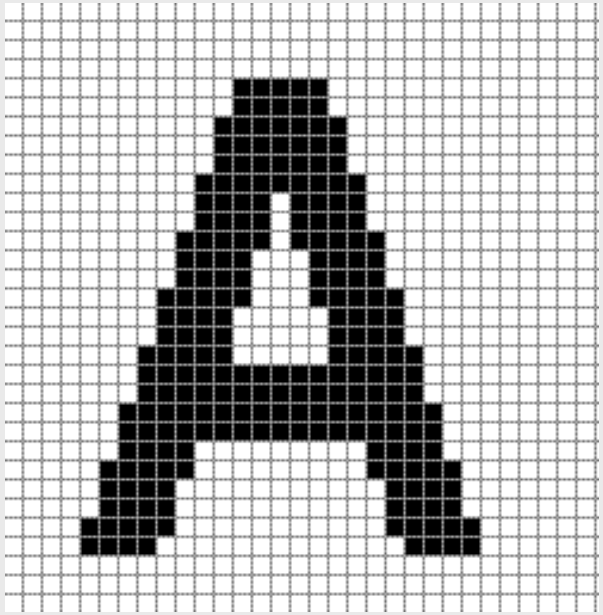




Drawbacks of previous neural networks

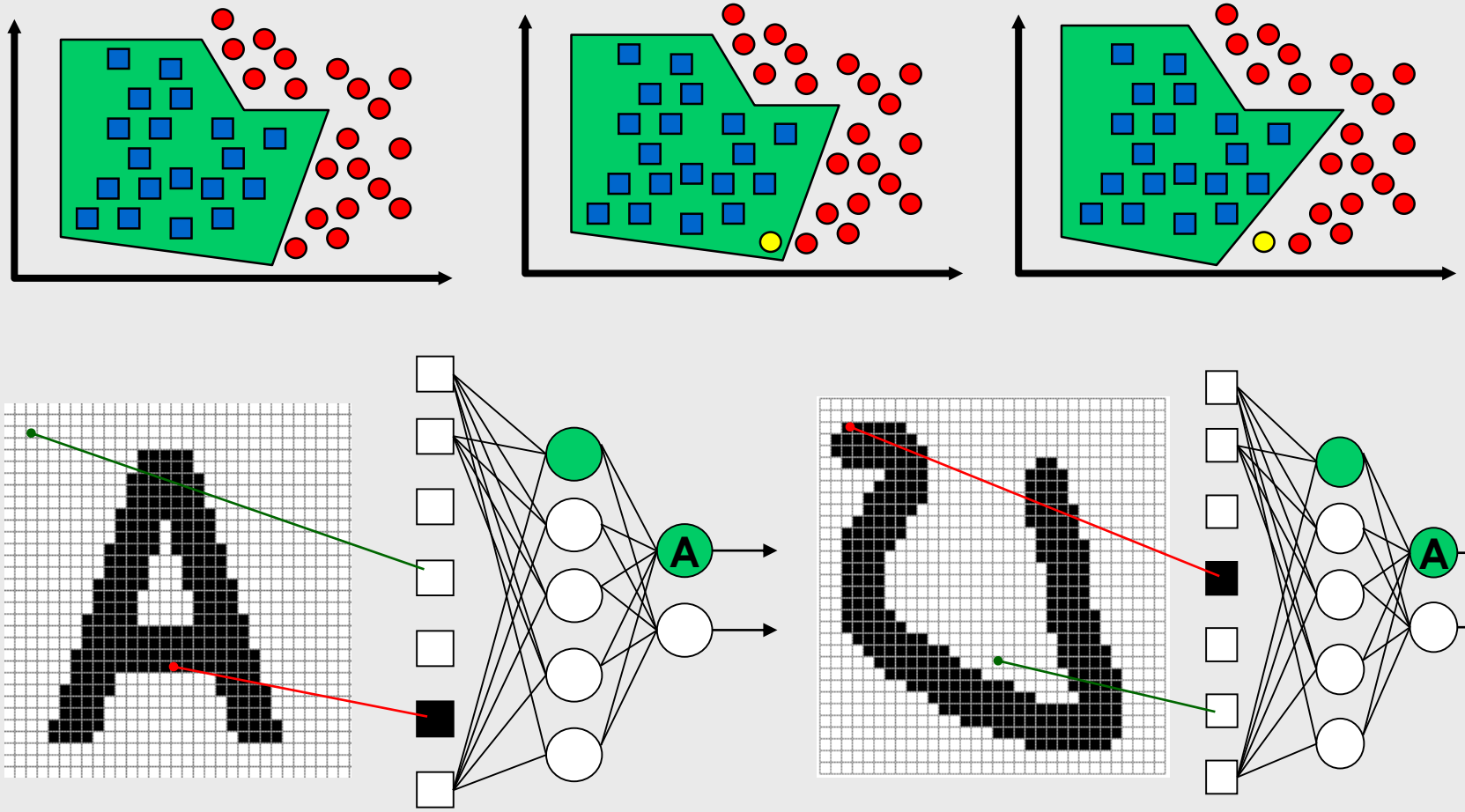
Black and white patterns: $2^{32*32} = 2^{1024}$

Gray scale patterns : $256^{32*32} = 256^{1024}$



32 * 32 input image

Drawbacks of previous neural networks



Improvement

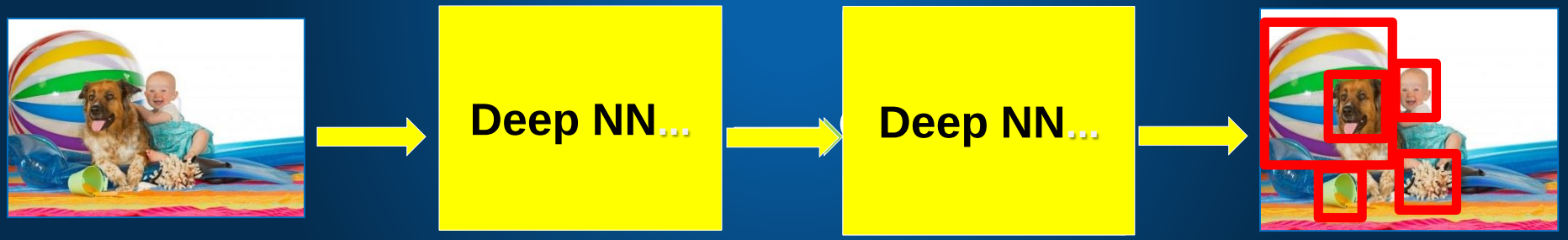
- Fully connected network of sufficient size can produce outputs that are invariant with respect to such variations.
  - Training time
  - Network size
  - Free parameters

Deep Learning –based Vision Pipeline.

Deep Learning:

- Build features automatically based on training data
- Combine feature extraction and classification

DL experts: define NN topology and train NN



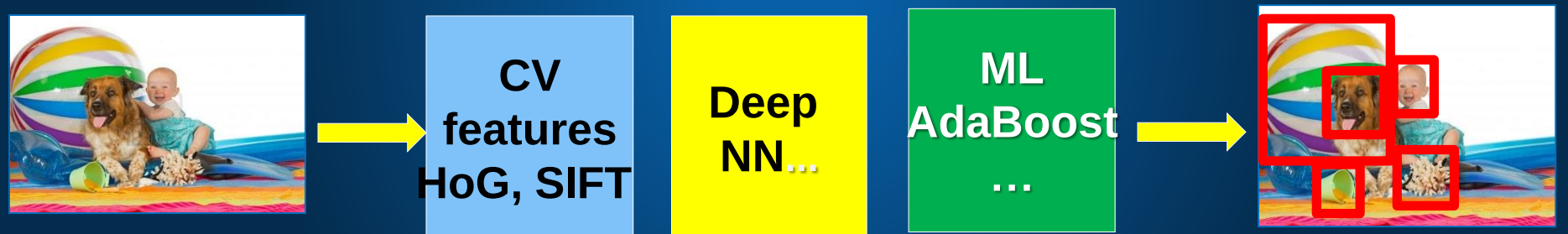
Deep Learning promise:
train good feature automatically,
same method for different domain

Computer Vision +Deep Learning + Machine Learning

We want to combine Deep Learning + CV + ML

- Combine pre-defined features with learned features;
- Use best ML methods for multi-class recognition

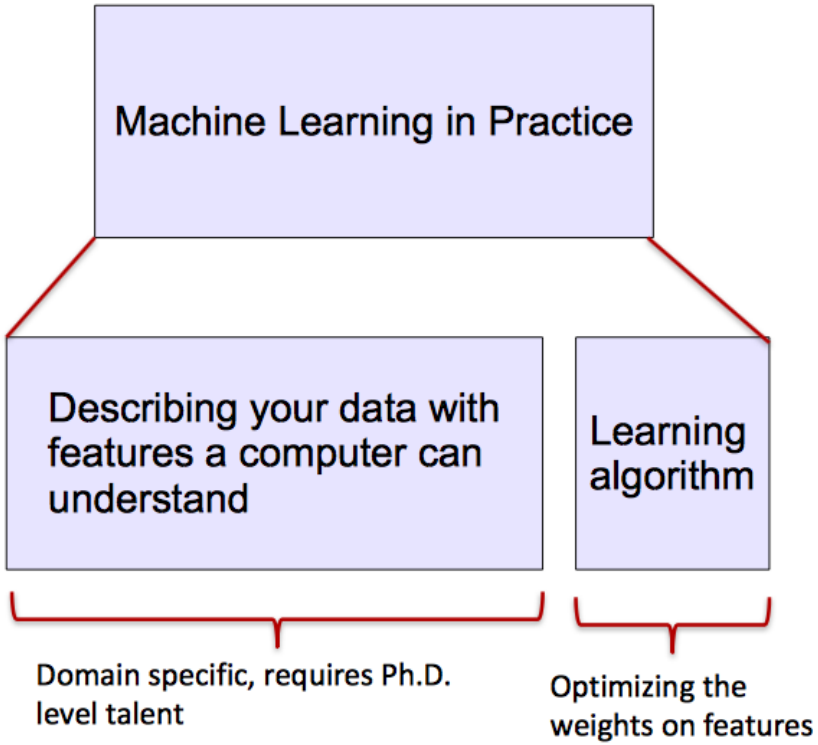
CV+DL+ML experts needed to build the best-in-class



Combine best of Computer Vision
Deep Learning and Machine Learning

ML vs. Deep Learning

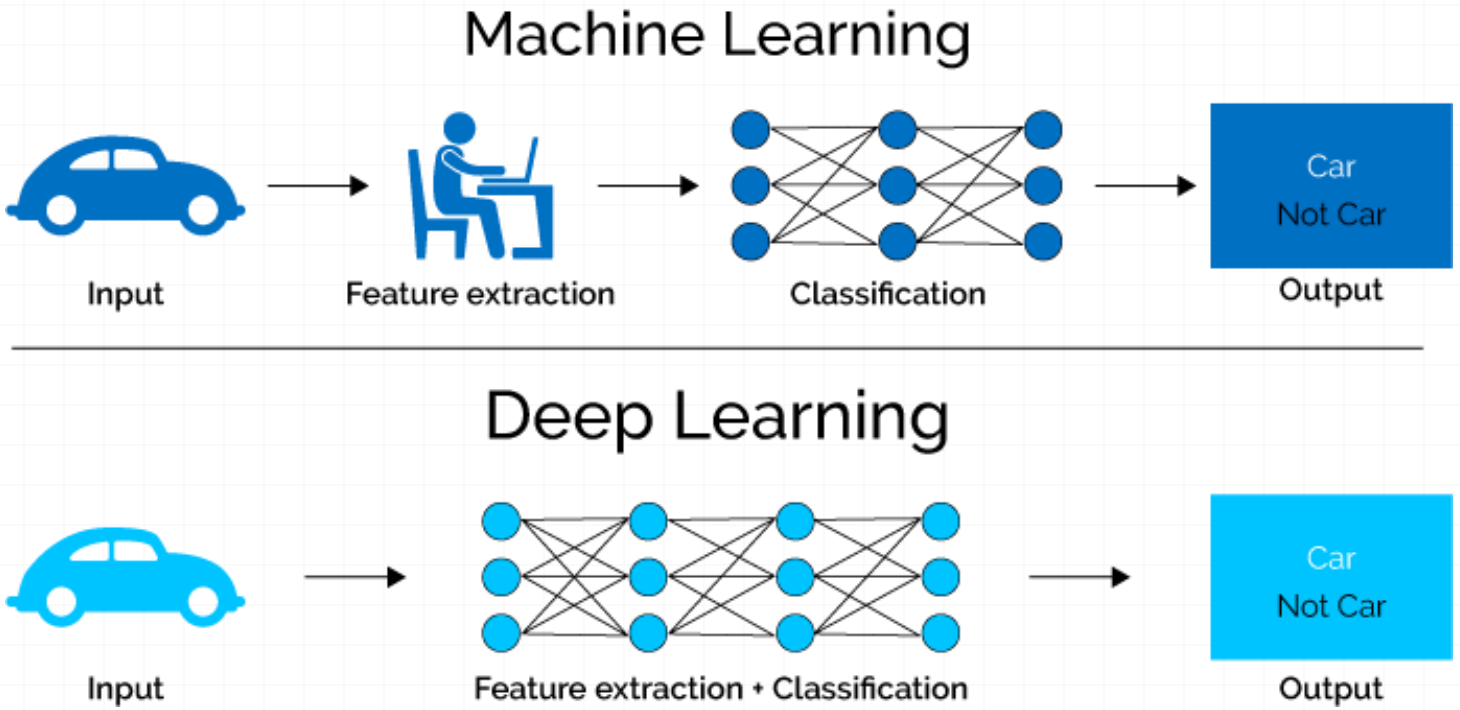
Most machine learning methods work well because of **human-designed representations** and **input features**
ML becomes just **optimizing weights** to best make a final prediction



Feature	NER
Current Word	✓
Previous Word	✓
Next Word	✓
Current Word Character n-gram	all
Current POS Tag	✓
Surrounding POS Tag Sequence	✓
Current Word Shape	✓
Surrounding Word Shape Sequence	✓
Presence of Word in Left Window	size 4
Presence of Word in Right Window	size 4

What is Deep Learning (DL) ?

A machine learning subfield of learning **representations** of data.
Exceptional effective at **learning patterns**.
Deep learning algorithms attempt to learn (multiple levels of) representation by using a **hierarchy of multiple layers**
If you provide the system **tons of information**, it begins to understand it and respond in useful ways.

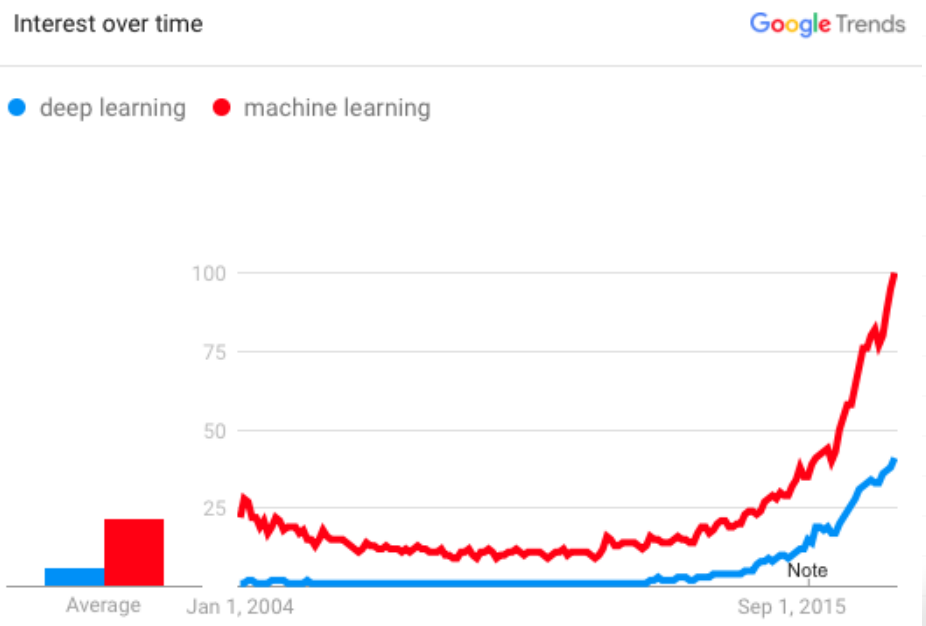


https://www.xenonstack.com/blog/static/public/uploads/media/machine-learning-vs-deep-learning.png

Why is DL useful?

- Manually designed features are often **over-specified**, **incomplete** and take a **long time to design** and validate
- Learned Features are **easy to adapt**, **fast** to learn
- Deep learning provides a very **flexible**, (almost?) **universal**, learnable framework for representing world, visual and linguistic information.
- Can learn both unsupervised and supervised
- Effective **end-to-end** joint system learning
- Utilize large amounts of training data

In ~2010 DL started outperforming other ML techniques
first in speech and vision, then NLP

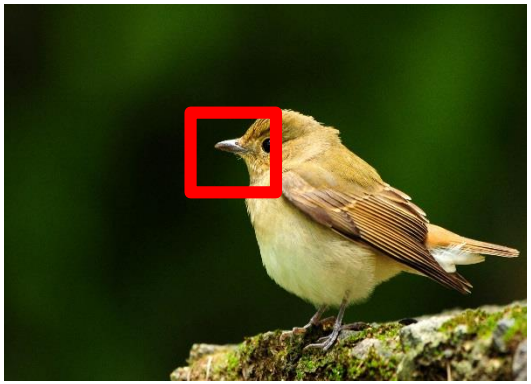


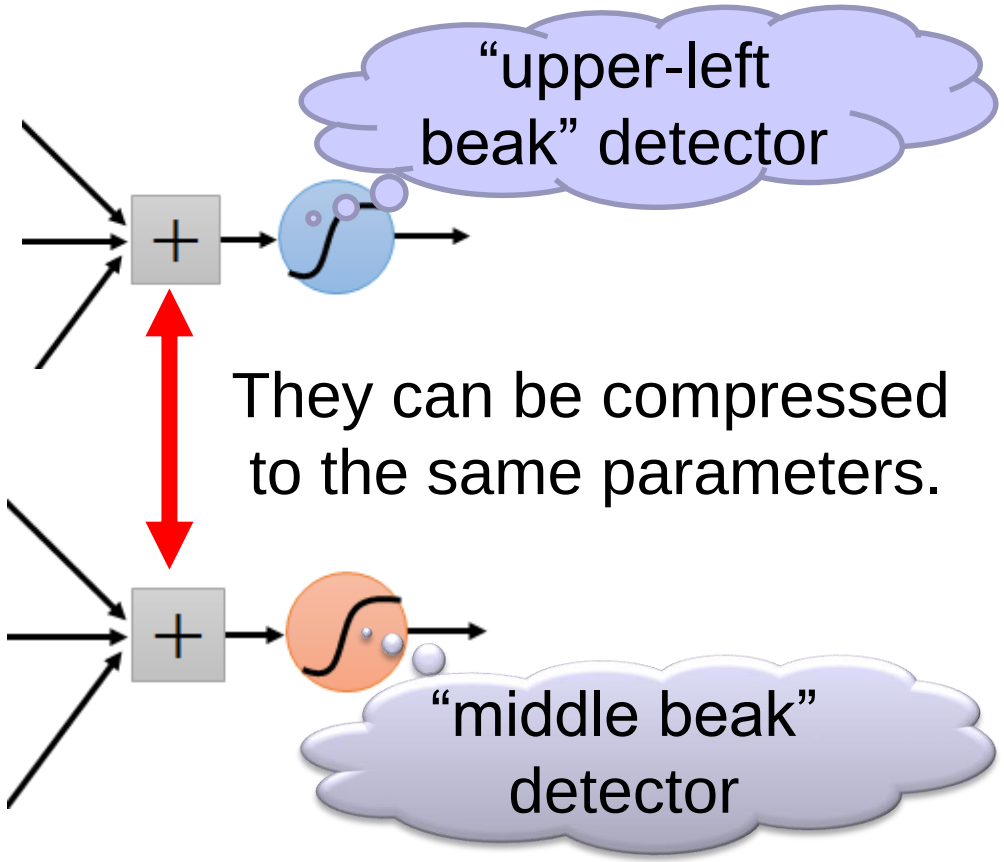
Consider learning an image:

- Some patterns are much smaller than the whole image

Can represent a small region with fewer parameters

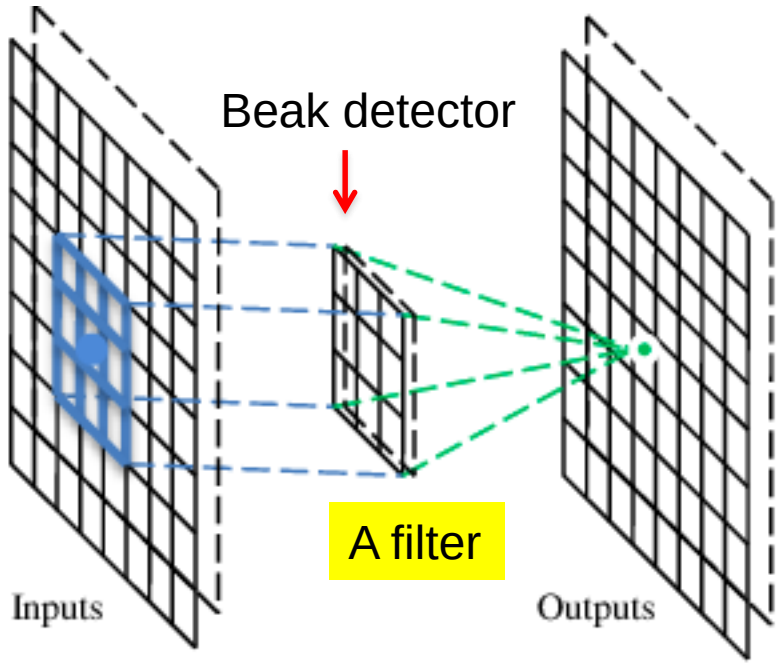
Same pattern appears in different places:
They can be compressed!
What about training a lot of such “small” detectors and each detector must “move around”.





A convolutional layer

A CNN is a neural network with some convolutional layers (and some other layers). A convolutional layer has a number of filters that does convolutional operation.



Convolution

These are the network parameters to be learned.

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

6 x 6 image

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

⋮ ⋮

Each filter detects a small pattern (3 x 3).

Convolution

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

stride=1

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

Dot product →

3 -1

6 x 6 image

Convolution

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

If stride=2

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

6 x 6 image

3 -3

Convolution

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

stride=1

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

6 x 6 image

3	-1	-3	-1
-3	1	0	-3
-3	-3	0	1
3	-2	-2	-1

Convolution

-1	1	-1
-1	1	-1
-1	1	-1

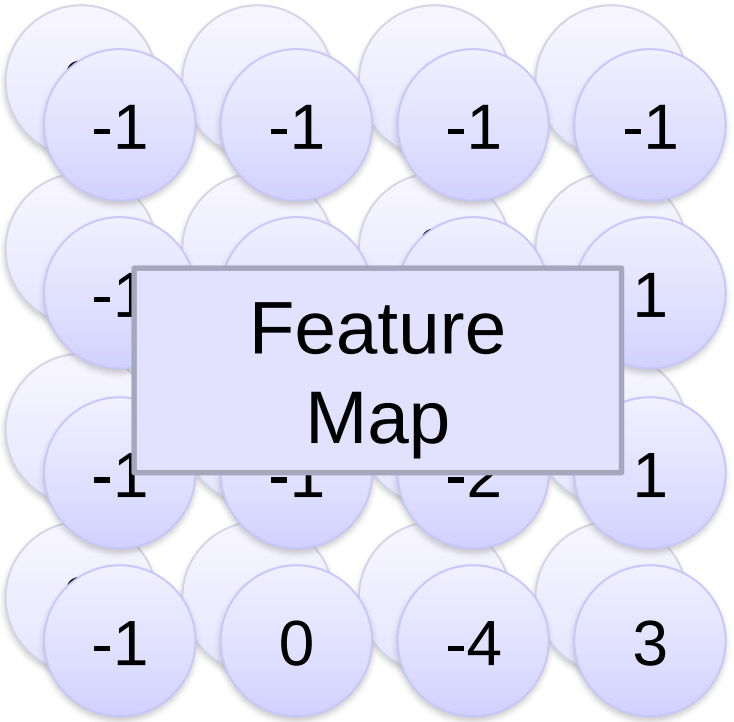
Filter 2

stride=1

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

6 x 6 image

Repeat this for each filter



Two 4 x 4 images
Forming 2 x 4 x 4 matrix

Color image: RGB 3 channels

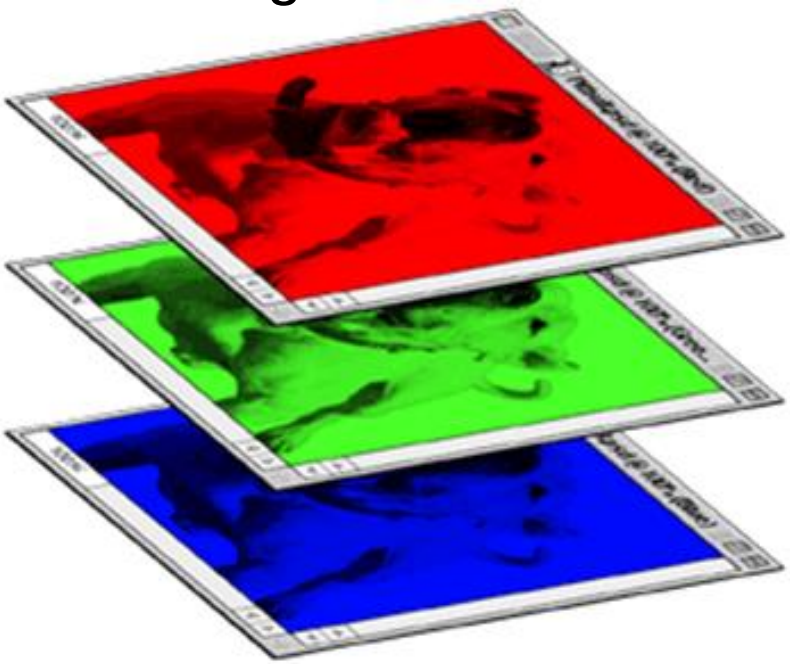
1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

Color image



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

Deep Learning Taxonomy

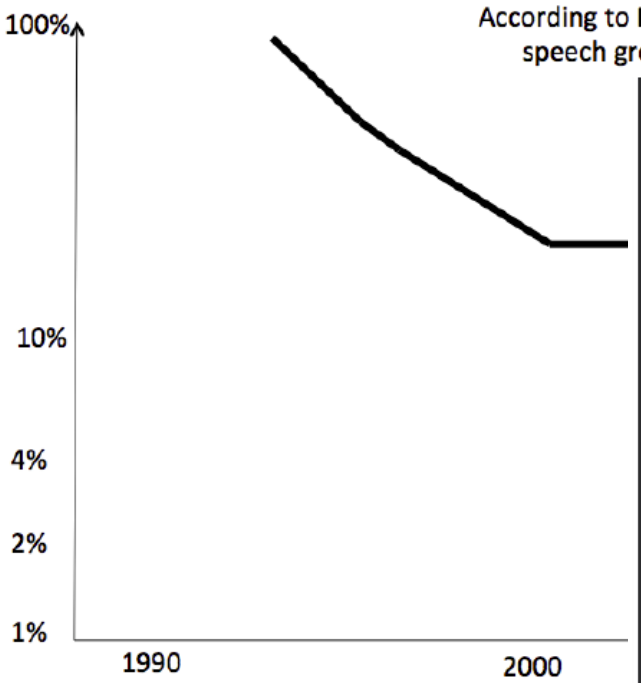
Supervised:

- Convolutional NN (LeCun)
- Recurrent Neural nets (Schmidhuber)

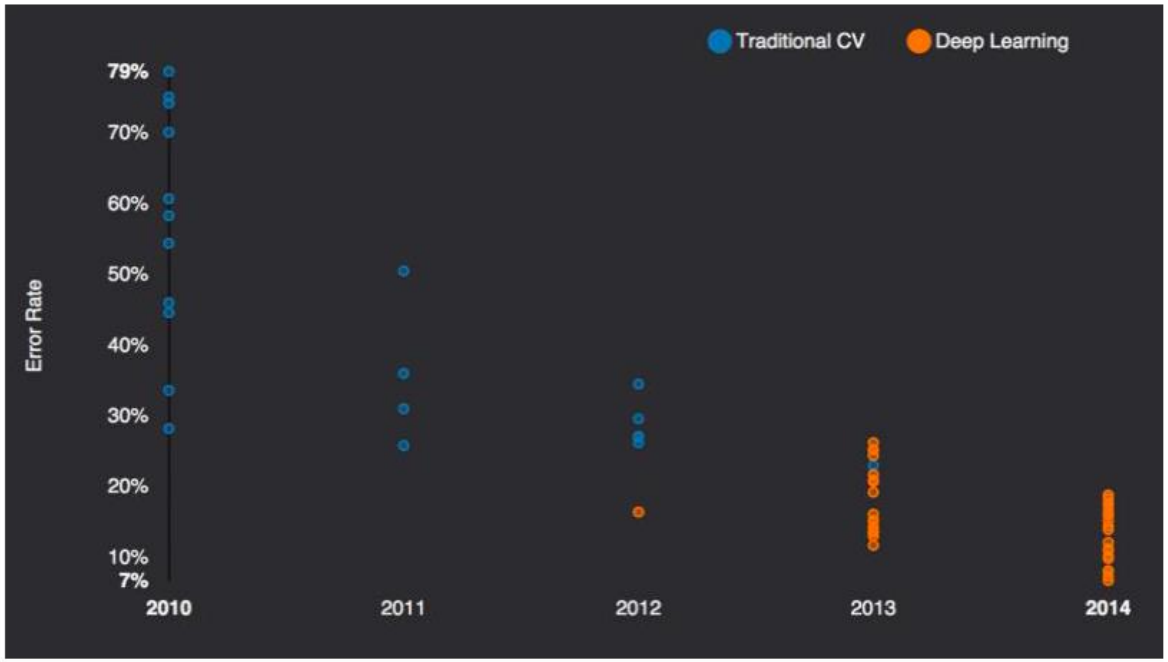
Unsupervised

- Deep Belief Nets / Stacked RBMs (Hinton)
- Stacked denoising autoencoders (Bengio)
- Sparse AutoEncoders (LeCun, A. Ng,)

State of the art in …



Deep Learning in Speech Recognition



ImageNet: The "computer vision World Cup"

Several big improvements in recent years in NLP

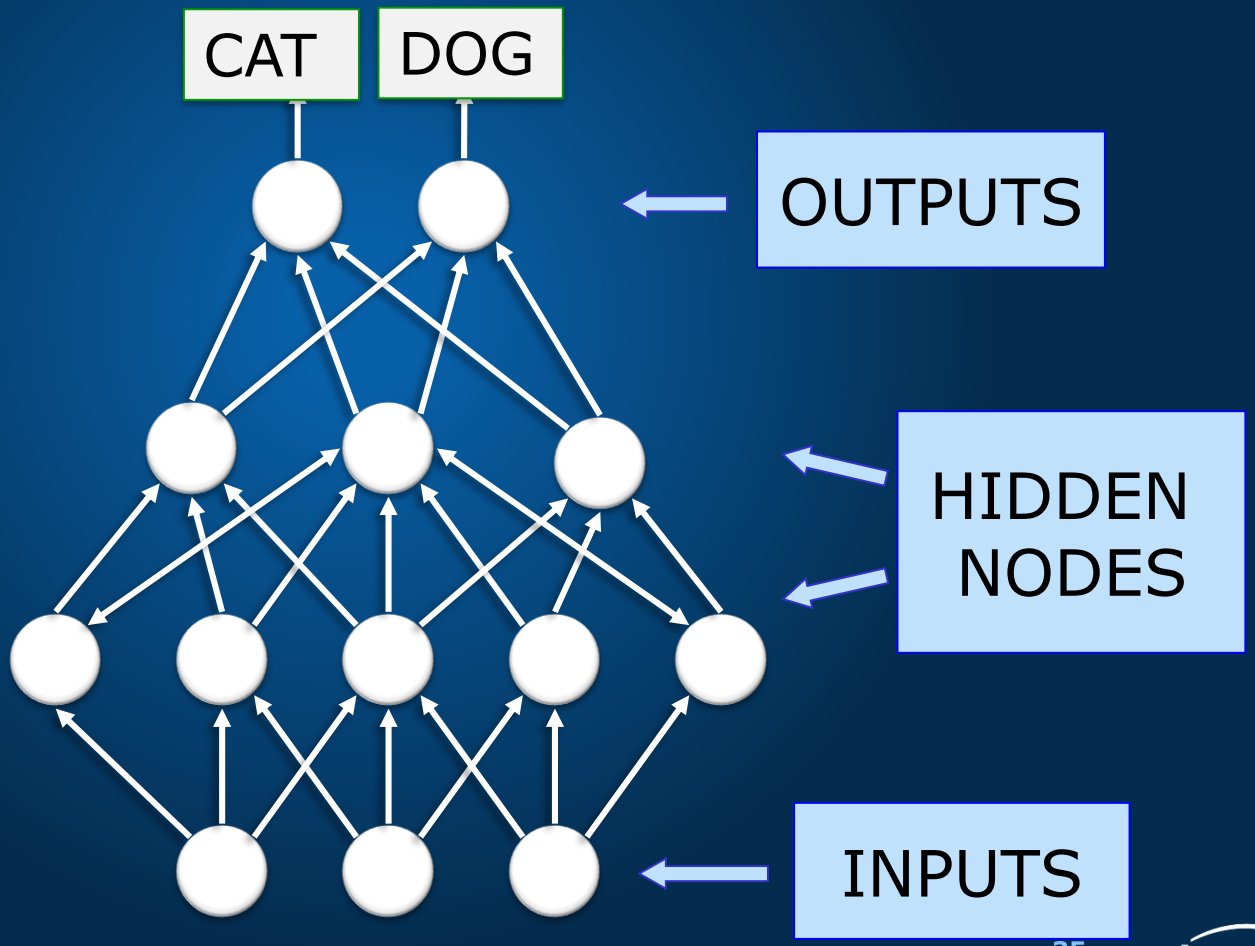
- ✓ Machine Translation
- ✓ Sentiment Analysis
- ✓ Dialogue Agents
- ✓ Question Answering
- ✓ Text Classification …

Leverage different levels of representation

- words & characters
- syntax & semantics

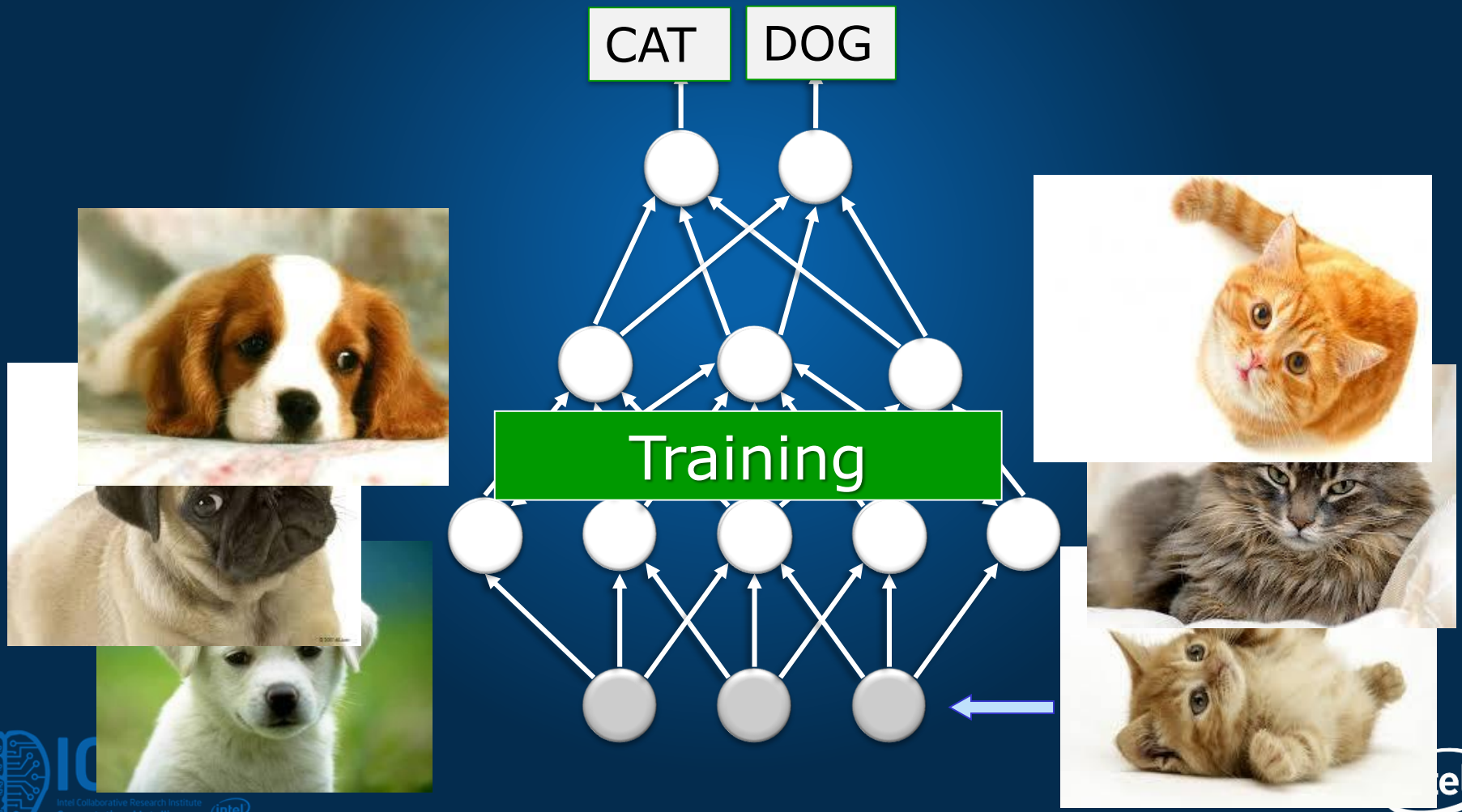
Deep Learning Basics

Deep Learning – is a set of machine learning algorithms based on multi-layer networks



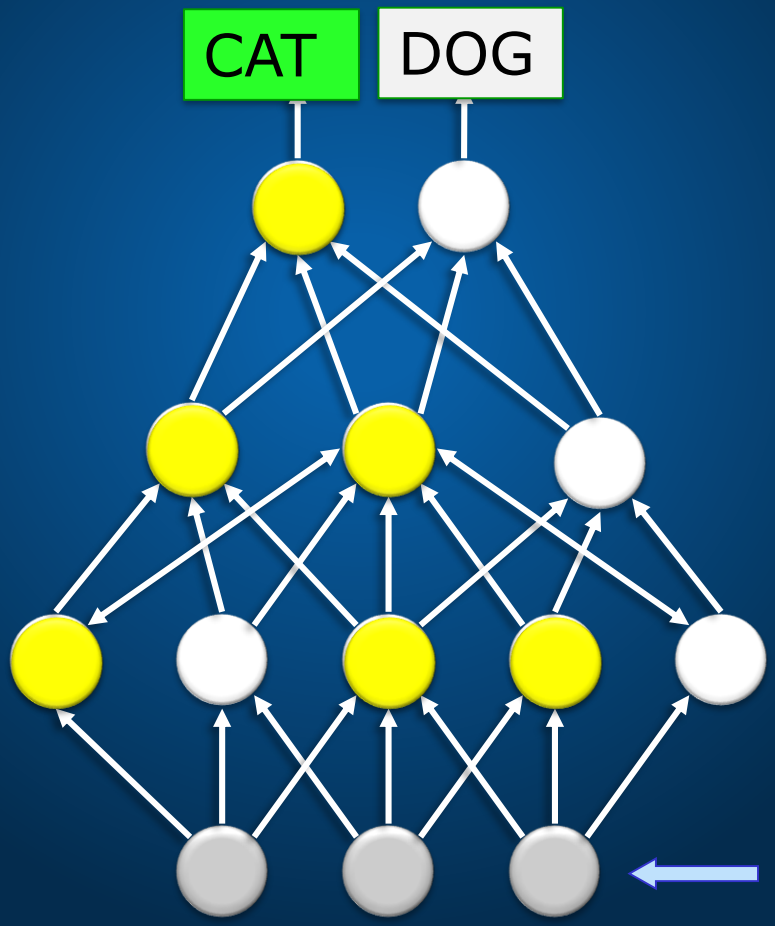
Deep Learning Basics

Deep Learning – is a set of machine learning algorithms based on multi-layer networks



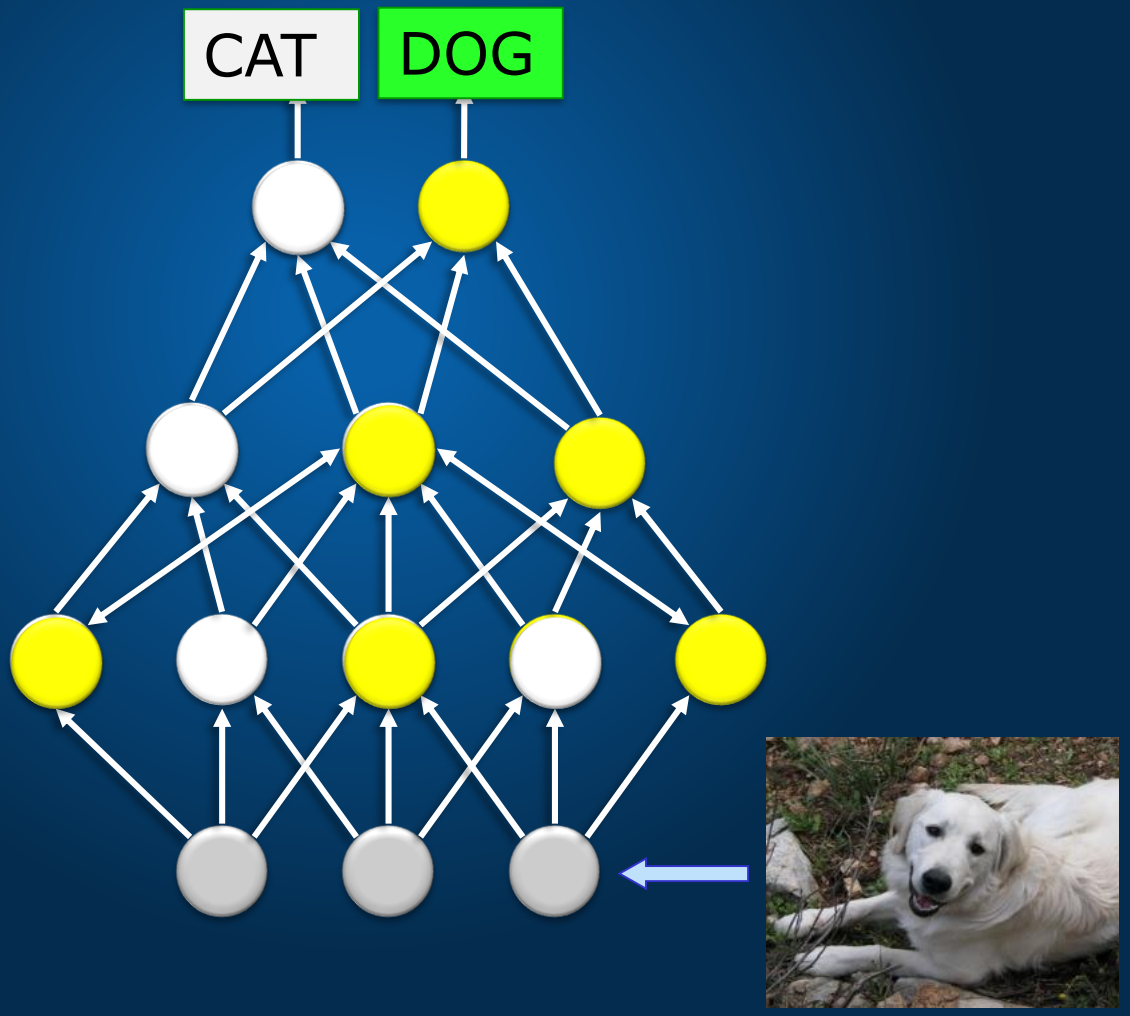
Deep Learning Basics

Deep Learning – is a set of machine learning algorithms based on multi-layer networks



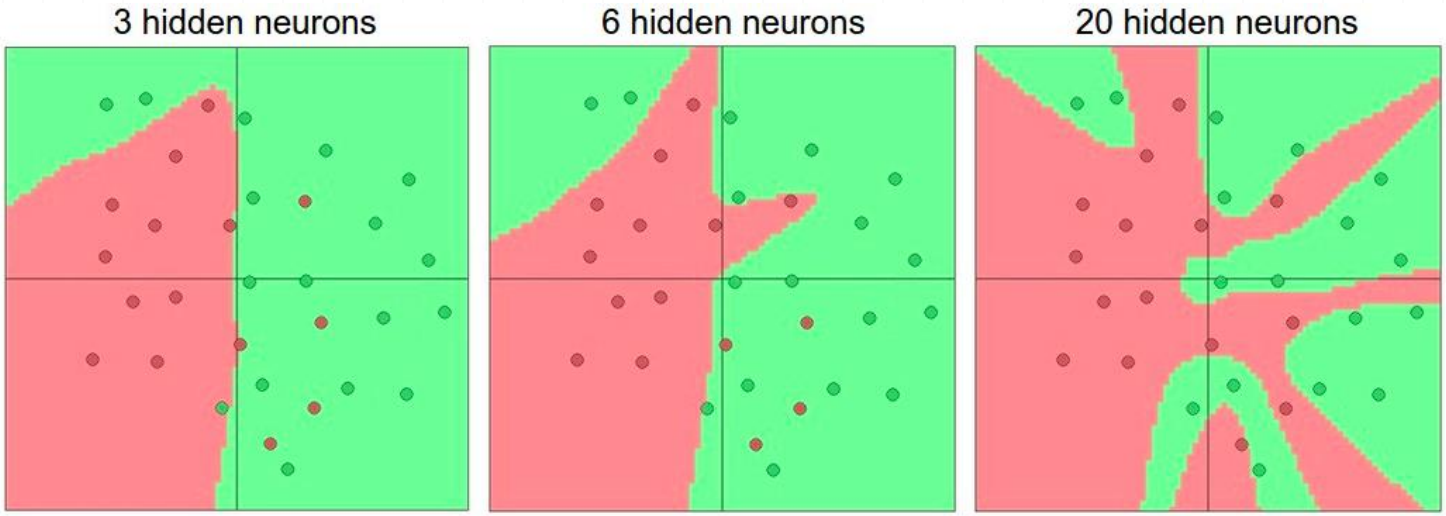
Deep Learning Basics

Deep Learning – is a set of machine learning algorithms based on multi-layer networks



Activation functions

Non-linearities needed to learn complex (non-linear) representations of data, otherwise the NN would be just a linear function $W_1 W_2 x = Wx$



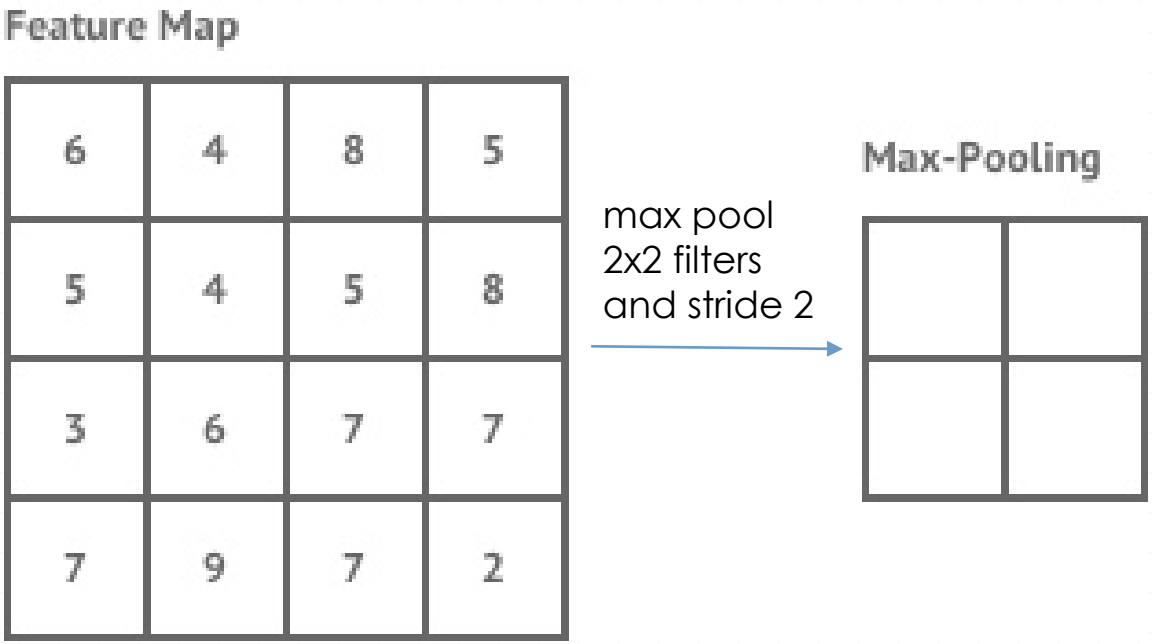
http://cs231n.github.io/assets/nn1/layer_sizes.jpeg

More layers and neurons can approximate more complex functions

Full list: https://en.wikipedia.org/wiki/Activation_function

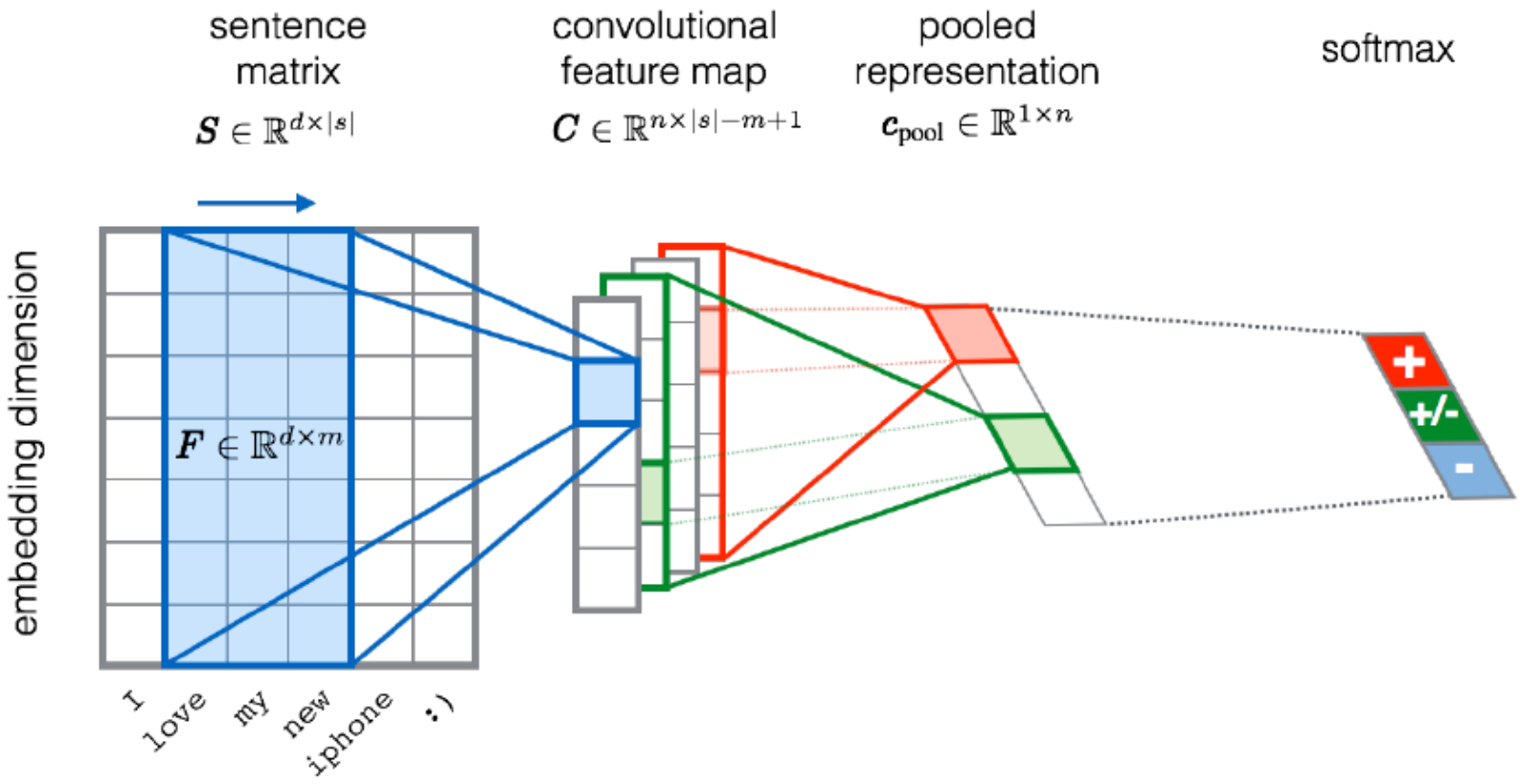
Convolutional Neural Networks (CNNs)

Main CNN idea for text:
Compute vectors for n-grams and **group them afterwards**



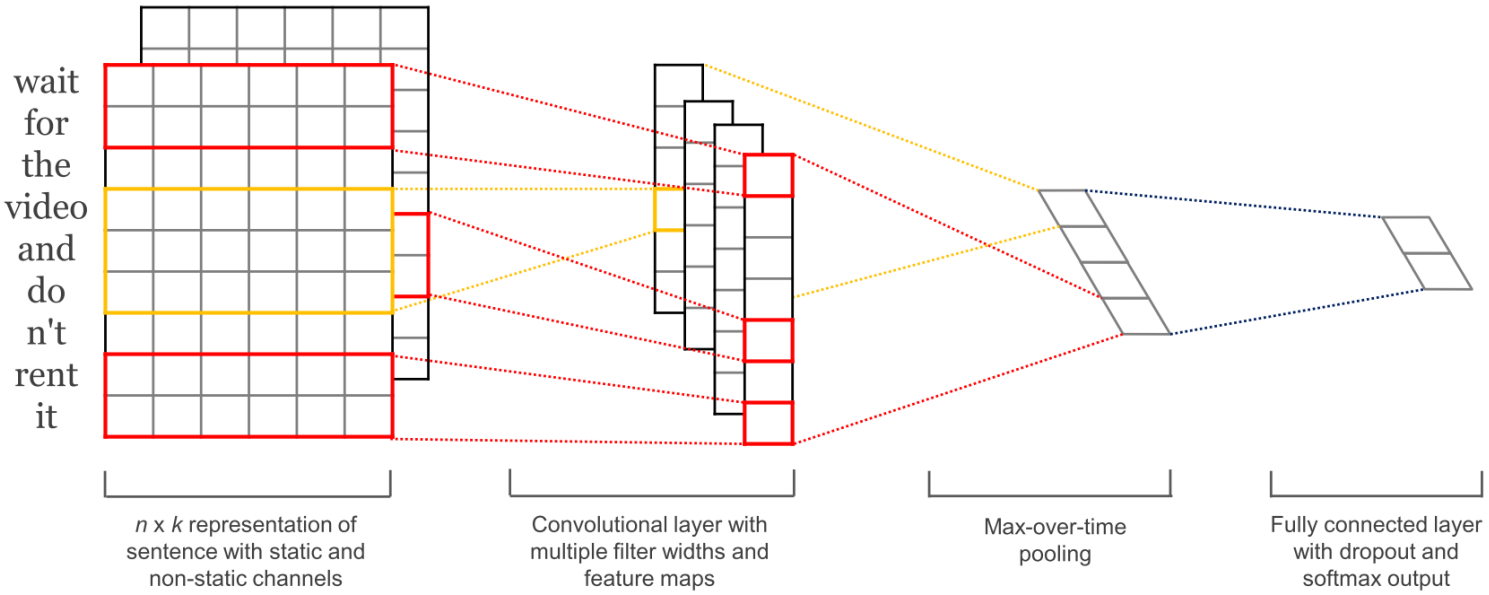
https://shafeentejani.github.io/assets/images/pooling.gif

CNN for text classification



Severyn, Aliaksei, and Alessandro Moschitti. "UNITN: Training Deep Convolutional Neural Network for Twitter Sentiment Classification." *SemEval@ NAACL-HLT*. 2015.

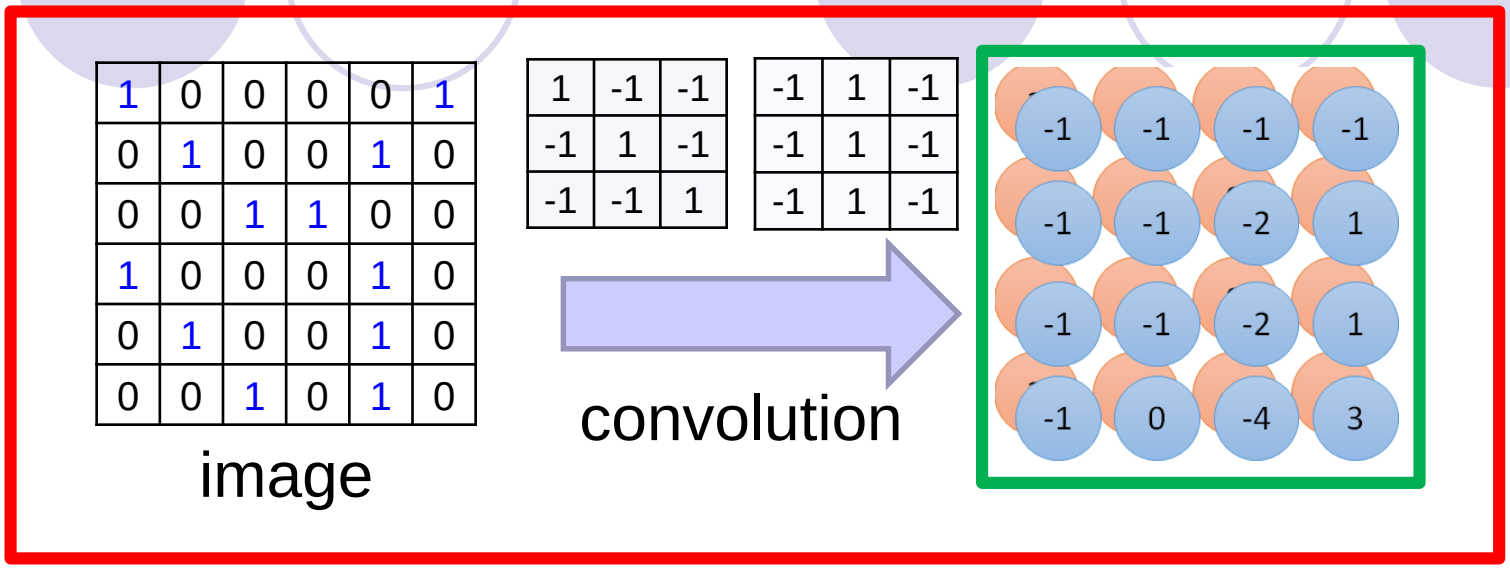
CNN with multiple filters



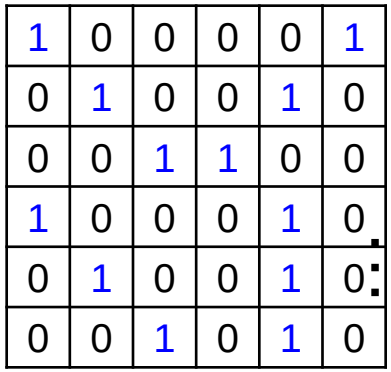
Kim, Y. "Convolutional Neural Networks for Sentence Classification", EMNLP (2014)

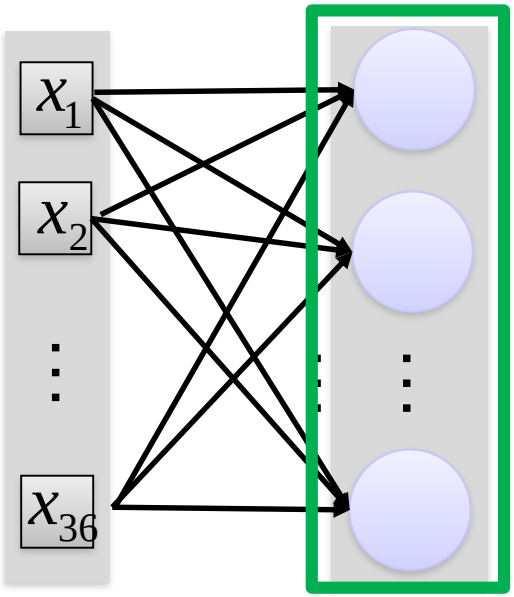
sliding over 3, 4 or 5 words at a time

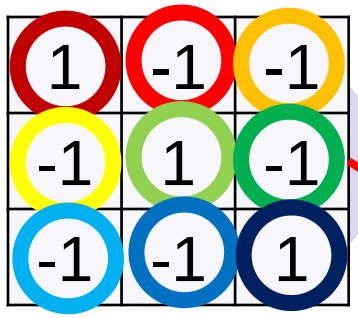
Convolution v.s. Fully Connected



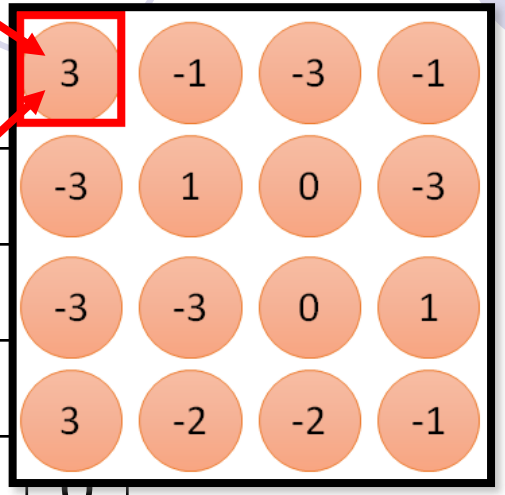
Fully-connected







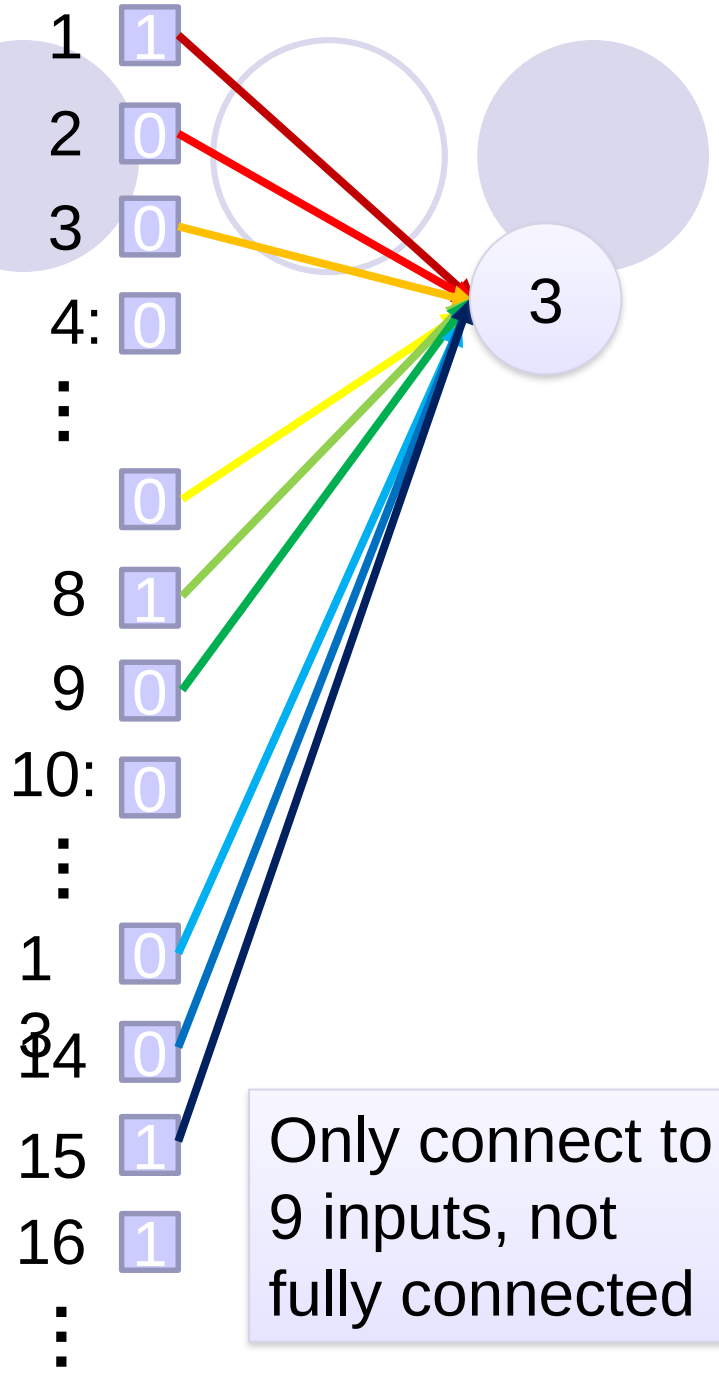
Filter 1



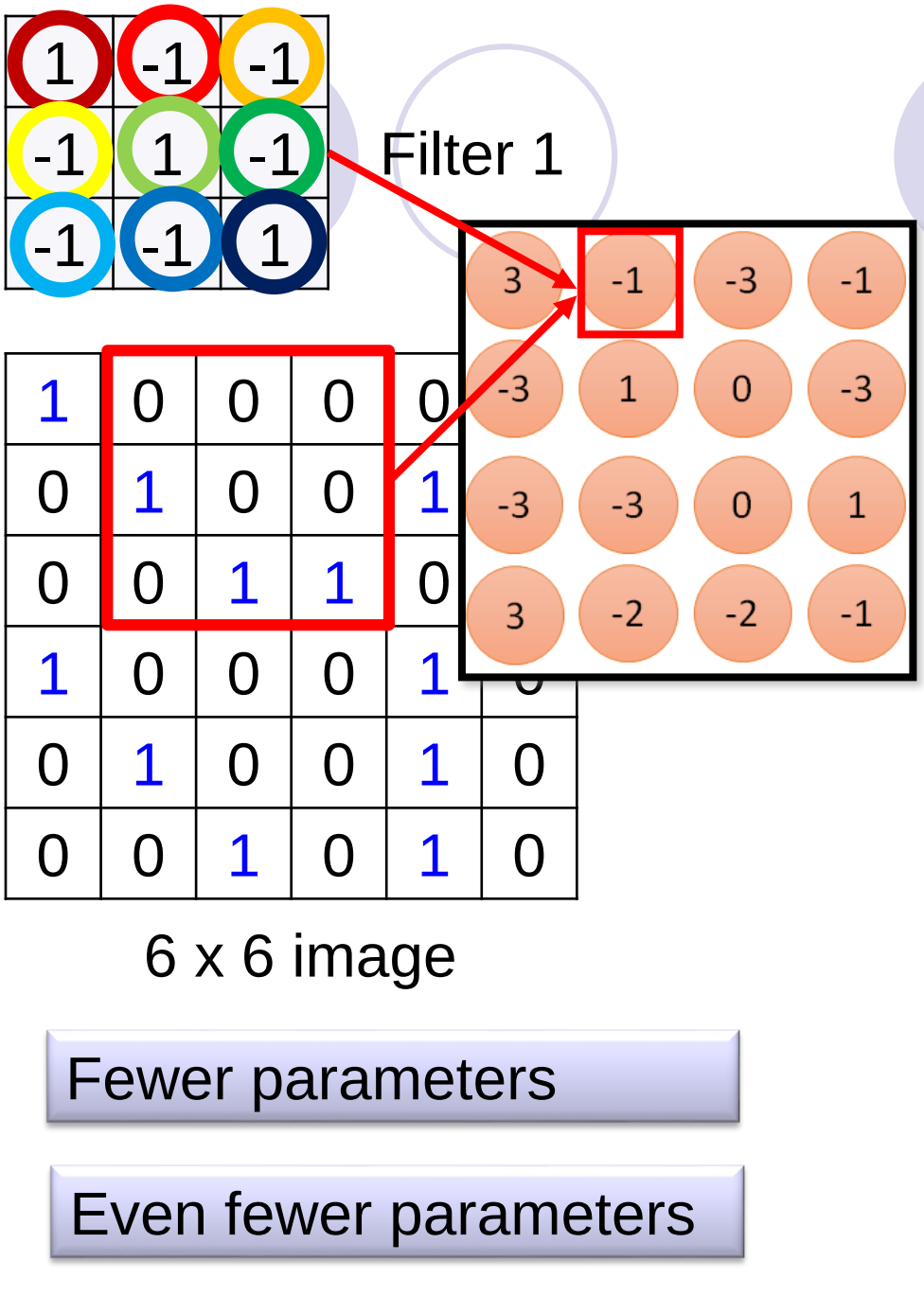
1	0	0	0	0	
0	1	0	0	1	
0	0	1	1	0	
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

fewer parameters!

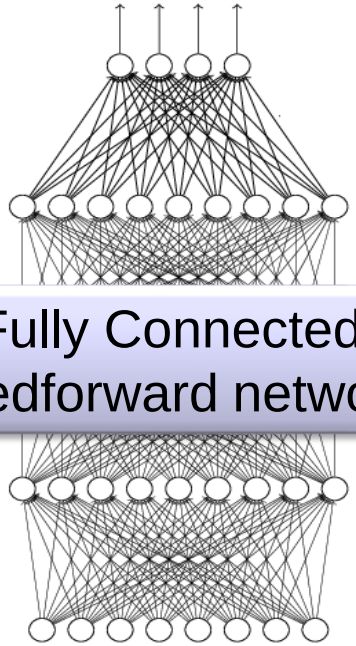


Only connect to 9 inputs, not fully connected


1 -1 -1
-1 1 -1
-1 -1 1
Filter 1
3 -1 -3 -1
-3 1 0 -3
-3 -3 0 1
3 -2 -2 -1
1 0 0 0 0
0 1 0 0 1
0 0 1 1 0
1 0 0 0 1
0 1 0 0 1 0
0 0 1 0 1 0
6 x 6 image
1:
2:
3:
4:
7:
8:
9:
10:
14:
15:
16:
3
-1
Fewer parameters
Even fewer parameters
Shared weights

The whole CNN

cat dog ……



Fully Connected Feedforward network

Convolution

Max Pooling

Convolution

Max Pooling

Can repeat many times

Flattened

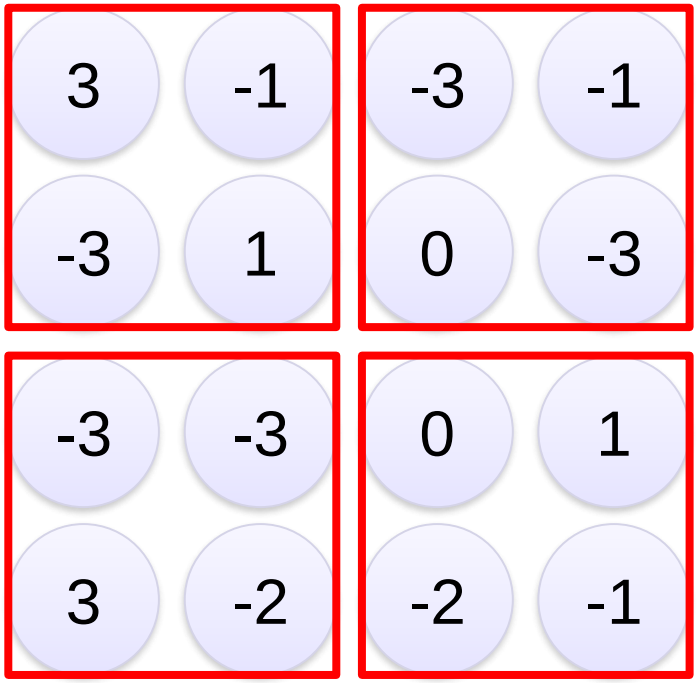
Max Pooling

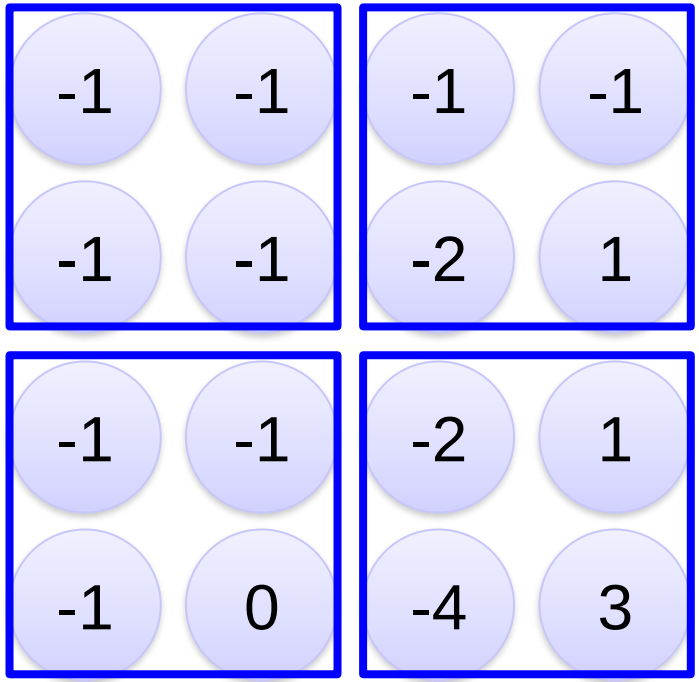
1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2





Why Pooling

- Subsampling pixels will not change the object

bird

Subsampling

bird

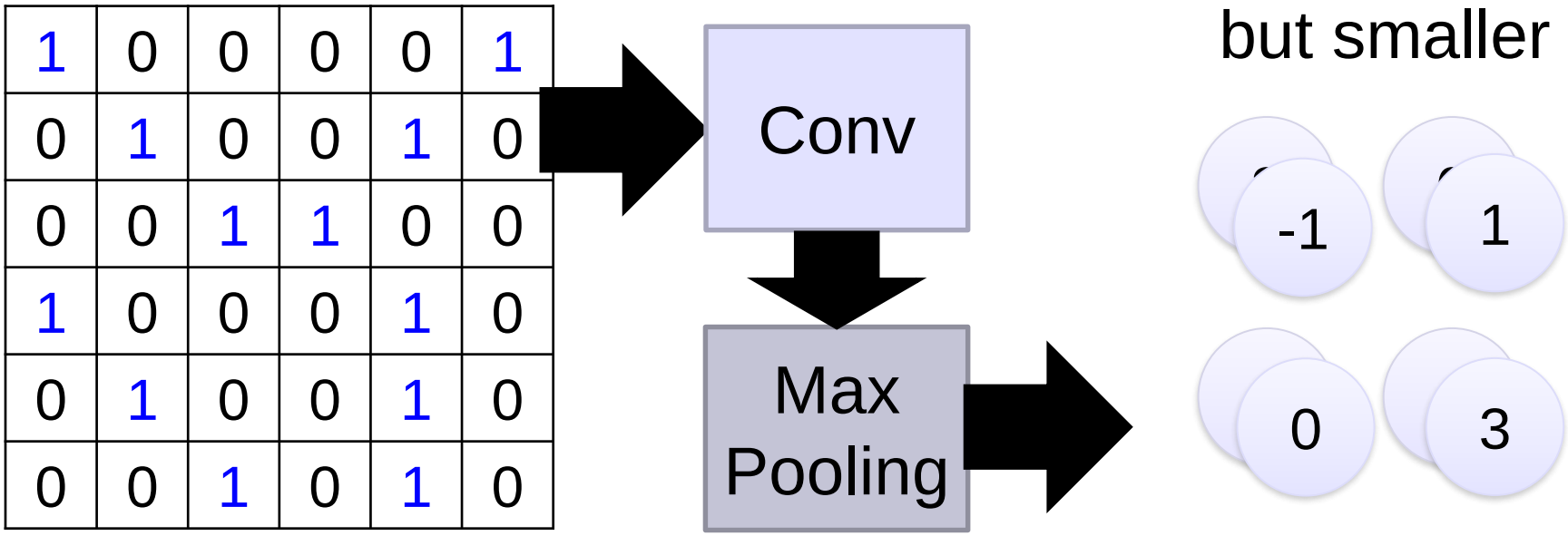
We can subsample the pixels to make image smaller

fewer parameters to characterize the image

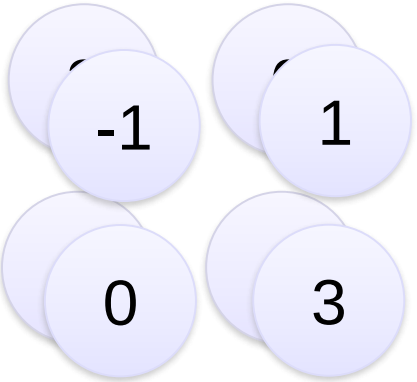
A CNN compresses a fully connected network in two ways:

- Reducing number of connections
- Shared weights on the edges
- Max pooling further reduces the complexity

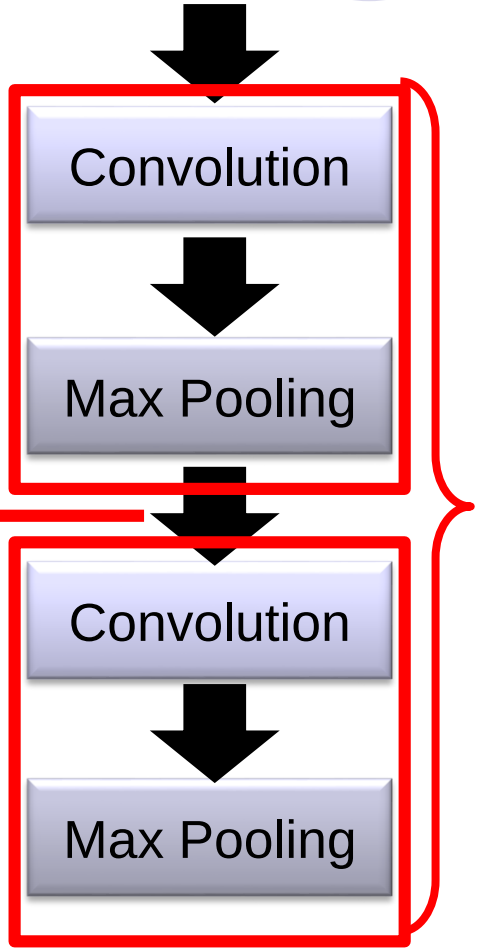
Max Pooling


1 0 0 0 0 1
0 1 0 0 1 0
0 0 1 1 0 0
1 0 0 0 1 0
0 1 0 0 1 0
0 0 1 0 1 0
Conv
Max Pooling
New image but smaller
-1
1
0
3
6 x 6 image
2 x 2 image
Each filter is a channel

The whole CNN



A new image



Can repeat many times

Smaller than the original image

The number of channels is the number of filters

The whole CNN

cat dog ……

Convolution

Max Pooling

A new image

Convolution

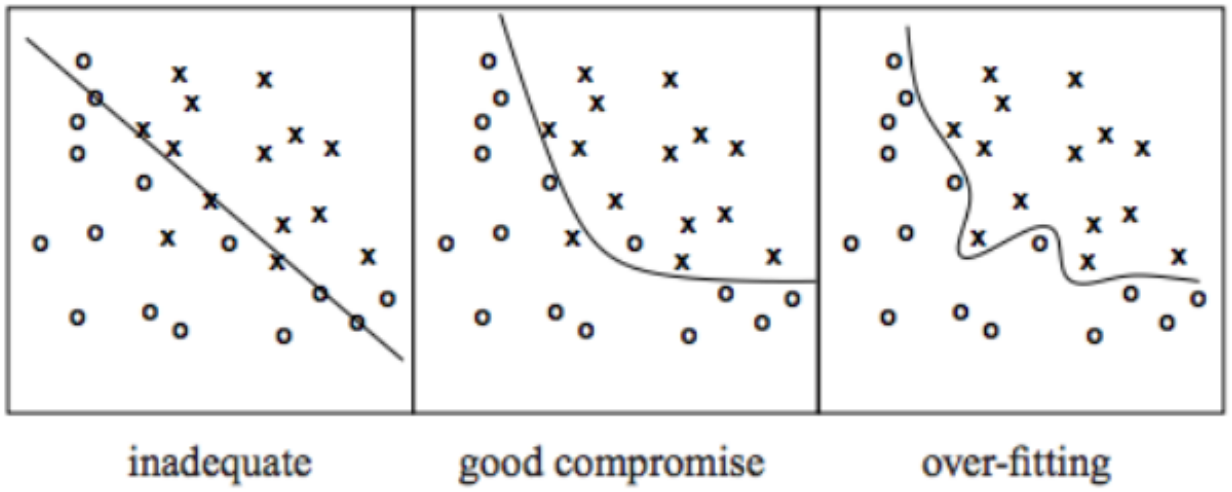
Max Pooling

A new image

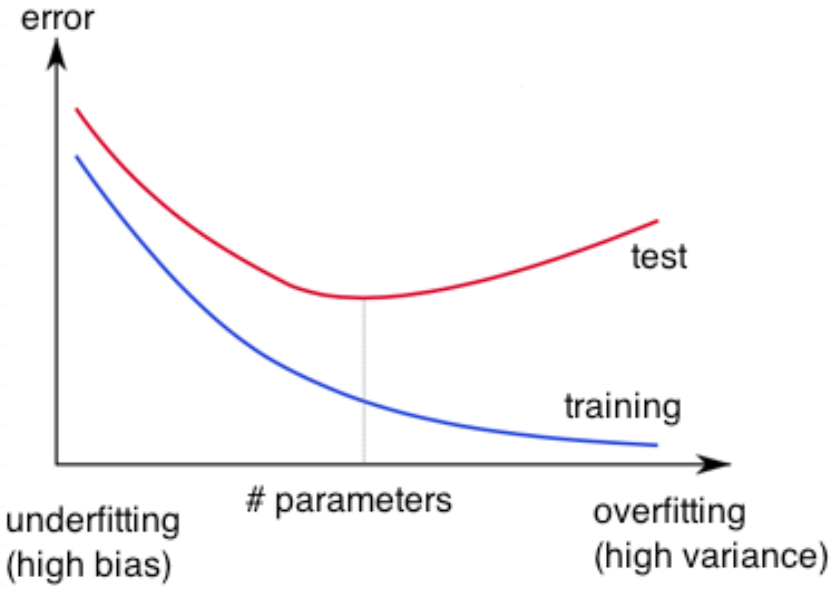
Flattened

Fully Connected Feedforward network

Overfitting



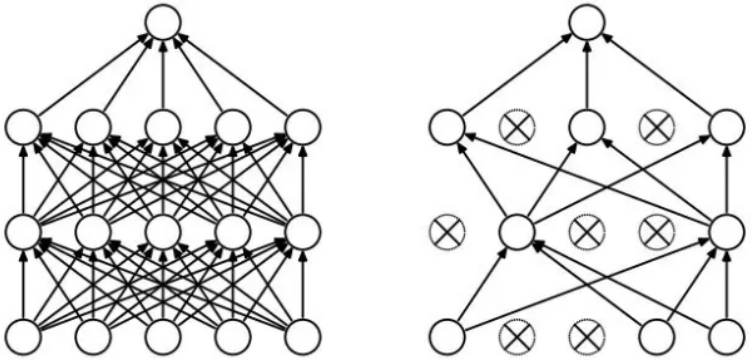
http://wiki.bethanycrane.com/overfitting-of-data



https://www.neuraldesigner.com/images/learning/selection_error.svg

Learned hypothesis may **fit** the training data very well, even outliers (**noise**) but fail to **generalize** to new examples (test data)

Regularization



Dropout

- Randomly drop units (along with their connections) during training
- Each unit retained with fixed probability p, independent of other units
- Hyper-parameter p to be chosen (tuned)

Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." Journal of machine learning research (2014)

L2 = weight decay

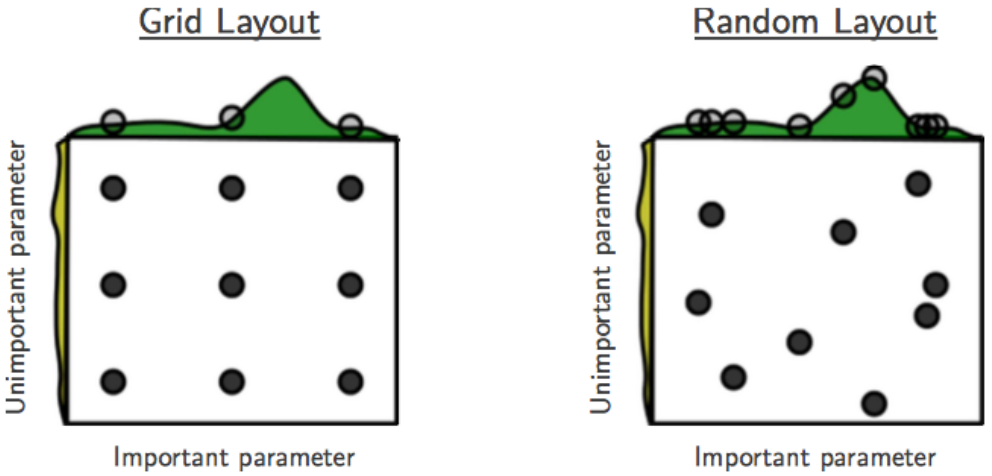
- Regularization term that penalizes big weights, added to the objective
- Weight decay value determines how dominant regularization is during gradient computation
- Big weight decay coefficient → big penalty for big weights

$$J_{reg}(\theta) = J(\theta) + \lambda \sum_{k} \theta_k^2$$

Early-stopping

- Use validation error to decide when to stop training
- Stop when monitored quantity has not improved after n subsequent epochs
- n is called patience

Tuning hyper-parameters



$g(x) \approx g(x) + h(y)$

g(x) shown in green
h(y) is shown in yellow

Bergstra, James, and Yoshua Bengio. "Random search for hyper-parameter optimization." Journal of Machine Learning Research, Feb (2012)

"Grid and random search of 9 trials for optimizing function $g(x) \approx g(x) + h(y)$
With grid search, nine trials only test g(x) in three distinct places.
With random search, all nine trials explore distinct values of g. "

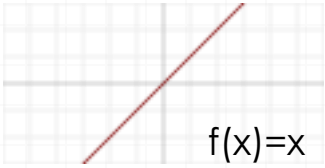
Both try configurations randomly and **blindly**
Next trial is independent to all the trials done before

Bayesian optimization for hyper-parameter tuning: Library available!

Make smarter choice for the next trial, minimize the number of trials

1. Collect the performance at several configurations
2. Make inference and decide what configuration to try next

Loss functions and output

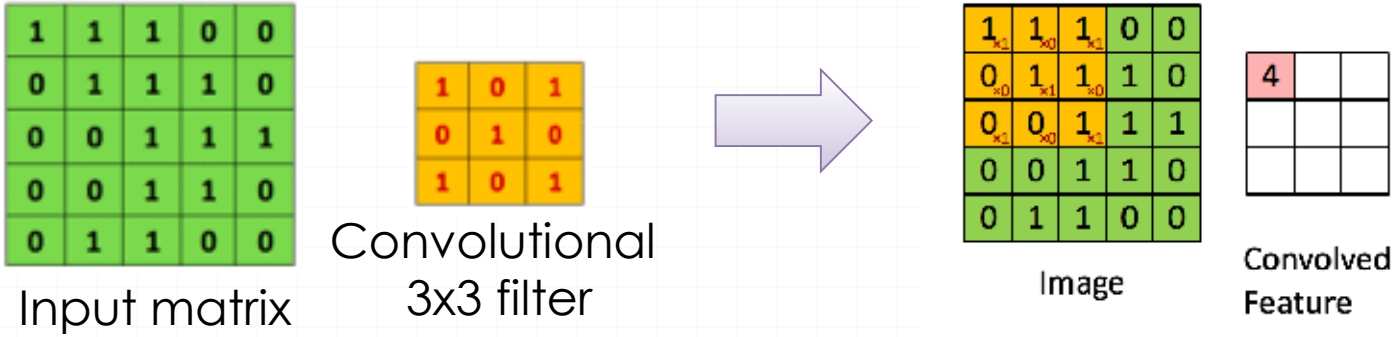
	Classification	Regression
Training examples	R^n x {class_1, …, class_n} (one-hot encoding)	R^n x R^m
Output Layer	Soft-max [map R^n to a probability distribution] $$P(y = j \mid \mathbf{x}) = \frac{e^{\mathbf{x}^\top \mathbf{w}_j}}{\sum_{k=1}^{K} e^{\mathbf{x}^\top \mathbf{w}_k}}$$	Linear (Identity) or Sigmoid 
Cost (loss) function	Cross-entropy $$J(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K}\left[y_k^{(i)} \log \hat{y}_k^{(i)} + \left(1 - y_k^{(i)}\right) \log\left(1 - \hat{y}_k^{(i)}\right)\right]$$	Mean Squared Error $$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - \hat{y}^{(i)}\right)^2$$ Mean Absolute Error $$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}\left\lvert y^{(i)} - \hat{y}^{(i)}\right\rvert$$

List of loss functions

Convolutional Neural Networks (CNNs)

Main CNN idea for text:
Compute vectors for n-grams and group them afterwards

Example: "this takes too long" compute vectors for:
This takes, takes too, too long, this takes too, takes too long, this takes too long



http://deeplearning.stanford.edu/wiki/index.php/Feature_extraction_using_convolution

Disadvantages

Disadvantages

- From a memory and capacity standpoint the CNN is not much bigger than a regular two layer network.
- At runtime the convolution operations are computationally expensive and take up about 67% of the time.
- CNN's are about 3X slower than their fully connected equivalents (size-wise).

Disadvantages

- **Convolution operation**
  - 4 nested loops (2 loops on input image & 2 loops on kernel)
- **Small kernel size**
  - make the inner loops very inefficient as they frequently JMP.
- **Cash unfriendly memory access**
  - Back-propagation require both row-wise and column-wise access to the input and kernel image.
  - 2-D Images represented in a row-wise-serialized order.
  - Column-wise access to data can result in a high rate of cash misses in memory subsystem.

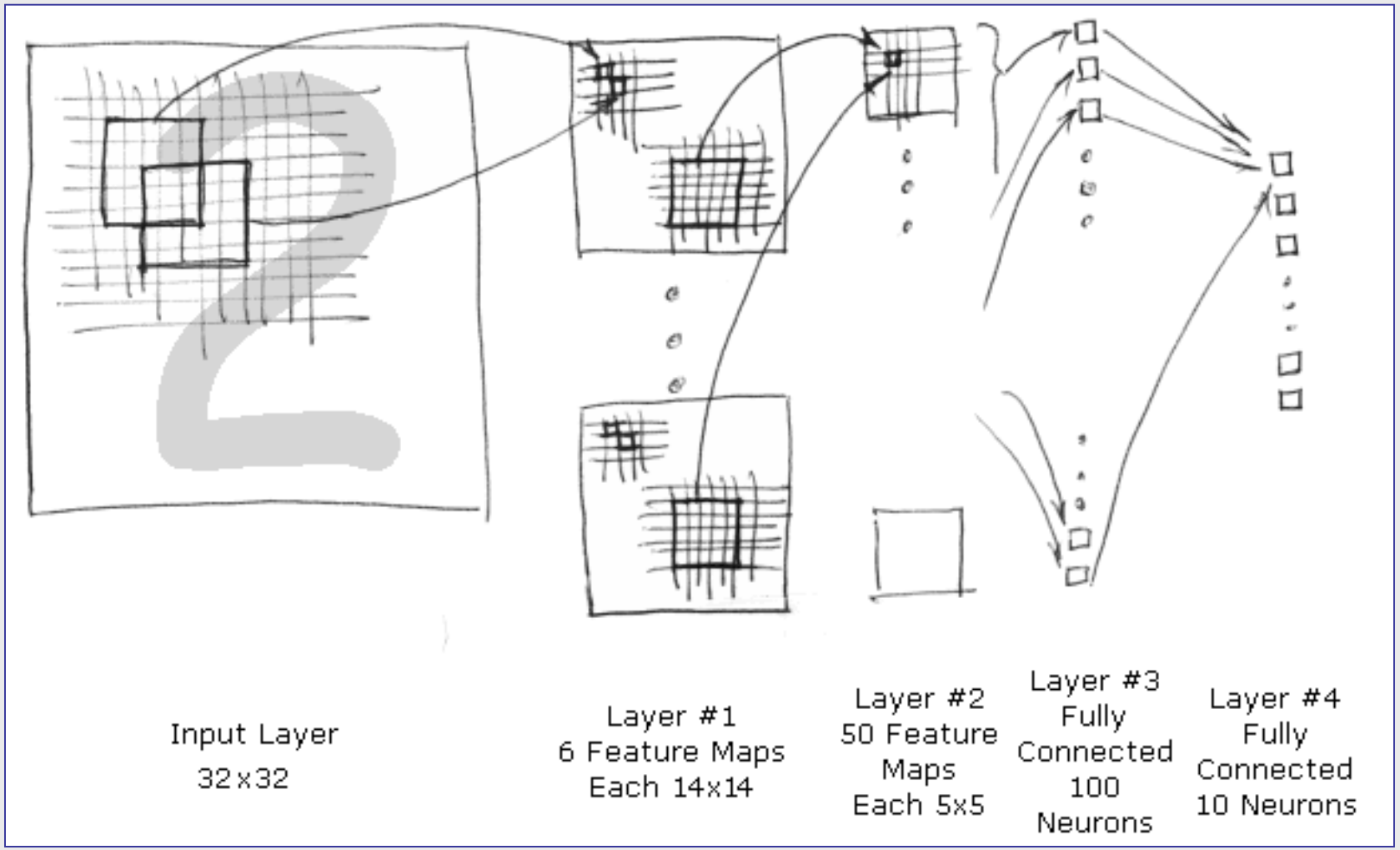
Application

Application

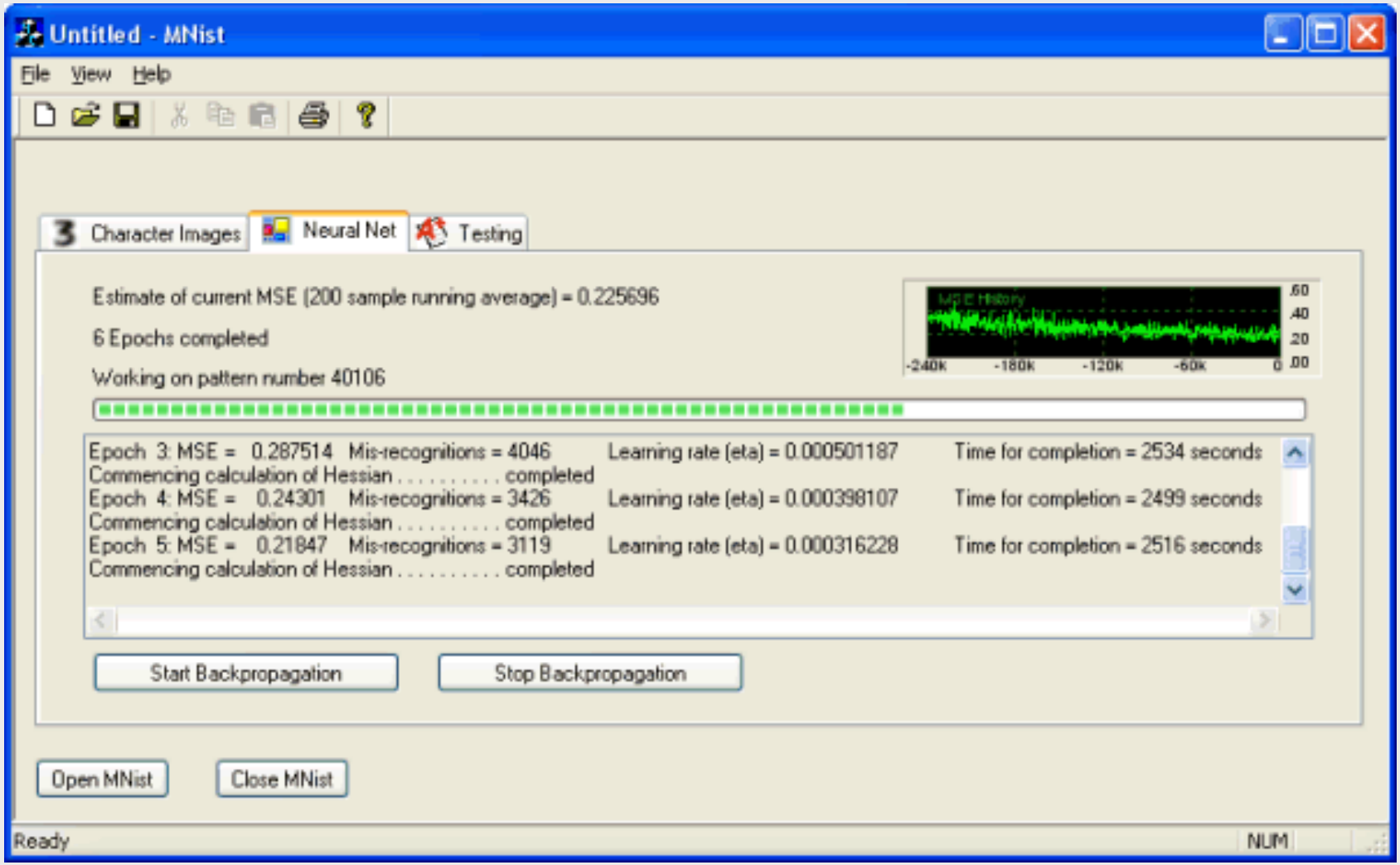
Mike O'Neill

- A Convolutional neural network achieves 99.26% accuracy on a modified NIST database of hand-written digits.
- MNIST database : Consist of 60,000 hand written digits uniformly distributed over 0-9.

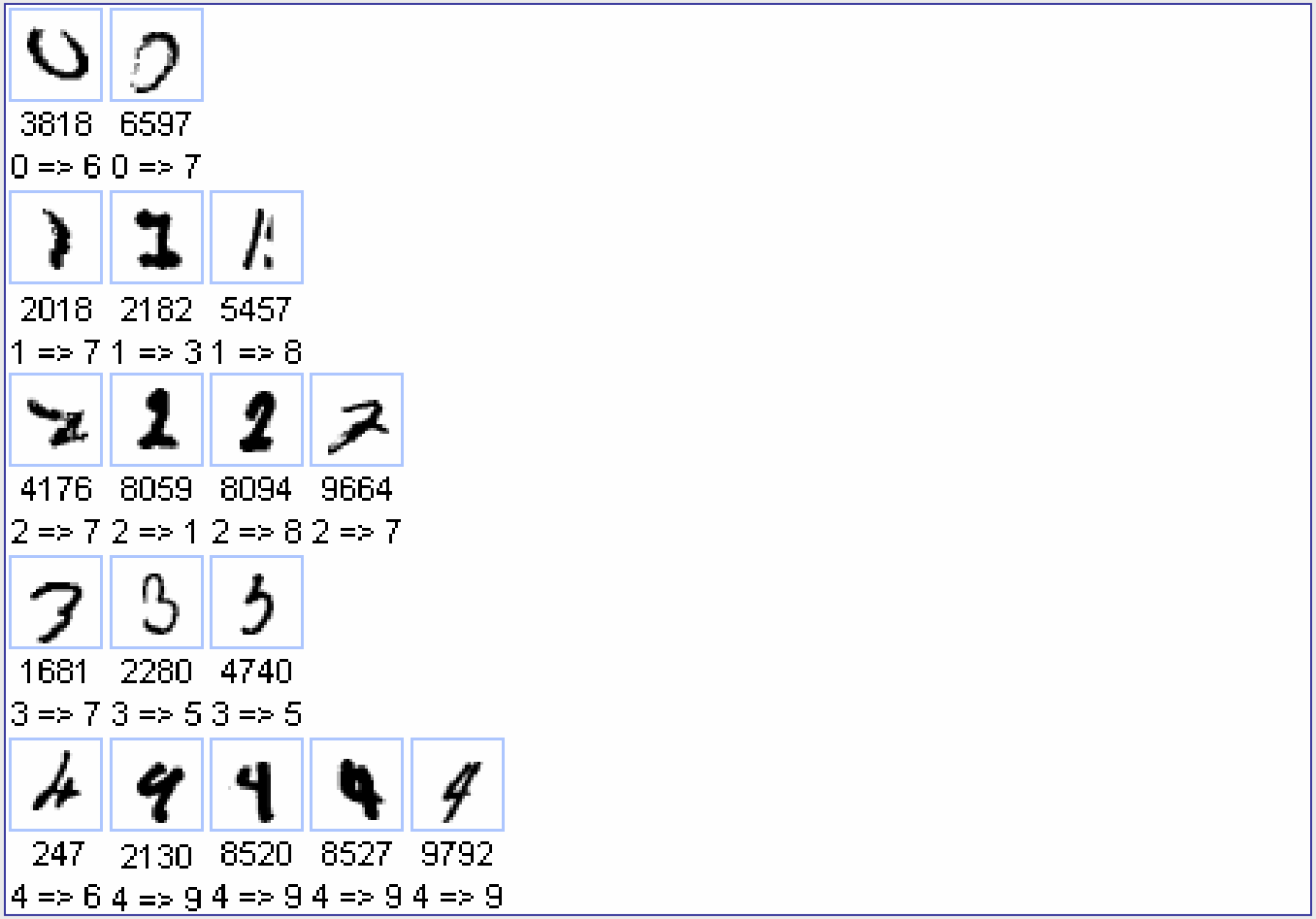
Application



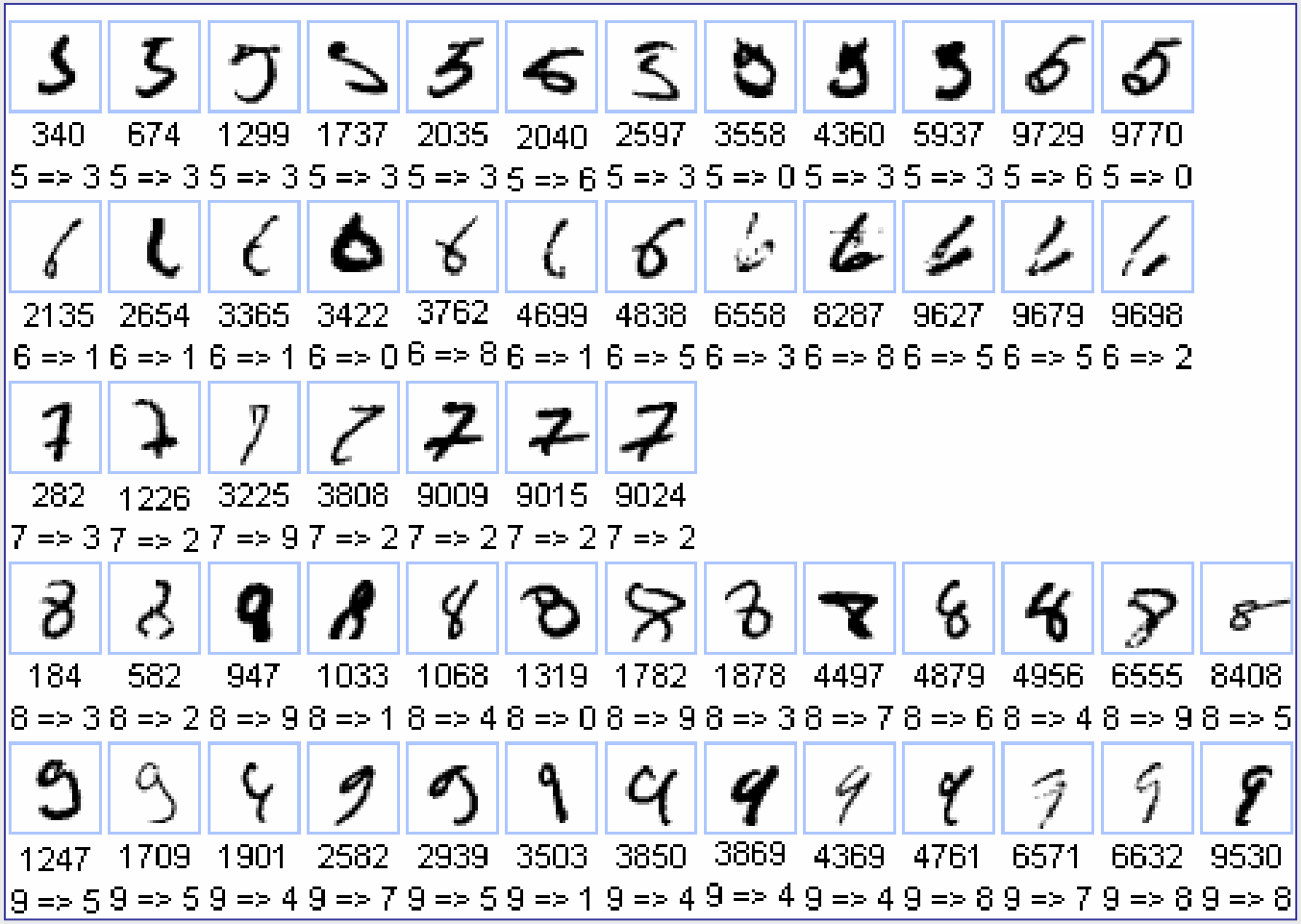
Application



Application



Application



References

[1].Y. LeCun and Y. Bengio.**"Convolutional networks for image s, speech, and time-series."** In M. A. Arbib, editor, *The Hand book of Brain Theory and Neural Networks*. MIT Press, 1995.

[2].Fabien Lauer, ChingY. Suen, Gérard Bloch,**"A trainable feature extractor for handwritten digit recognition"**,Elsevier, october 2006.

[3].Patrice Y. Simard, Dave Steinkraus, John Platt, **"Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis,"** International Conference on Document Analysis and Recognition (ICDAR), IEEE Computer Society, Los Alamitos, pp. 958-962, 2003.

Questions

Pairs of blurred license plates and their respective reconstructions computed by CNN.

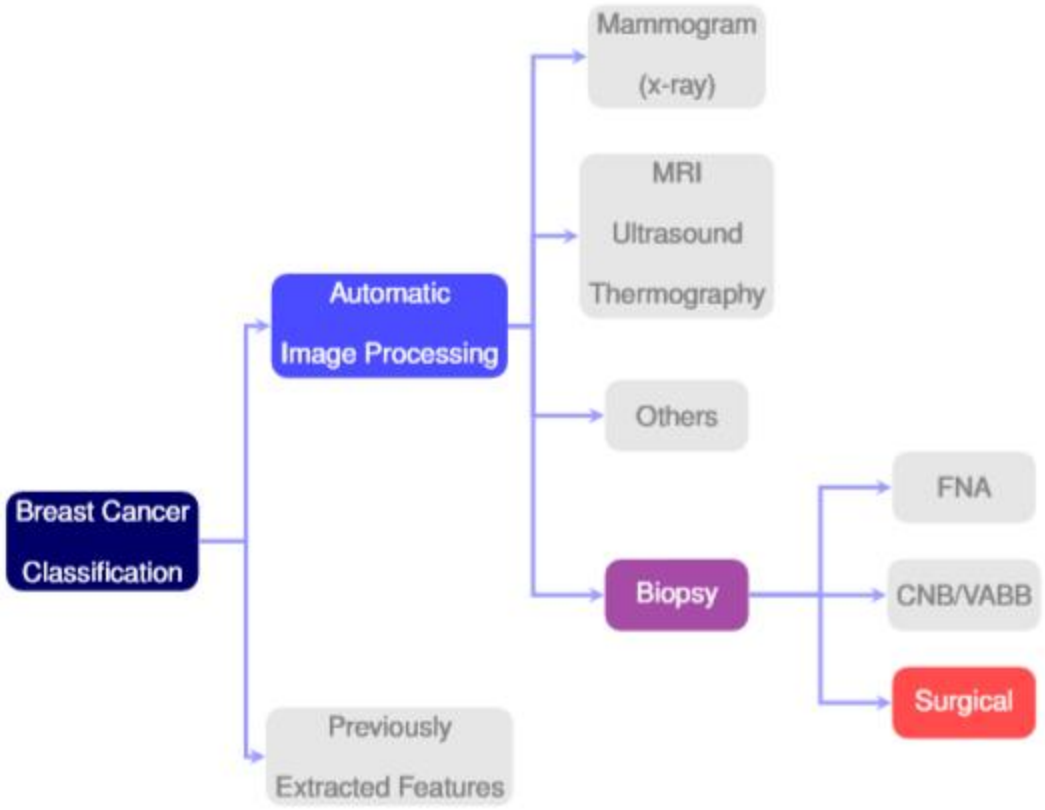


Figure 3.1: Categories of related works on breast cancer classification.
Source: The author (2015).

There are three main types of layers used to build CNN architectures: *convolutional layer*, *pooling layer*, and *fully-connected layer*. Normally, a full CNN architecture is obtained by stacking several of these layers. An example of typical CNN architecture with two feature stages is shown in Figure 2.8.

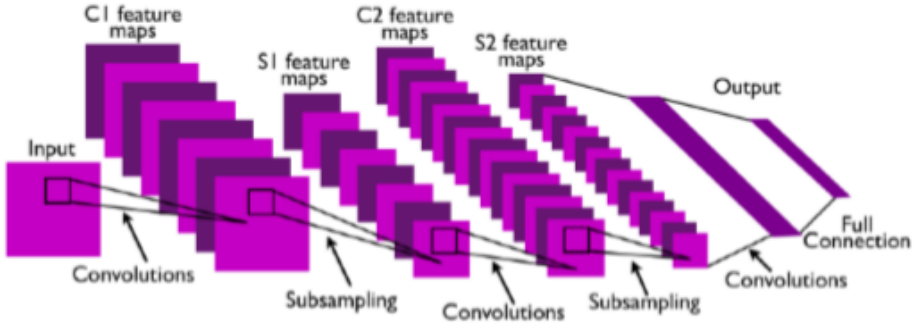


Figure 2.8: An example of a typical CNN architecture with two feature stages.
Source: Extracted from [168].

In a CNN, the key computation is the convolution of a feature detector with an input signal. Convolutional layer computes the output of neurons connected to local regions in the input, each one computing a dot product between their weights and the region they are connected to in the input volume. The set of weights which is convolved with the input is called **filter** or **kernel**. Every filter is spatially small (width and height), but extends through the full depth of the input volume. For inputs such as images, typical filters are small areas (e. g., 3×3, 5×5, or 8×8) and each neuron is only connected to this area in the previous layer. The weights are shared across neurons, leading the filters to learn frequent patterns that occur in any part of the image. The distance between the applications of filters are called **stride**. If the stride hyperparameter is smaller than the filter size, the convolution is applied in overlapping windows.

Convolution with a collection of filters, like the learned filters (also named feature maps or activation maps) in Figure 2.9, improves the representation: at the first layer of a CNN the features go from individual pixels to simple primitives, like horizontal and vertical lines, circles, and patches of color. We can see that the model learns the filters for horizontal and vertical edges, and also learns the filters that resemble Gabor filters (edge detectors) [36, 98]. In contrast to conventional single-channel image processing filters, these CNN filters are computed across all of the input channels. Due to its translation-invariant property, convolutional filters yield a high response wherever a feature is detected.

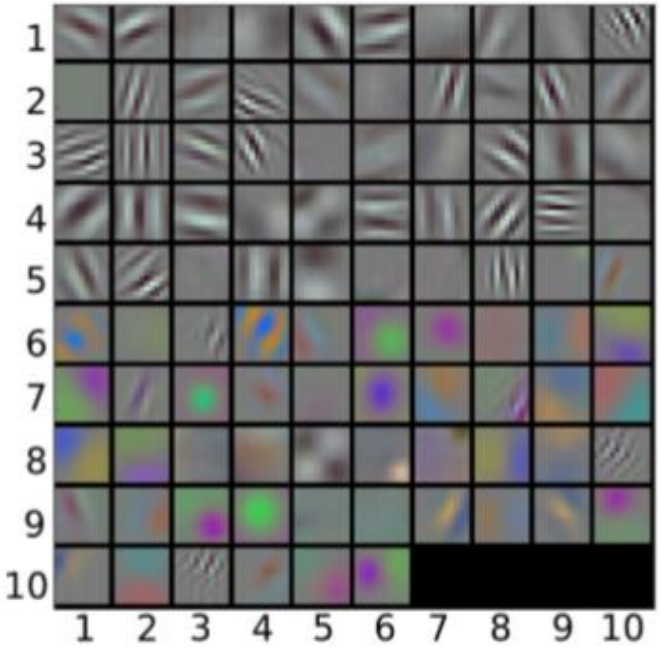


Figure 2.9: The first layer of learned convolutional filters in CaffeNet. This Caffe [141] reference model is based on AlexNet [158]. These filters are tuned to edges of different orientations, frequency, phase and colors. The filter outputs expand the dimensionality of the visual representation from the three color channels of the image to these 96 primitives. Deeper layers further enrich the representation.
Source: The author (2015).

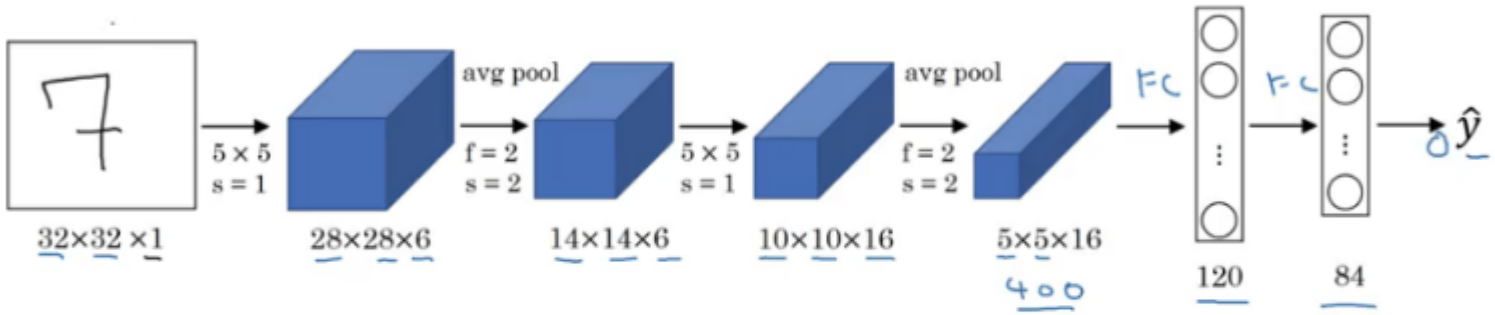
When different features can be learned in different spatial locations the weights sharing scheme of the convolutional layer is relaxed and this layer is then referred to as *locally-connected layer*.

The insertion of a pooling (subsampling) layer between two successive convolutional layers is common. The main objective of this practice is to progressively reduce the spatial size of the representation. Thus, reducing the number of parameters and computations required by the network helps the overfitting control. The pooling layer downsamples the volume spatially in each depth slice of the input volume independently. Thus, the pool operator resizes the input along the width and the height, discarding the activations. In practice, the *max pooling* function, which applies a window function to the input patch, and computes the maximum in that neighborhood, has shown better results [248]. Furthermore, the pooling units can perform other functions like *L2-norm pooling* or *average pooling*.

In a fully-connected layer, neurons have full connections to all activations in the previous layer and their activations can be computed using a matrix multiplication followed by a bias offset. This type of layer is standard in a regular NN. The last fully-connected layer holds the net output, such as probability distributions over classes [90, 158].

LeNet-5

Let's start with LeNet-5:

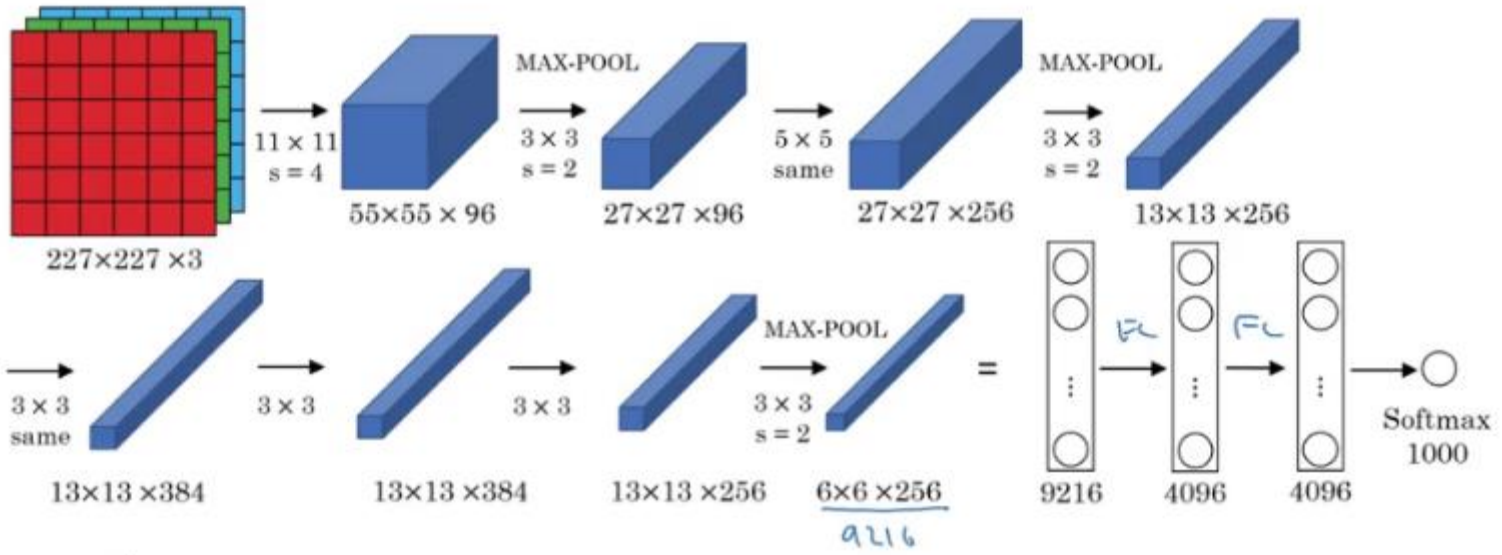


It takes a grayscale image as input. Once we pass it through a combination of convolution and pooling layers, the output will be passed through fully connected layers and classified into corresponding classes. The total number of parameters in LeNet-5 are:

- **Parameters:** 60k
- **Layers flow:** Conv -> Pool -> Conv -> Pool -> FC -> FC -> Output
- **Activation functions:** Sigmoid/tanh and ReLu

AlexNet

An illustrated summary of AlexNet is given below:



This network is similar to LeNet-5 with just more convolution and pooling layers:

- **Parameters:** 60 million
- **Activation function:** ReLu