

# CS490/590 Lecture 11: Image Classification

Eren Gultepe  
Department of Computer Science SIUE

Adapted from Roger Grosse

# Overview

- Object recognition is the task of identifying which object category is present in an image.
- It's challenging because objects can differ widely in position, size, shape, appearance, etc., and we have to deal with occlusions, lighting changes, etc.
- Why we care about it
  - Direct applications to image search
  - Closely related to **object detection**, the task of locating all instances of an object in an image
    - E.g., a self-driving car detecting pedestrians or stop signs
- For the past 6 years, all of the best object recognizers have been various kinds of conv nets.

# Recognition Datasets

- In order to train and evaluate a machine learning system, we need to collect a dataset. The design of the dataset can have major implications.
- Some questions to consider:
  - Which categories to include?
  - Where should the images come from?
  - How many images to collect?
  - How to normalize (preprocess) the images?

# Image Classification

- Conv nets are just one of many possible approaches to image classification. However, they have been by far the most successful for the last 6 years.
- Biggest image classification “advances” of the last two decades
  - Datasets have gotten much larger (because of digital cameras and the Internet)
  - Computers got much faster
    - Graphics processing units (GPUs) turned out to be really good at training big neural nets; they’re generally about 30 times faster than CPUs.
  - As a result, we could fit bigger and bigger neural nets.



# MNIST Dataset

- MNIST dataset of handwritten digits
  - **Categories:** 10 digit classes
  - **Source:** Scans of handwritten zip codes from envelopes
  - **Size:** 60,000 training images and 10,000 test images, grayscale, of size  $28 \times 28$
  - **Normalization:** centered within in the image, scaled to a consistent size
    - The assumption is that the digit recognizer would be part of a larger pipeline that segments and normalizes images.
- In 1998, Yann LeCun and colleagues built a conv net called **LeNet** which was able to classify digits with 98.9% test accuracy.
  - It was good enough to be used in a system for automatically reading numbers on checks.

# ImageNet

ImageNet is the modern object recognition benchmark dataset. It was introduced in 2009, and has led to amazing progress in object recognition since then.

ILSVRC

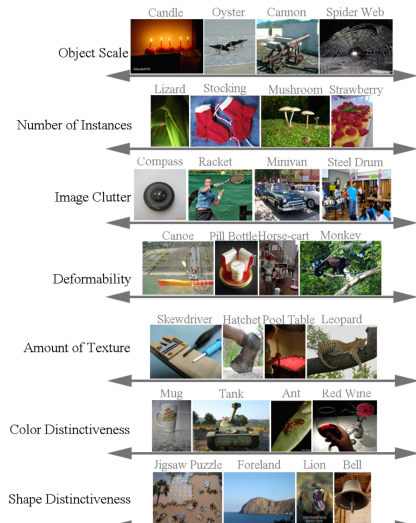


# ImageNet

- Used for the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), an annual benchmark competition for object recognition algorithms
- Design decisions
  - **Categories:** Taken from a lexical database called WordNet
    - WordNet consists of “synsets”, or sets of synonymous words
    - They tried to use as many of these as possible; almost 22,000 as of 2010
    - Of these, they chose the 1000 most common for the ILSVRC
    - The categories are really specific, e.g. hundreds of kinds of dogs
  - **Size:** 1.2 million full-sized images for the ILSVRC
  - **Source:** Results from image search engines, hand-labeled by Mechanical Turkers
    - Labeling such specific categories was challenging; annotators had to be given the WordNet hierarchy, Wikipedia, etc.
  - **Normalization:** none, although the contestants are free to do preprocessing

# ImageNet

Images and object categories vary on a lot of dimensions



Russakovsky et al.

# ImageNet

Size on disk:

MNIST  
60 MB

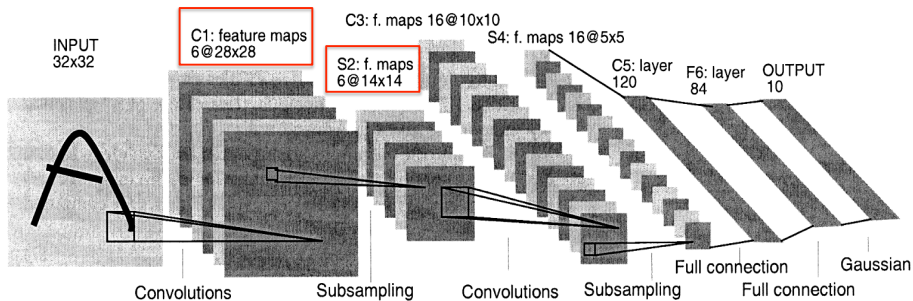


ImageNet  
50 GB



# LeNet

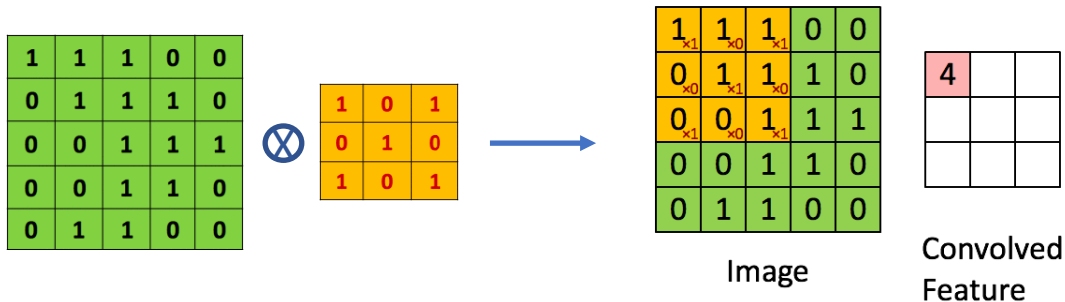
Here's the LeNet architecture, which was applied to handwritten digit recognition on MNIST in 1998:



# Size of a Conv Net

- Ways to measure the size of a network:
  - **Number of units.** This is important because the activations need to be stored in memory during training (i.e. backprop).
  - **Number of weights.** This is important because the weights need to be stored in memory, and because the number of parameters determines the amount of overfitting.
  - **Number of connections.** This is important because there are approximately 3 add-multiply operations per connection (1 for the forward pass, 2 for the backward pass).
- We saw that a fully connected layer with  $M$  input units and  $N$  output units has  $MN$  connections and  $MN$  weights.
- The story for conv nets is more complicated.

# A closer look at convolutions



Input: 5 x 5

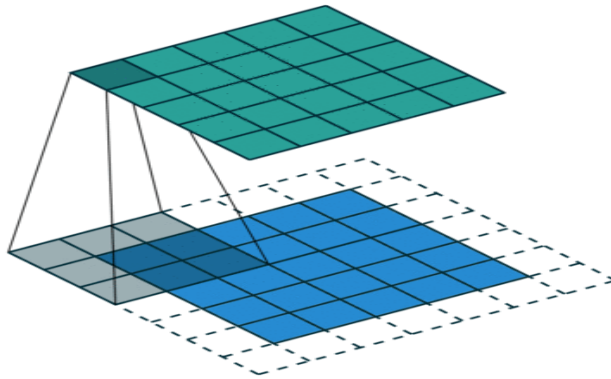
Kernel: 3 x 3

Output: 3x 3

$$3 = (5 - 3) + 1$$



# Zero Padding



Input: 5 x 5

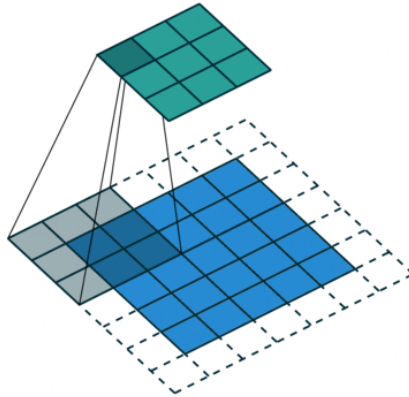
Output: 5 x 5

Kernel: 3 x 3

Padding: 1

$$5 = (5 - 3 + 2 * 1) + 1$$

# Stride



Input: 5 x 5

Kernel: 3 x 3

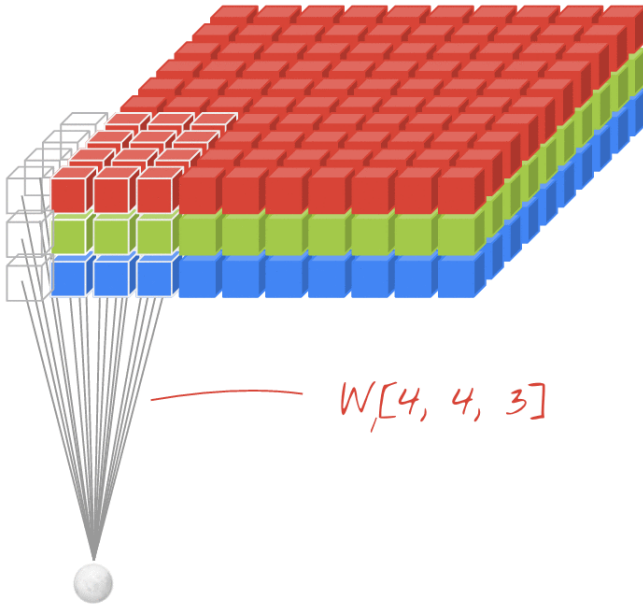
Padding: 1

Stride = 2

Output: 3 x 3

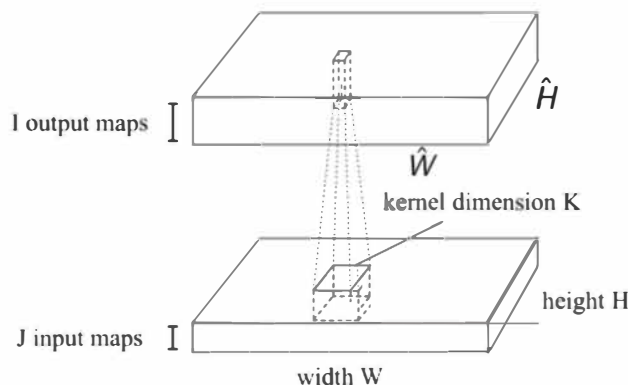
$$3 = (5 - 3 + 2 * 1) / 2 + 1$$

# Convolution Depth



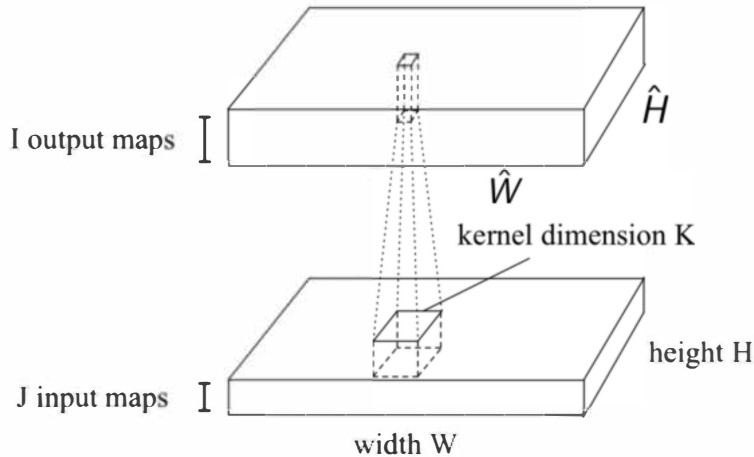
**Note: Kernel depth always equals to the input depth.**

## Summary : Convolution Layer



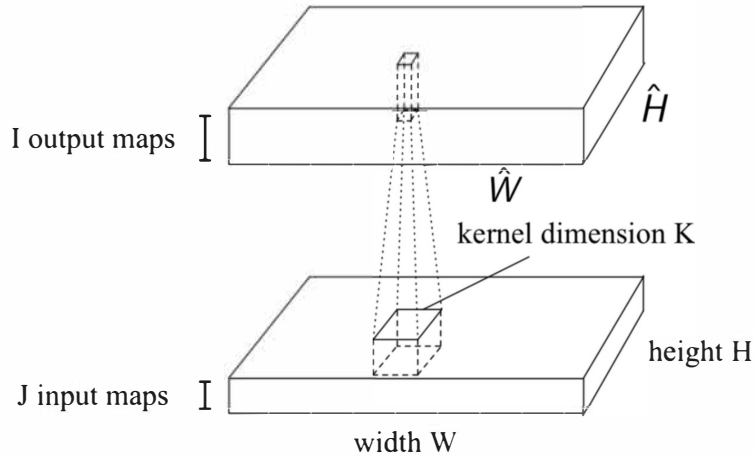
- Input: An array of size  $W \times H \times J$
- Hyper-parameters:
  - Number of filters:  $M$
  - Size of filters:  $K$
  - the stride:  $S$
  - Number of zero-padding:  $P$
- Output: Feature maps of size  $\hat{W} \times \hat{H} \times I$ 
  - $\hat{W} = (W - K + 2P)/S + 1$
  - $\hat{H} = (H - K + 2P)/S + 1$
  - $I = M$

# Let's do some counting: Size of a Conv Net



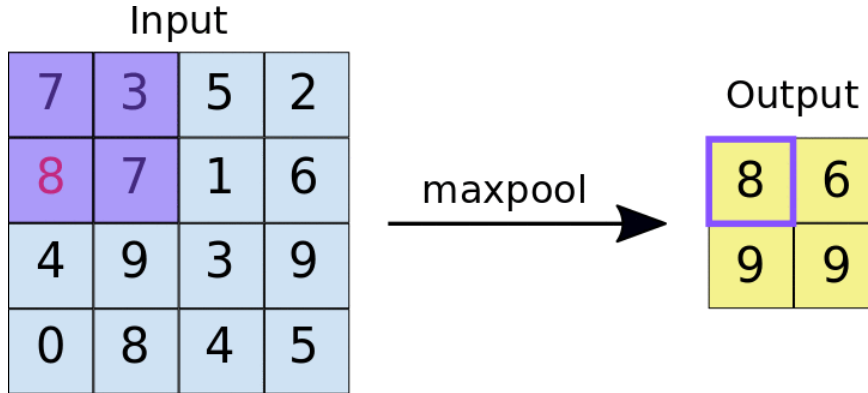
	fully connected layer	convolution layer
# output units	$\hat{W}\hat{H}I$	$\hat{W}\hat{H}I$
# weights	$W\hat{W}H\hat{H}IJ$	$K^2IJ$
# connections	$W\hat{W}H\hat{H}IJ$	$\hat{W}\hat{H}K^2IJ$

# Let's do some counting: Size of a Conv Net (Including bias terms)



	<b>fully connected layer</b>	<b>convolution layer</b>
<b># output units</b>	$\hat{W}\hat{H}I$	$\hat{W}\hat{H}I$
<b># weights</b>	$W\hat{W}H\hat{H}IJ + \hat{W}\hat{H}I$	$K^2IJ + I$
<b># connections</b>	$W\hat{W}H\hat{H}IJ + \hat{W}\hat{H}I$	$\hat{W}\hat{H}K^2IJ + \hat{W}\hat{H}I$

# Pooling Layers



Input: 4 x 4

Kernel: 2 x 2

Stride: 2

Output: 2 x 2

$$2 = (4 - 2) / 2 + 1$$

## Size of a Conv Net

Sizes of layers in LeNet:

Layer	Type	# units	# connections	# weights
C1	convolution	4704	117,600	150
S2	pooling	1176	4704	0
C3	convolution	1600	240,000	2400
S4	pooling	400	1600	0
F5	fully connected	120	48,000	48,000
F6	fully connected	84	10,080	10,080
output	fully connected	10	840	840

Conclusions?



# Size of a Conv Net

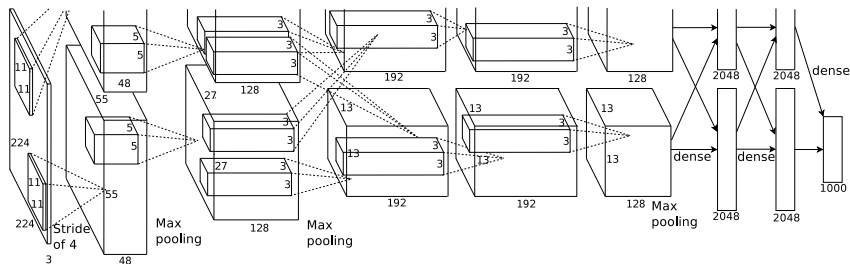
- Rules of thumb:
  - Most of the units and connections are in the convolution layers.
  - Most of the weights are in the fully connected layers.
- If you try to make layers larger, you'll run up against various resource limitations (i.e. computation time, memory)
- Conv nets have gotten a LOT larger since 1998!

# Size of a Conv Net

	<b>LeNet (1989)</b>	<b>LeNet (1998)</b>	<b>AlexNet (2012)</b>
<b>classification task</b>	digits	digits	objects
<b>categories</b>	10	10	1,000
<b>image size</b>	$16 \times 16$	$28 \times 28$	$256 \times 256 \times 3$
<b>training examples</b>	7,291	60,000	1.2 million
<b>units</b>	1,256	8,084	658,000
<b>parameters</b>	9,760	60,000	60 million
<b>connections</b>	65,000	344,000	652 million
<b>total operations</b>	11 billion	412 billion	200 quadrillion (est.)

# AlexNet

- AlexNet, 2012. 8 weight layers. 16.4% top-5 error (i.e. the network gets 5 tries to guess the right category).



(Krizhevsky et al., 2012)

- They used lots of tricks we've covered in this course (ReLU units, weight decay, data augmentation, SGD with momentum, dropout)
- AlexNet's stunning performance on the ILSVRC is what set off the deep learning boom of the last 6 years.

# GoogLeNet

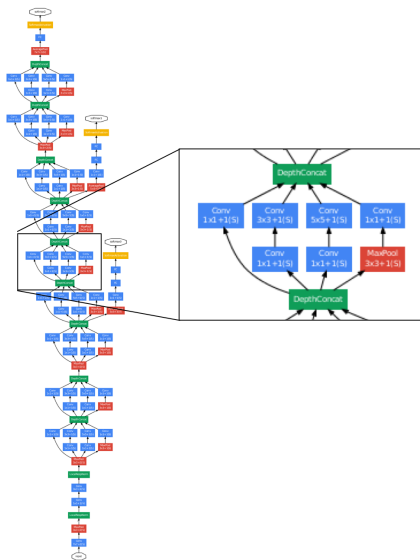
GoogLeNet, 2014.

22 weight layers

Fully convolutional (no fully connected layers)

Convolutions are broken down into a bunch of smaller convolutions

6.6% test error on ImageNet



- They were really aggressive about cutting the number of parameters.
  - Motivation: train the network on a large cluster, run it on a cell phone
    - Memory at test time is the big constraint.
    - Having lots of units is OK, since the activations only need to be stored at training time (for backpropagation).
    - Parameters need to be stored both at training and test time, so these are the memory bottleneck.
  - How they did it
    - No fully connected layers (remember, these have most of the weights)
    - Break down convolutions into multiple smaller convolutions (since this requires fewer parameters total)
  - GoogLeNet has “only” 2 million parameters, compared with 60 million for AlexNet
  - This turned out to improve generalization as well. (Overfitting can still be a problem, even with over a million images!)

# Classification

ImageNet results over the years. Note that errors are top-5 errors (the network gets to make 5 guesses).

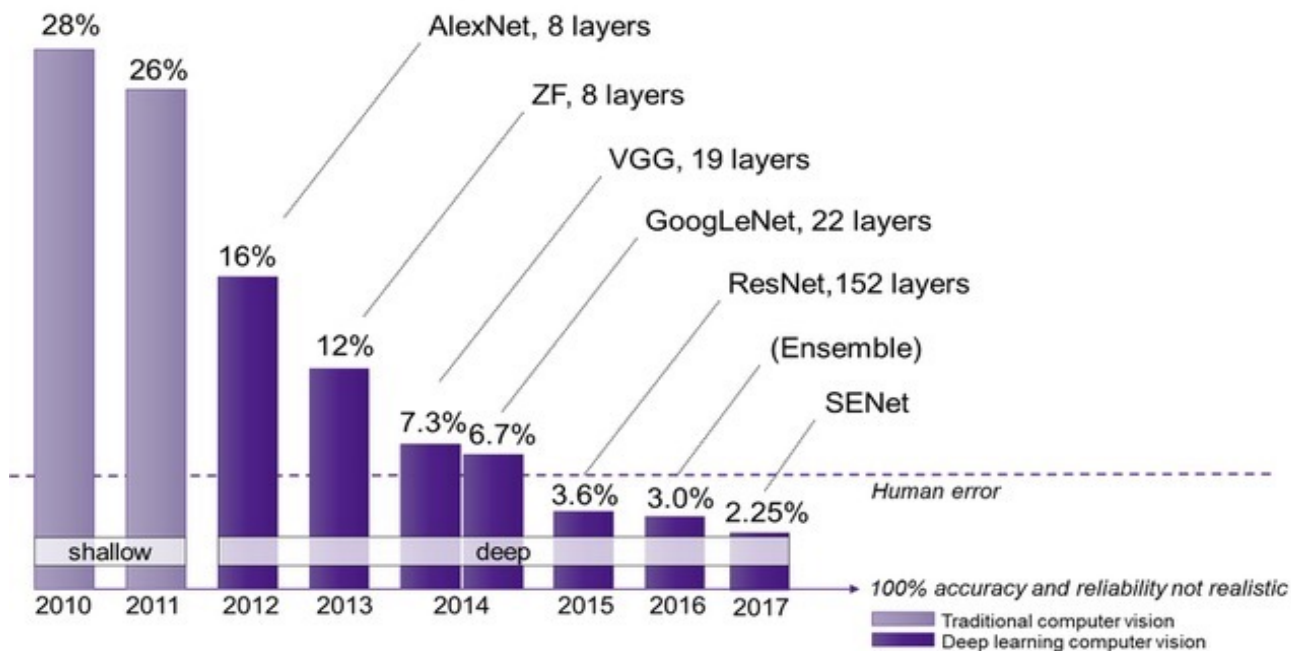
<b>Year</b>	<b>Model</b>	<b>Top-5 error</b>
2010	Hand-designed descriptors + SVM	28.2%
2011	Compressed Fisher Vectors + SVM	25.8%
2012	AlexNet	16.4%
2013	a variant of AlexNet	11.7%
2014	GoogLeNet	6.6%
2015	deep residual nets	4.5%

We'll cover deep residual nets later in the course, since they require an idea we haven't covered yet.

Human-performance is around 5.1%.

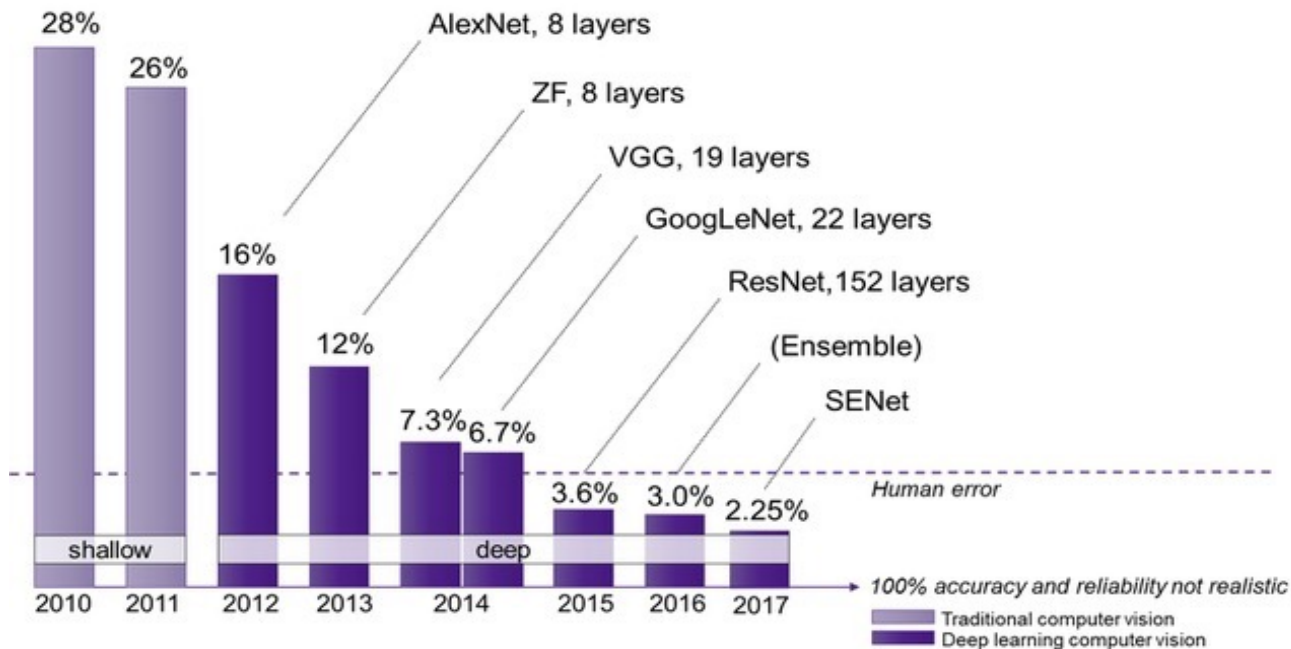
They stopped running the object recognition competition because the performance is already so good.

## A brief history of ImageNet Competition



Observation 1: Deep models (mostly CNNs) dominate the task of object recognition.

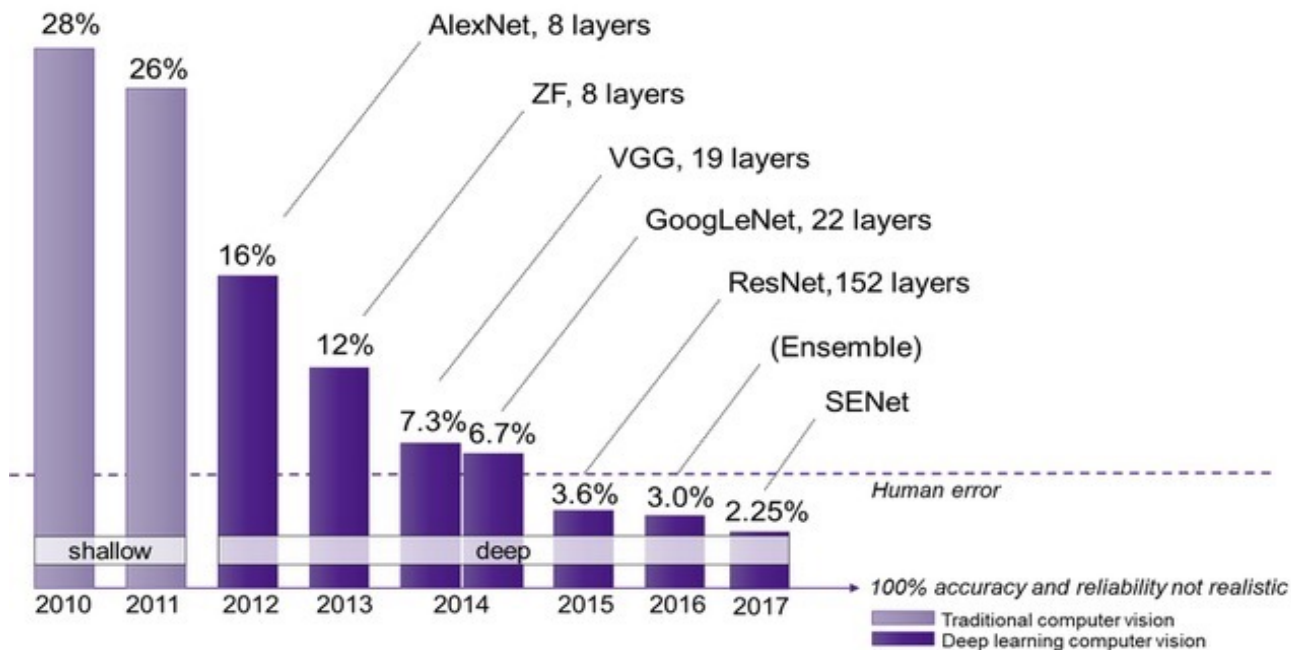
## A brief history of ImageNet Competition



Observation 2: The accuracy by CNNs improves every year and even outperforms human performances.

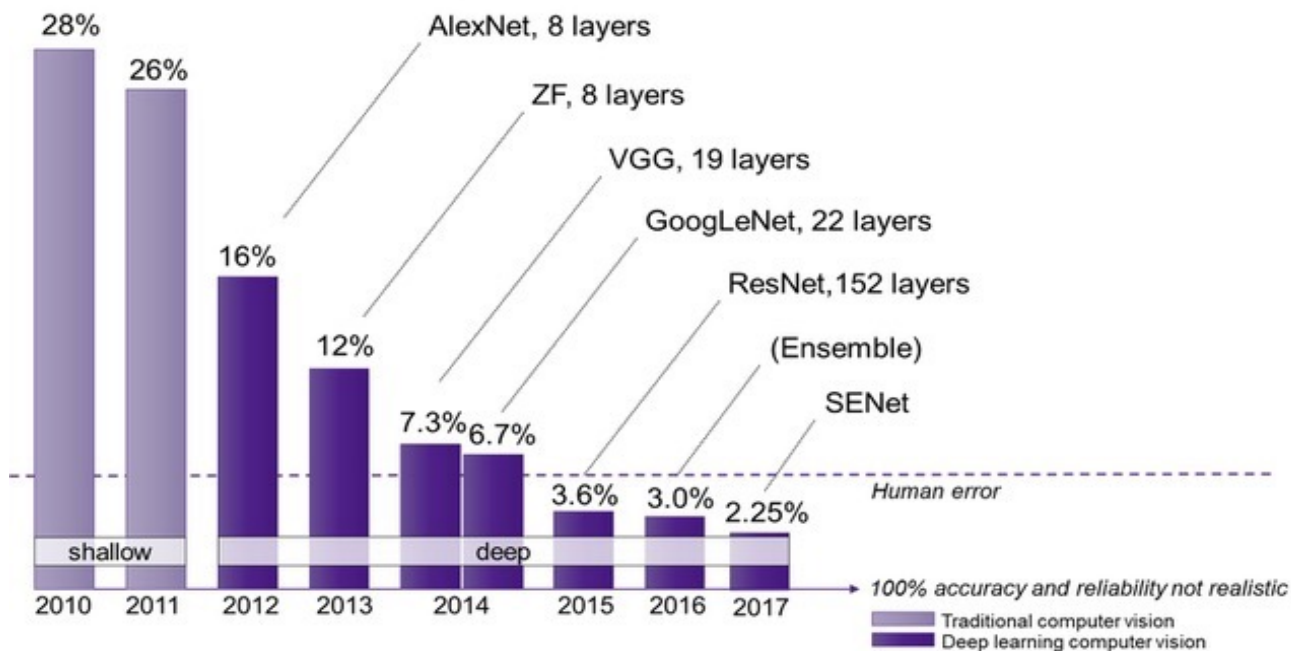


## A brief history of ImageNet Competition



Observation 3: The STOA CNNs are getting deeper and deeper.

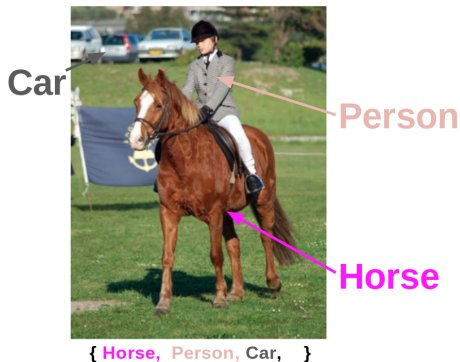
## A brief history of ImageNet Competition



Observation 4: The performance of object recognition is so good that they stopped the competition nowadays.

## Beyond Classification

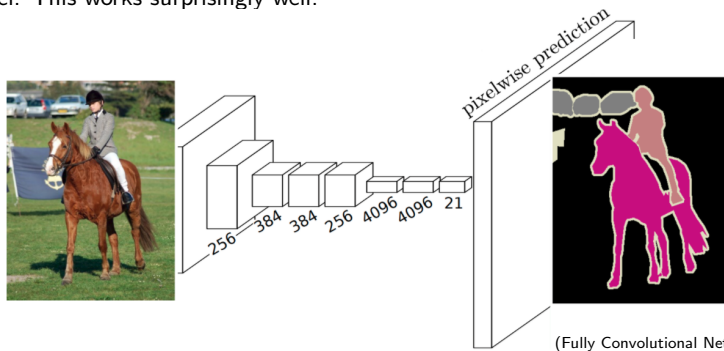
- The classification nets map the entire input image to a pre-defined class categories.
- But there are more than just class labels in an image.
  - where is the foreground object? how many? what is in the background?



(PASCAL VOC 2012)

# Semantic Segmentation

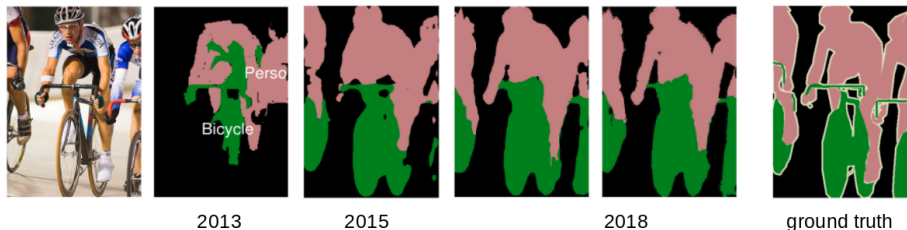
- Semantic segmentation, a natural extension of classification, focuses on making dense classification of class labels for **every pixel**.
- It is an important step towards complete scene understanding in computer vision.
  - Semantic segmentation is a stepping stone for many of the high-level vision tasks, such as object detection, Visual Question Answering (VQA).
- A naive approach is to adapt the existing object classification conv nets for each pixel. This works surprisingly well.



(Fully Convolutional Networks, 2015)

# Semantic Segmentation

- After the success of CNN classifiers, segmentation models quickly moved away from hand-craft features and pipelines but instead use CNN as the main structure.
- Pre-trained ImageNet classification network serves as a building block for all the state-of-the-art CNN-based segmentation models.



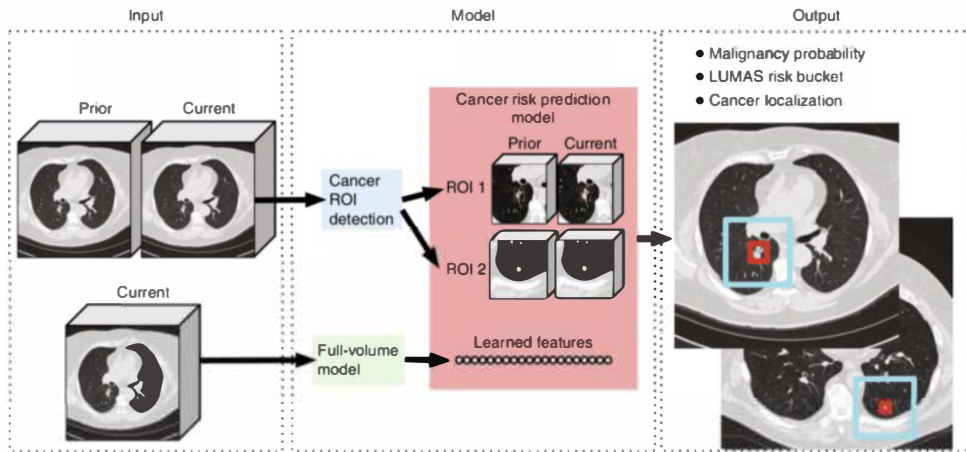
from left to right (Li, et. al., (CSI), CVPR, 2013; Long, et. al., (FCN), CVPR 2015; Chen et. al., (DeepLab), PAMI 2018)

# Supervised Pre-training and Transfer Learning

- In practice, we will rarely train an image classifier from scratch.
  - It is unlikely we will have millions of cleanly labeled images for our specific datasets.
- If the dataset is a computer vision task, it is common to fine-tune a pre-trained conv net on ImageNet or OpenImage.
- Just like semantic segmentation tasks, we will fix most of the weights in the pre-trained network. Only the weights in the last layer will be randomly initialized and learnt on the current dataset/task.
- When and how?
  - How many training examples we have in the new dataset/task? Fewer new examples: more weights from the pre-trained networks are fixed.
  - How similar is the new dataset to our pre-training dataset? Microscopy images v.s. natural images: more fine-tuning is needed for dissimilar datasets.
  - Learning rate for the fine-tuning stage is often much lower than the learning rate used for training from scratch.

# CNN on HealthCare

- CNN has also been widely used in processing medical images.
- Pre-trained ImageNet classification network serves as a building block for most of medical image segmentation/classification models.



from: google AI: End-to-end lung cancer screening with three-dimensional deep learning on low-dose chest computed tomography. Nature medicine, 2019.

# Transfer Learning : Fine-Tuning

