Convolutional Neural Networks

A Brief High-Level Introduction

AI+X
INSTITUTE FOR ARTIFICIAL INTELLIGENCE

Perceptron (P)

The perceptron can only work well for linearly separable data

Feed Forward (FF)

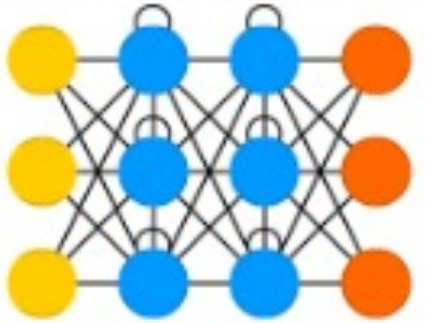
Feed forward backpropagation neural networks are universal approximators.

However, we do not know the transfer function, configuration OR if we can learn the function for a given problem.

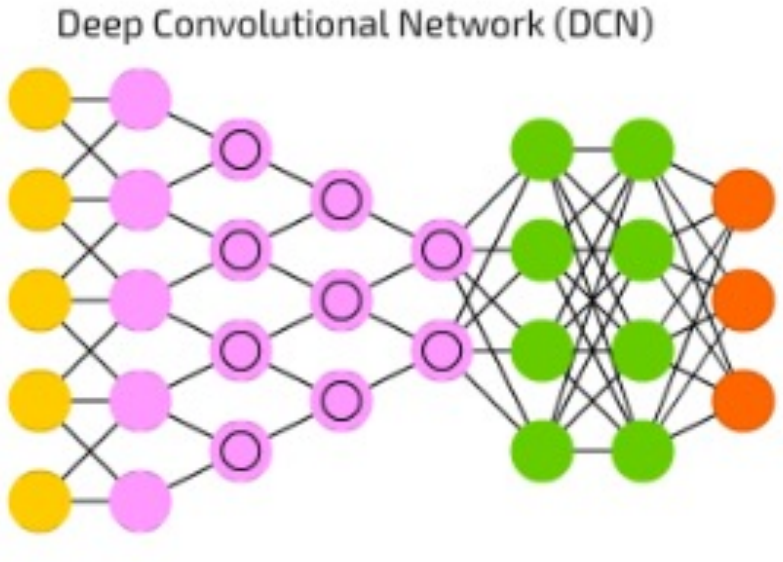
Deep Feed Forward (DFF)

A Deep network has more than 1 hidden layer. A bit more brain like and often several more hidden layers.

Recurrent Neural Network (RNN)



Recurrent neural networks are great for times series and perhaps images. They feed in previous epoch information.



A CNN breaks data into smaller pieces. It then feeds the data after convolution and pooling which does a kind of feature extraction into a feed forward backpropagation neural network (often of 2 layers).

Most contest wins on images are from the above type of network.

Images as input

- Think of 28x28 images of handwritten characters (MNIST):

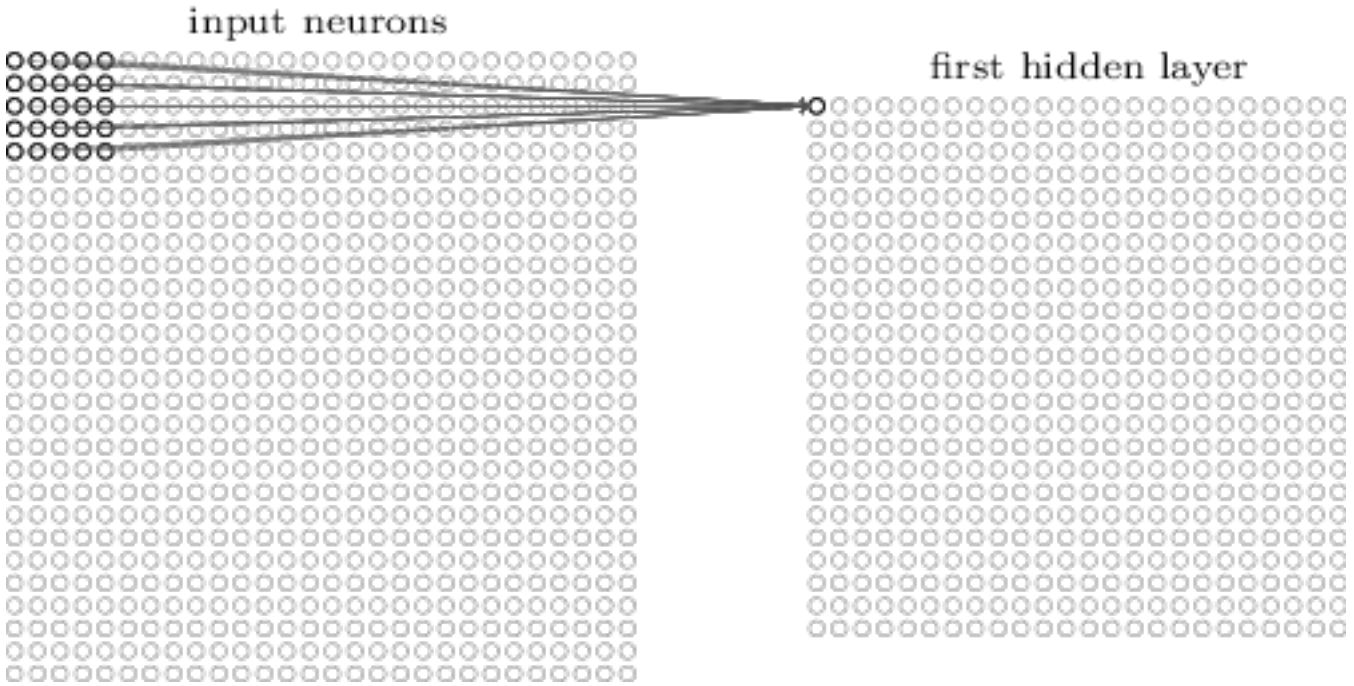
- Let’s just use the pixel intensities as features (784)

Images from ImageNet

Some of the 1000 classes of images

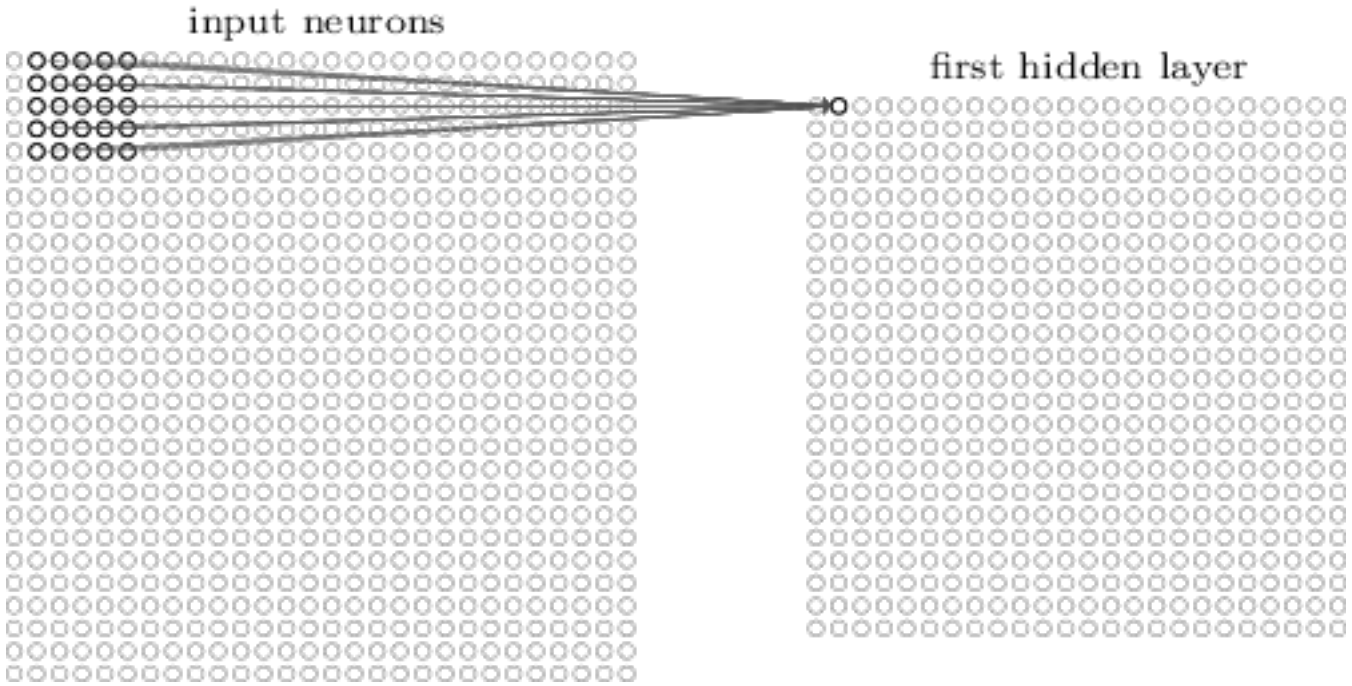
Using convolution

- Let’s use 5x5 “local receptive fields” and slide them by a stride of 1.
- Gives us a 24x24 convolutional hidden layer.



Using convolution

- Let's use 5x5 "local receptive fields" and slide them by a stride of 1.
- Gives us a 24x24 convolutional hidden layer.



Some details

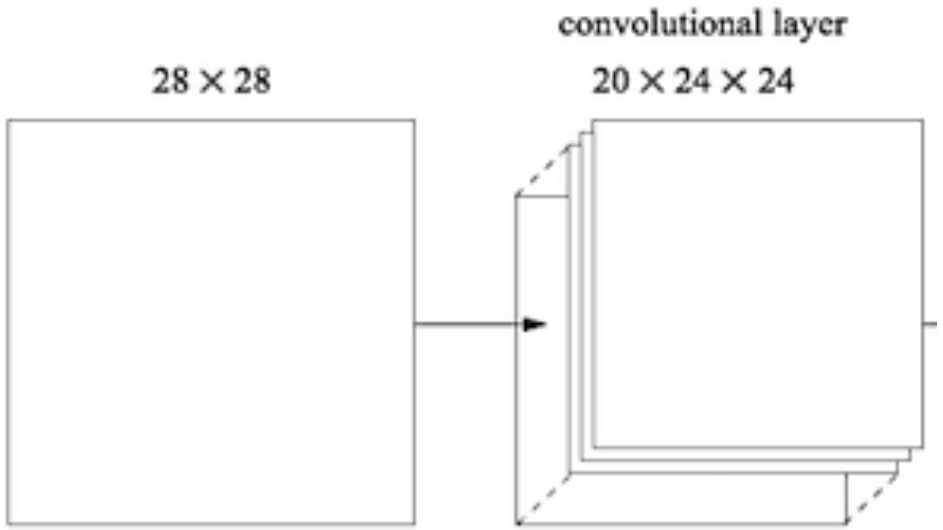
- Each hidden unit in the convolutional layer has a bias and 25 weights attached to it.
- The weights and bias will be the same for all 576 hidden units. Formally, the output of the j,kth hidden neuron for activation function f is:

$$f\left(b + \sum_{l=0}^{4} \sum_{m=0}^{4} w_{l,m} o_{j+l,k+m}\right)$$

- Where b is the bias, **w** is the vector of 25 weights and **o** is the output of the appropriate neuron from the window in the convolutional layer.

Convolutional layer

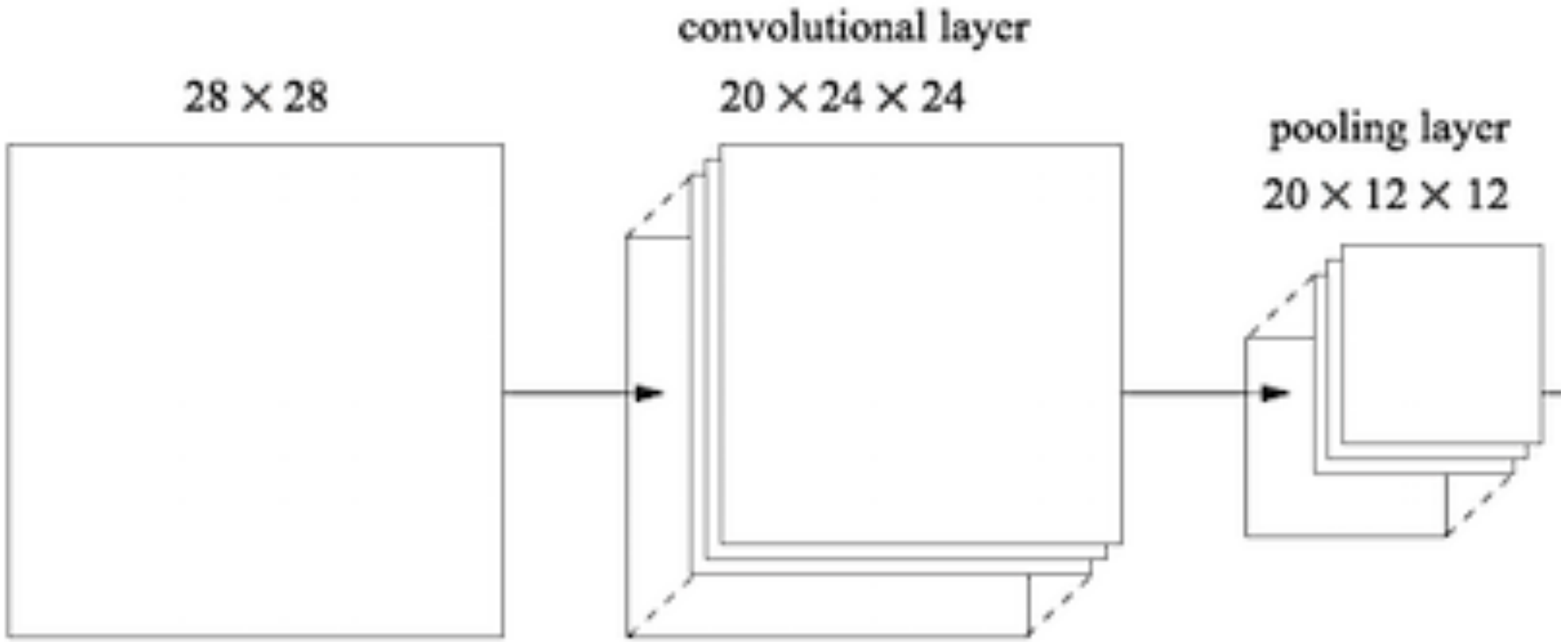
- We can think of a single convolution as pulling out one feature from an image.
- So, maybe in the hidden layer we need n convolutions. Below n= 20.



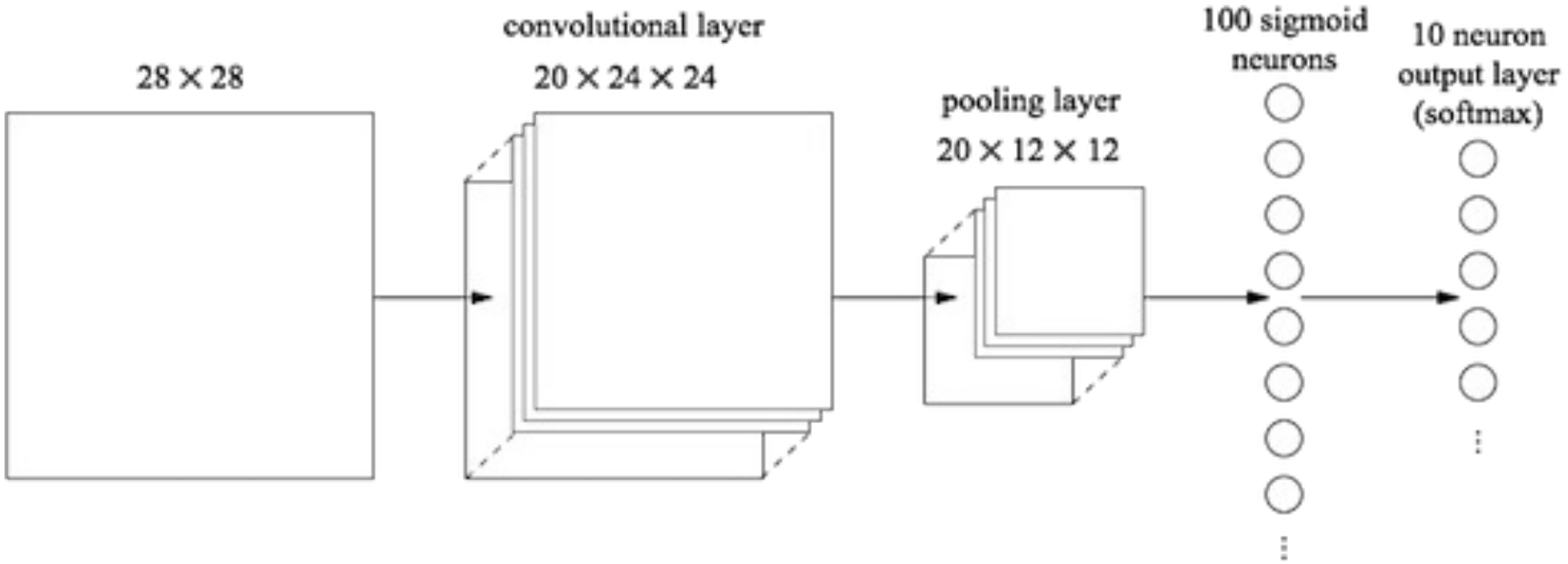
Max and average Pooling layer

- Pooling layers are used after convolutional layers to simplify or reduce the information.
- They will be applied to each of the convolutions so we would have 20 applications in this example.
- Often a 2x2 window is used (so 24x24 gets reduced to 12x12). Non-overlapping (or stride of 2).
- The max part is that the maximum value from the window is all that is passed on.
- For an average pooling layer you get the average of the window.

Max Pooling layer

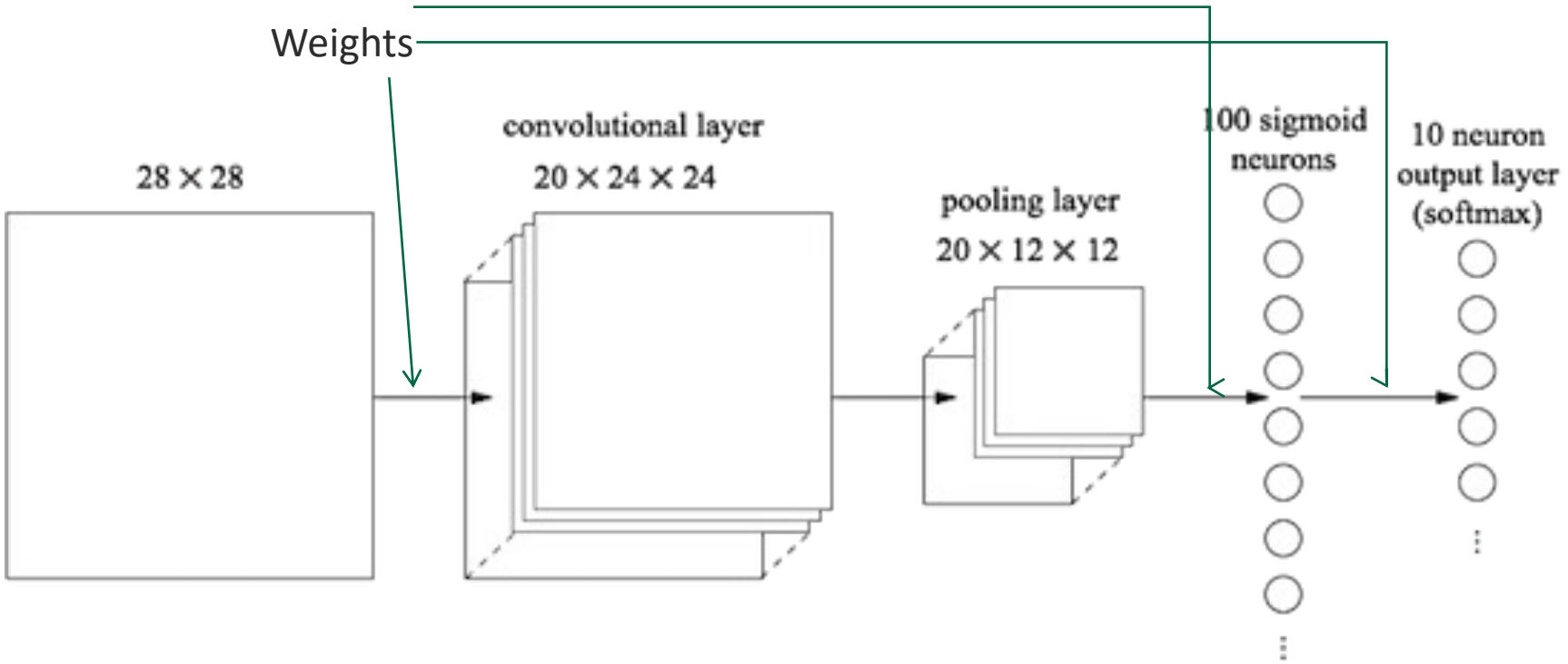


Small 5-layer CNN for MNIST images



Plenty of weights. Just from pooling to the sigmoid layer there are 2880 weights to each neuron (so 288,000 weights). How many overall to learn?

Small 5-layer CNN for MNIST images



How many overall to learn?
520 + 288,000 + 1,000 = 289,520

MNIST Results

- Elastic distortions allow training data augmentation by creating realistic, possible images
- Large/deep convolutional neural net (elastic distortions) 0.35 error on test data (vs. 0.95 for no distortions and an early CNN
- Ensemble of 35 convolutional networks (elastic distortions) 0.23 error

Imagenet Results and Architectures

- Large Scale Visual Recognition Challenge – Done yearly with 1000 categories of RGB images
- Training data is: 10,000,000 labeled images depicting 10,000+ object categories
- Test data for 2017 was: 150,000 photographs, collected from flickr and other search engines, hand labeled with the presence or absence of 1000 objects
- Validation data was a random 50,000 from the 150,000 examples

Large Scale Visual Recognition Challenge

Name	Layers	Top-5 Error %
AlexNet (2012)	8	15.3
VGG Net (2014)	19	7.3
ResNet (2016)	152	3.6
Squeeze and Excitation ResNet – Ensemble (2017)	152	2.25

VGG-B-net Architecture

- 3x3 kernel or receptive field for all convolution layers, stride of 1 and padding of 1, so no reduction in image size
- Max pooling with a 2x2 window with a stride of 2
- Two fully connected layers of 4096 units with softmax outputs
- Rectified linear unit (RELU) activation function. All negative sums made 0

VGG-B-net Architecture

Layer	
Input	224x224x3
Convolutional	3x3x64 3x3x64
maxpool	2x2
convolutional	3x3x128 3x3x128
maxpool	2x2
convolutional	3x3x256 3x3x256
maxpool	2x2
convolutional	3x3x512 3x3x512
maxpool	2x2

VGG-B-net Architecture - Continued

Layer	
Fully connected	4096
Fully connected	4096
Output - Softmax	1000

ResNet-50 Architecture (224x224x3 input). Top 5 Error 5.25%

	7x7x64 stride 2 **3x3 maxpool stride 2**
	1×1, 64 3×3, 64 x 3 1×1, 256
	1×1, 128 3×3, 128 x 4 1×1, 512
	1×1, 256 3×3, 256 x 6 1×1, 1024
	1×1, 512 3×3, 512 x 3 1×1, 2048
	average pool, 1000-d fully connected, softmax

Quick Summary

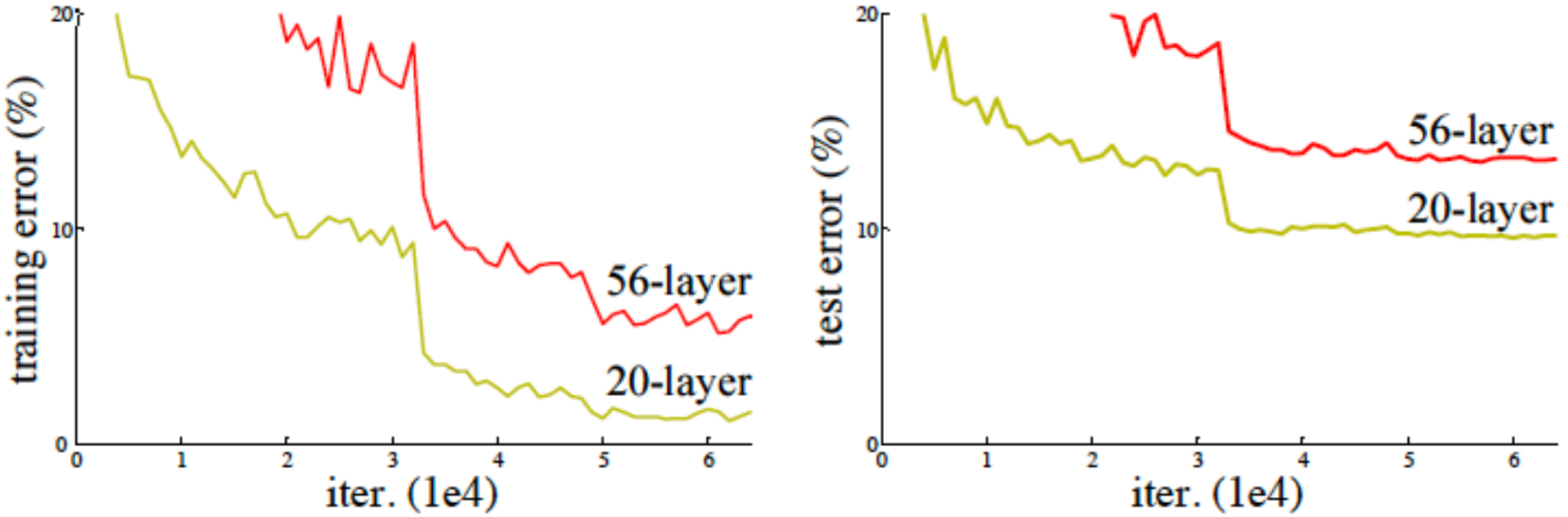
- CNN's take, for example, a raw image as input. Then convolution and max pooling are typically applied.
- You can have multiple convolutions followed by max pooling.
- You can have multiple "traditional" hidden units.
- You may use dropout in training
  - In dropout, you do not change a randomly chosen set of weights during training

Quick Summary

- You WILL have lots of weights to train, 289,520 in our simple example.
- You need lots of examples to train these (so big data), but training will be slow.
- You can try tuning a deep neural network built for a different, but related problem.
- Accuracy is the best for images, text, voice, today.

Going Deeper

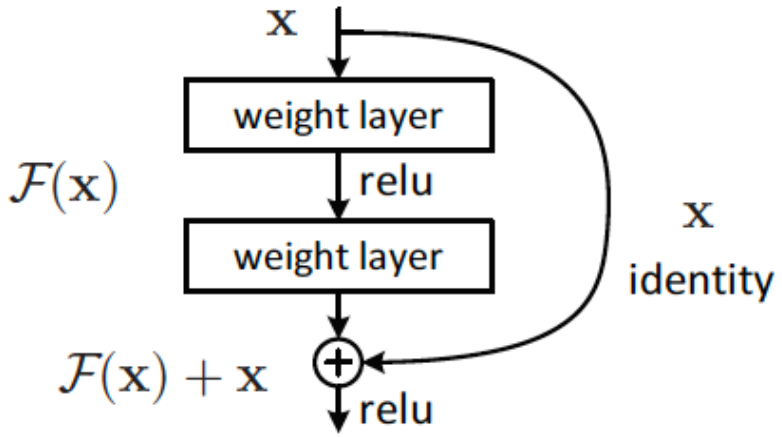
- If multiple layers helps accuracy let's add even more?



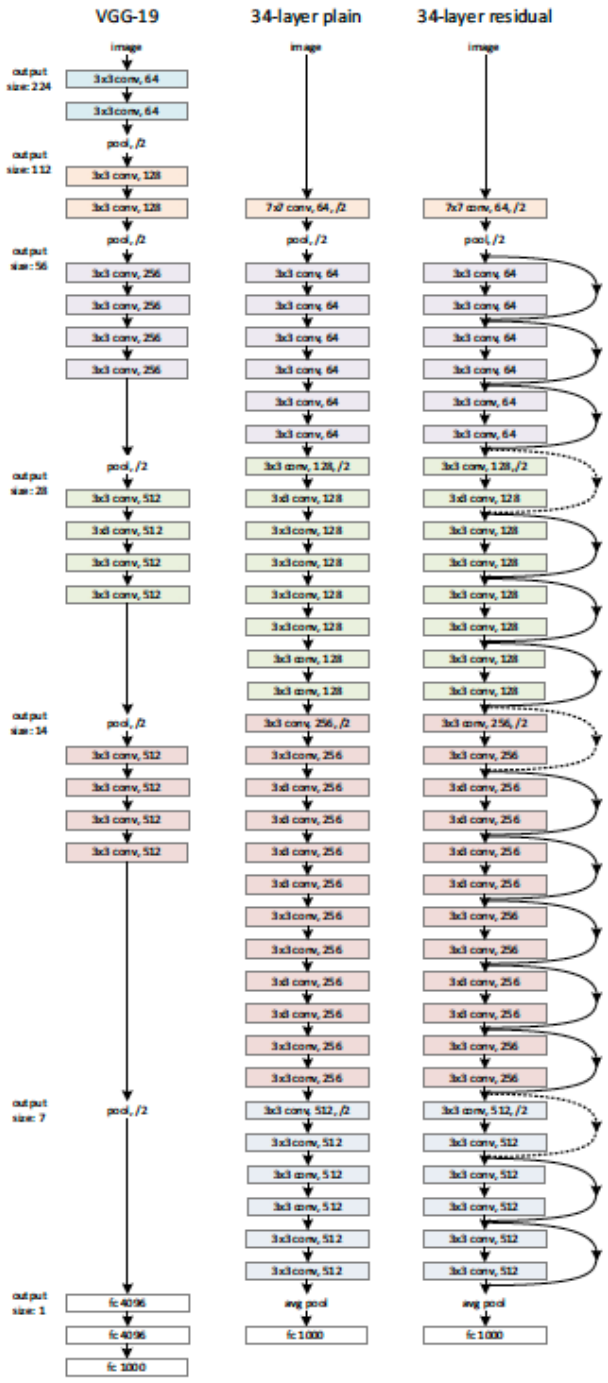
A 20 layer and 56 layer CNN applied to CIFAR-10 image data. Deeper is worse.

Source: He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition. 2016 (pp. 770-778).

Solution for deeper networks: deep residual learning -> Resnet



The input x, through a "skip connection" is used after skipping a Layer. F(x) = relu($\mathbf{x_a}$ **WL**) where $\mathbf{x_a}$ is the input vector from above
and **WL** is the Weight vector for the first layer above.
We add the skip x and Relu(z) = z if z >0, else 0.



No more parameters for Resnet than plain
Top-1 error (%, 10-crop testing) on ImageNet validation.

	Plain	Resnet
18 layers	27.94	27.88
34 layers	28.54	25.03

How to use DNN's or CNN's

1. Use GPU's for training
2. Start with the architecture of a pre-trained model in Matlab there is matconvnet, http://www.vlfeat.org/matconvnet/
3. Use the above or one of
    1. Tensor flow: https://www.tensorflow.org *
    2. Caffe: http://caffe.berkeleyvision.org
    3. PyTorch https://pytorch.org/ *

How to use CNN's

- You can start with a pre-trained CNN, say on ImageNet.
- ImageNet has 14 million examples in 1000 classes like hammer, house cat, plate, etc.
- Features for your data can be extracted from one of the layers in a CNN trained on ImageNet. We have used the layer before the outputs, for example.
- Better, tune a pre-trained CNN if you have a fair amount of data. Fix most weights and tune just a few.
- The last two bullets have to do with transfer learning.

Transfer Learning

- Rectified linear units are used to take the output of a node x and map it to max(0,x).
- You can extract features pre or post RELU.
- The extracted features can be used in any other classifier, though some check for correlations may be in order.
- Also, if you extract from an ImageNet trained network for example you may have up to 4096 features and need to do some reduction.

Transfer Learning

- Ideally, you will tune a trained network with some examples to use features from it.
- How many do you need? Open question, but as many as you can is a good answer.
- What classes to use in training? How about the ones most often active when you send your data through the network.
- If I use something like ImageNet for a CT or MRI, what channel?
  - One of R, G, B – open question. All three could be best.

Transfer Learning Example

- 40 Adenocarcinoma images.
  - 20 Stage IV and 20 Stage I/II (Short/Long survival)
  - Can we classify them?
- Send images through the R channel in CNN-F of Matconvnet trained on ImageNet
- Use 56x56 window around the tumor
- Take 4096 features from 7^{th} layer, select 10 with Symmetric Uncertainty
- Accuracy of 77.5% is as good as using 219 typical features

How to use CNN's

- Want to build a CNN or DNN from scratch?
  - Choose your favorite type of CNN/DNN and get access to a GPU machine
  - You would want about as many examples as weights (more if possible)
  - An ImageNet size CNN has on the order of 4 million weights and can take weeks to train with a powerful GPU cluster (though someone did 14M examples in < 1 week
  - You can do some image rotations and other operations to generate more examples (augmentation)
  - In medical imaging 4M examples seems a stretch, but you can try smaller networks (smaller number of images)

You have reached the end of the lecture.

References:

Figures from Online Book Neural Networks and Deep Learning by Michael Nielsen: http://neuralnetworksanddeeplearning.com/chap6.html and The Asimov Institute – Neural network zoo http://www.asimovinstitute.org/neural-network-zoo/