A.I. Wiki

A Beginner's Guide to Convolutional Neural Networks (CNNs)

Contents

Introduction to Deep Convolutional Neural Networks

Convolutional neural networks are deep artificial neural networks that are used primarily to classify images (e.g. name what they see), cluster them by similarity (photo search), and perform object recognition within scenes. They are algorithms that can identify faces, individuals, street signs, tumors, platypuses and many other aspects of visual data.

Convolutional networks perform optical character recognition (OCR) to digitize text and make natural-language processing possible on analog and hand-written documents, where the images are symbols to be transcribed. CNNs can also be applied to sound when it is represented visually as a spectrogram. More recently, convolutional networks have been applied directly to text analytics as well as graph data with graph convolutional networks.

The efficacy of convolutional nets (ConvNets or CNNs) in image recognition is one of the main reasons why the world has woken up to the efficacy of deep learning. They are powering major advances in computer vision (CV), which has obvious applications for self-driving cars, robotics, drones, security, medical diagnoses, and treatments for the visually impaired.

Images Are 4-D Tensors?

Convolutional neural networks ingest and process images as tensors, and tensors are matrices of numbers with additional dimensions.

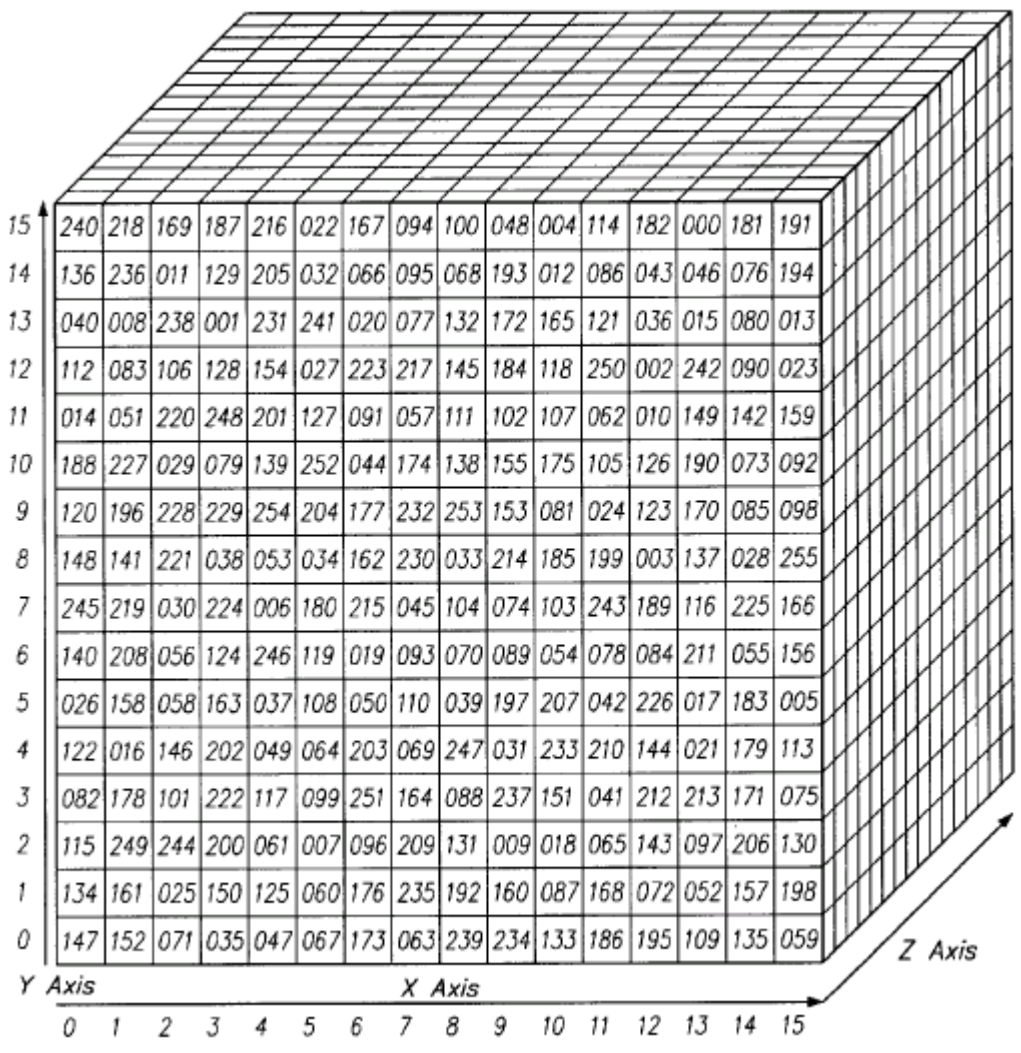
They can be hard to visualize, so let's approach them by analogy. A scalar is just a number, such as 7; a vector is a list of numbers (e.g., `[7,8,9]`); and a matrix is a rectangular grid of numbers occupying several rows and columns like a spreadsheet. Geometrically, if a scalar is a zero-dimensional point, then a vector is a one-dimensional line, a matrix is a two-dimensional plane, a stack of matrices is a three-dimensional cube, and when each element of those matrices has a stack of *feature maps* atttached to it, you enter the fourth dimension. For reference, here's a 2 x 2 matrix:

```
[ 1, 2 ]
[ 5, 8 ]
```

A tensor encompasses the dimensions beyond that 2-D plane. You can easily picture a three-dimensional tensor, with the array of numbers arranged in a cube. Here's a 2 x 3 x 2 tensor presented flatly (picture the bottom element of each 2-element array extending along the z-axis to intuitively grasp why it's called a 3-dimensional array):

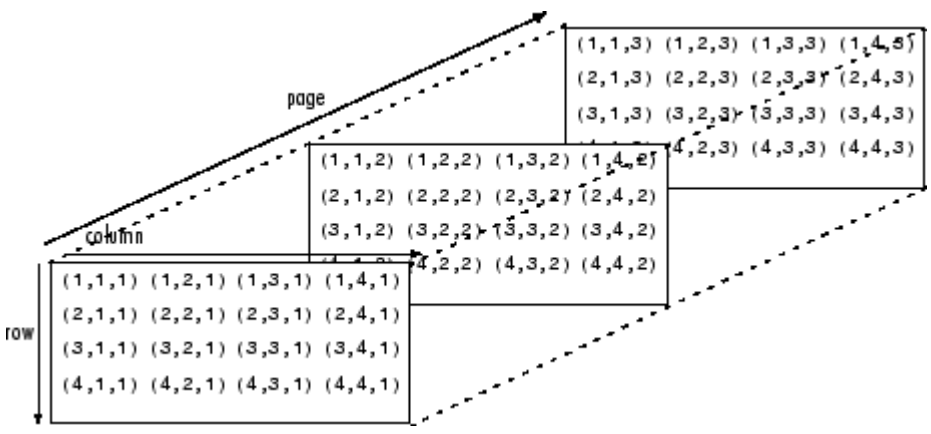
$$\begin{pmatrix} \begin{pmatrix} 2 \\ 3 \end{pmatrix} & \begin{pmatrix} 3 \\ 5 \end{pmatrix} & \begin{pmatrix} 4 \\ 7 \end{pmatrix} \\ \begin{pmatrix} 3 \\ 4 \end{pmatrix} & \begin{pmatrix} 4 \\ 6 \end{pmatrix} & \begin{pmatrix} 5 \\ 8 \end{pmatrix} \end{pmatrix}$$

In code, the tensor above would appear like this: `[[[2,3],[3,5],[4,7]],[[3,4],[4,6],[5,8]]]`. And here's a visual:



In other words, tensors are formed by arrays nested within arrays, and that nesting can go on infinitely, accounting for an arbitrary number of dimensions far greater than what we can visualize spatially. A 4-D

tensor would simply replace each of these scalars with an array nested one level deeper. Convolutional networks deal in 4-D tensors like the one below (notice the nested array).



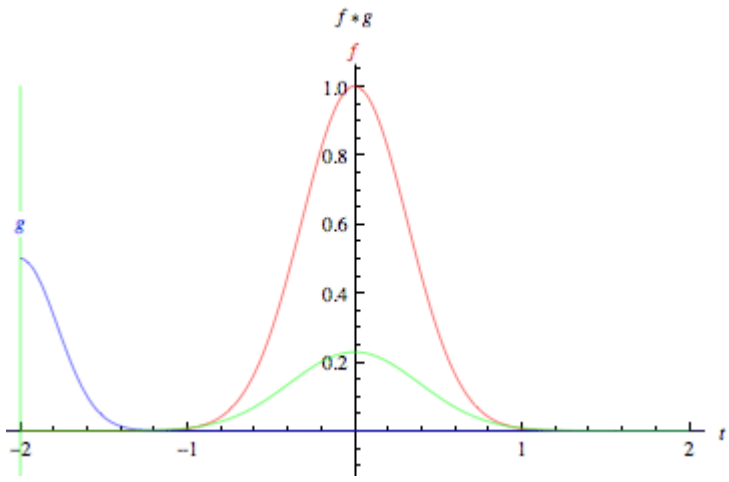
ND4J and Deeplearning4j use `NDArray` synonymously with tensor, or multi-dimensional array. A tensor's dimensionality `(1,2,3…n)` is called its order; i.e. a fifth-order tensor would have five dimensions.

The width and height of an image are easily understood. The depth is necessary because of how colors are encoded. Red-Green-Blue (RGB) encoding, for example, produces an image three layers deep. Each layer is called a "channel", and through convolution it produces a stack of feature maps (explained below), which exist in the fourth dimension, just down the street from time itself. (Features are just details of images, like a line or curve, that convolutional networks create maps of.)

So instead of thinking of images as two-dimensional areas, in convolutional nets they are treated as four-dimensional volumes. These ideas will be explored more thoroughly below.

Convolutional Definition

From the Latin *convolvere*, "to convolve" means to roll together. For mathematical purposes, a convolution is the integral measuring how much two functions overlap as one passes over the other. Think of a convolution as a way of mixing two functions by multiplying them.



Credit: Mathworld. "The green curve shows the convolution of the blue and red curves as a function of t, the position indicated by the vertical green line. The gray region indicates the product `g(tau)f(t−tau)` as a function of t, so its area as a function of t is precisely the convolution."

Look at the tall, narrow bell curve standing in the middle of a graph. The integral is the area under that curve. Near it is a second bell curve that is shorter and wider, drifting slowly from the left side of the graph to the right. The product of those two functions' overlap at each point along the x-axis is their convolution. So in a sense, the two functions are being "rolled together."

With image analysis, the static, underlying function (the equivalent of the immobile bell curve) is the input image being analyzed, and the second, mobile function is known as the filter, because it picks up a signal or feature in the image. The two functions relate through multiplication. To visualize convolutions as matrices rather than as bell curves, please see Andrej Karpathy's excellent animation under the heading "Convolution Demo."

The next thing to understand about convolutional nets is that they are passing *many* filters over a single image, each one picking up a different signal. At a fairly early layer, you could imagine them as passing a horizontal line filter, a vertical line filter, and a diagonal line filter to create a map of the edges in the image.

Convolutional networks take those filters, slices of the image's feature space, and map them one by one; that is, they create a map of each place that feature occurs. By learning different portions of a feature space, convolutional nets allow for easily scalable and robust feature engineering.

(Note that convolutional nets analyze images differently than RBMs. While RBMs learn to reconstruct and identify the features of each image as a whole, convolutional nets learn images in pieces that we call feature maps.)

So convolutional networks perform a sort of search. Picture a small magnifying glass sliding left to right across a larger image, and recommencing at the left once it reaches the end of one pass (like typewriters do). That moving window is capable recognizing only one thing, say, a short vertical line. Three dark pixels stacked atop one another. It moves that vertical-line-recognizing filter over the actual pixels of the image, looking for matches.

Each time a match is found, it is mapped onto a feature space particular to that visual element. In that space, the location of each vertical line match is recorded, a bit like birdwatchers leave pins in a map to mark where they last saw a great blue heron. A convolutional net runs many, many searches over a single image – horizontal lines, diagonal ones, as many as there are visual elements to be sought.

Convolutional nets perform more operations on input than just convolutions themselves.

After a convolutional layer, input is passed through a nonlinear transform such as *tanh* or *rectified linear* unit, which will squash input values into a range between -1 and 1.

How Convolutional Neural Networks Work

The first thing to know about convolutional networks is that they don't perceive images like humans do. Therefore, you are going to have to think in a different way about what an image means as it is fed to and processed by a convolutional network.

Convolutional networks perceive images as volumes; i.e. three-dimensional objects, rather than flat canvases to be measured only by width and height. That's because digital color images have a red-blue-green (RGB) encoding, mixing those three colors to produce the color spectrum humans perceive. A convolutional network ingests such images as three separate strata of color stacked one on top of the other.

So a convolutional network receives a normal color image as a rectangular box whose width and height are measured by the number of pixels along those dimensions, and whose depth is three layers deep, one for each letter in RGB. Those depth layers are referred to as *channels*.

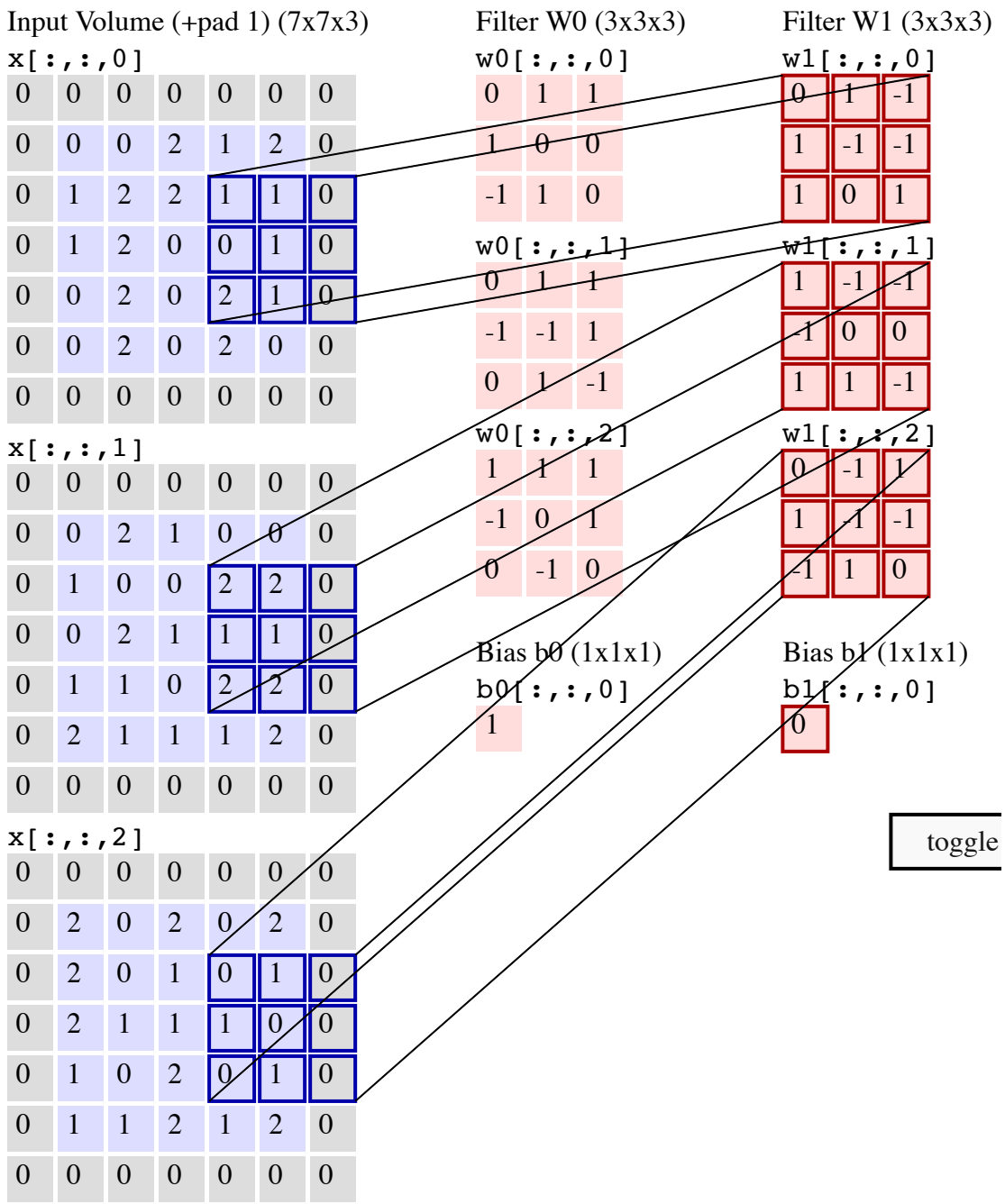
As images move through a convolutional network, we will describe them in terms of input and output volumes, expressing them mathematically as matrices of multiple dimensions in this form: 30x30x3. From layer to layer, their dimensions change for reasons that will be explained below.

You will need to pay close attention to the precise measures of each dimension of the image volume, because they are the foundation of the linear algebra operations used to process images.

Now, for each pixel of an image, the intensity of R, G and B will be expressed by a number, and that number will be an element in one of the three, stacked two-dimensional matrices, which together form the image volume.

Those numbers are the initial, raw, sensory features being fed into the convolutional network, and the ConvNets purpose is to find which of those numbers are significant signals that actually help it classify images more accurately. (Just like other feedforward networks we have discussed.)

Rather than focus on one pixel at a time, a convolutional net takes in square patches of pixels and passes them through a *filter*. That filter is also a square matrix smaller than the image itself, and equal in size to the patch. It is also called a *kernel*, which will ring a bell for those familiar with support-vector machines, and the job of the filter is to find patterns in the pixels.



Credit for this excellent animation goes to Andrej Karpathy.

Imagine two matrices. One is 30x30, and another is 3x3. That is, the filter covers one-hundredth of one image channel's surface area.

We are going to take the dot product of the filter with this patch of the image channel. If the two matrices have high values in the same positions, the dot product's output will be high. If they don't, it will be low. In this way, a single value – the output of the dot product – can tell us whether the pixel pattern in the underlying image matches the pixel pattern expressed by our filter.

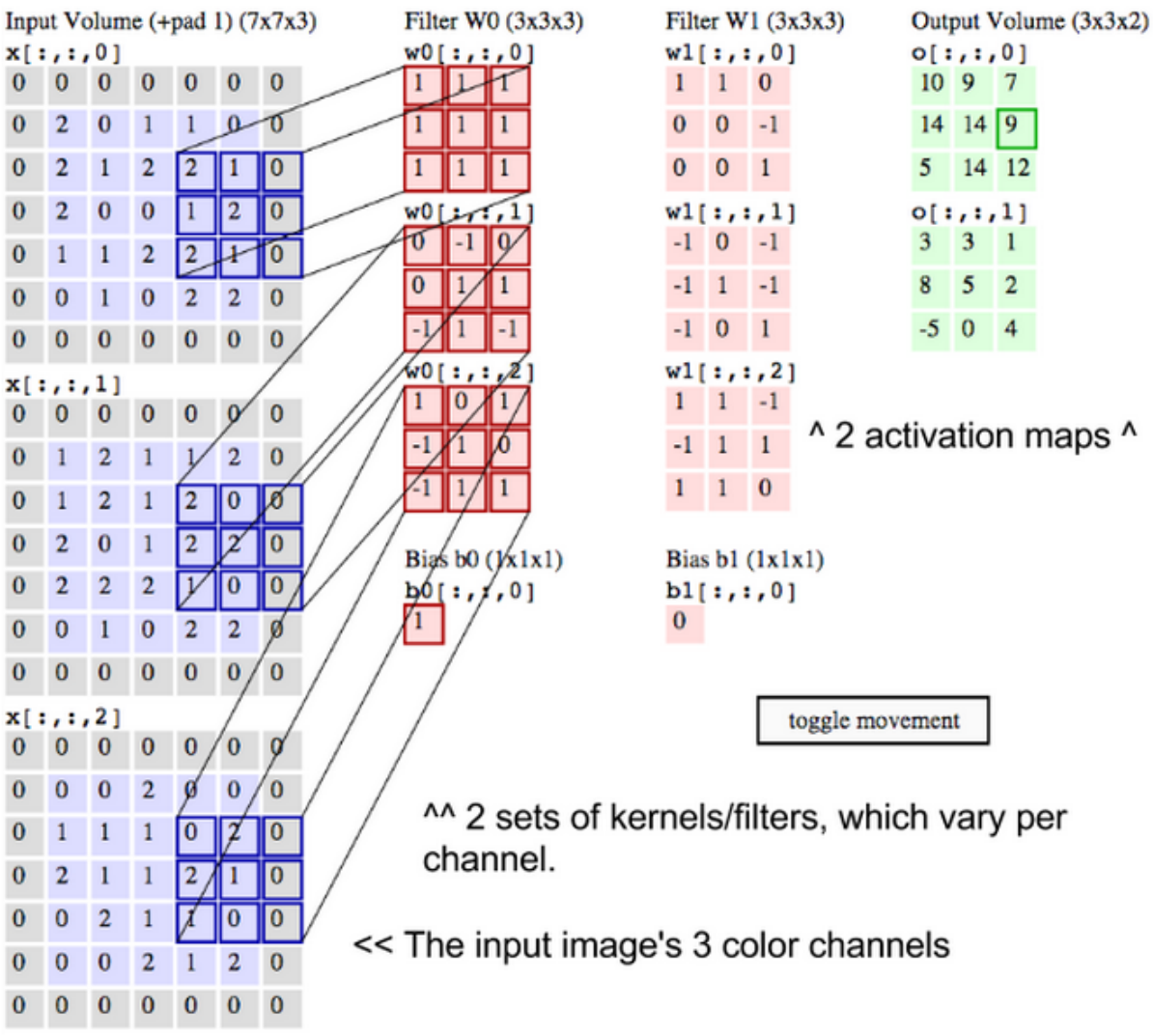
Let's imagine that our filter expresses a horizontal line, with high values along its second row and low values in the first and third rows. Now picture that we start in the upper lefthand corner of the underlying image, and we move the filter across the image step by step until it reaches the upper righthand corner. The size of the step is known as *stride*. You can move the filter to the right one column at a time, or you can choose to make larger steps.

At each step, you take another dot product, and you place the results of that dot product in a third matrix known as an *activation map*. The width, or number of columns, of the activation map is equal to the number of steps the filter takes to traverse the underlying image. Since larger strides lead to fewer steps, a big stride will produce a smaller activation map. This is important, because the size of the matrices that convolutional networks process and produce at each

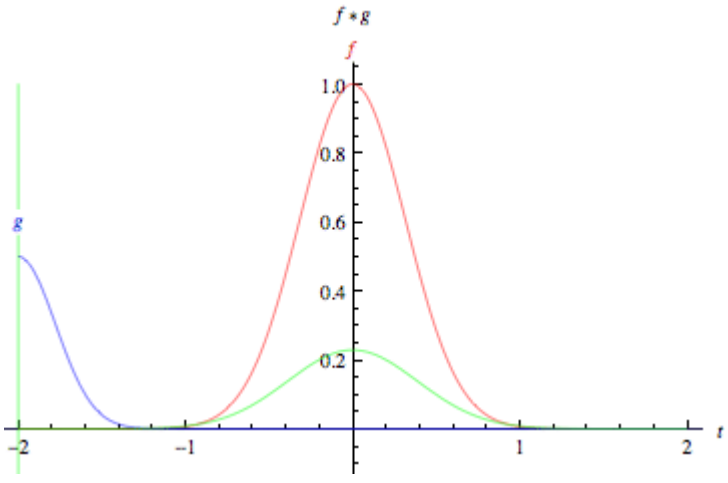
layer is directly proportional to how computationally expensive they are and how much time they take to train. A larger stride means less time and compute.

A filter superimposed on the first three rows will slide across them and then begin again with rows 4-6 of the same image. If it has a stride of three, then it will produce a matrix of dot products that is 10x10. That same filter representing a horizontal line can be applied to all three channels of the underlying image, R, G and B. And the three 10x10 activation maps can be added together, so that the aggregate activation map for a horizontal line on all three channels of the underlying image is also 10x10.

Now, because images have lines going in many directions, and contain many different kinds of shapes and pixel patterns, you will want to slide other filters across the underlying image in search of those patterns. You could, for example, look for 96 different patterns in the pixels. Those 96 patterns will create a stack of 96 activation maps, resulting in a new volume that is 10x10x96. In the diagram below, we've relabeled the input image, the kernels and the output activation maps to make sure we're clear.



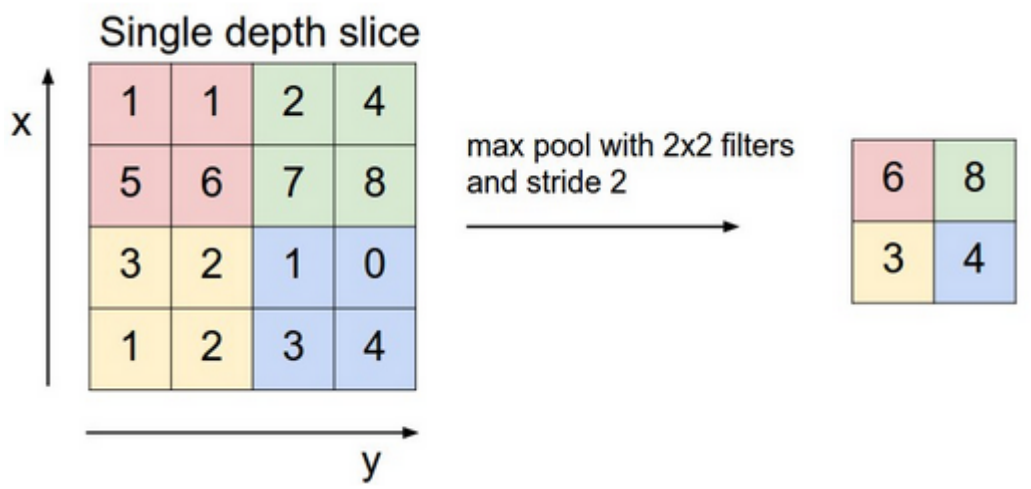
What we just described is a convolution. You can think of Convolution as a fancy kind of multiplication used in signal processing. Another way to think about the two matrices creating a dot product is as two functions. The image is the underlying function, and the filter is the function you roll over it.



One of the main problems with images is that they are high-dimensional, which means they cost a lot of time and computing power to process. Convolutional networks are designed to reduce the dimensionality of images in a variety of ways. Filter stride is one way to reduce dimensionality. Another way is through downsampling.

Max Pooling/Downsampling with CNNs

The next layer in a convolutional network has three names: max pooling, downsampling and subsampling. The activation maps are fed into a downsampling layer, and like convolutions, this method is applied one patch at a time. In this case, max pooling simply takes the largest value from one patch of an image, places it in a new matrix next to the max values from other patches, and discards the rest of the information contained in the activation maps.



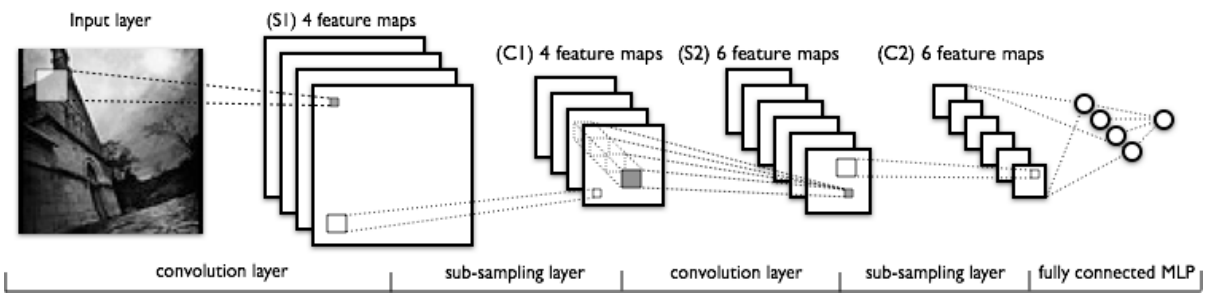
Credit to [Andrej Karpathy](#).

Only the locations on the image that showed the strongest correlation to each feature (the maximum value) are preserved, and those maximum values combine to form a lower-dimensional space.

Much information about lesser values is lost in this step, which has spurred research into alternative methods. But downsampling has the advantage, precisely because information is lost, of decreasing the amount of storage and processing required.

Alternating Layers

The image below is another attempt to show the sequence of transformations involved in a typical convolutional network.



From left to right you see:

- The actual input image that is scanned for features. The light rectangle is the filter that passes over it.
- Activation maps stacked atop one another, one for each filter you employ. The larger rectangle is one patch to be downsampled.
- The activation maps condensed through downsampling.
- A new set of activation maps created by passing filters over the first downsampled stack.
- The second downsampling, which condenses the second set of activation maps.
- A fully connected layer that classifies output with one label per node.

As more and more information is lost, the patterns processed by the convolutional net become more abstract and grow more distant from visual patterns we recognize as humans. So forgive yourself, and us, if convolutional networks do not offer easy intuitions as they grow deeper.

DL4J Code Example

Here’s one example of how you might configure a ConvNet with Deeplearning4j:

```
import java.io.File;


/**
 * Created by agibsonccc on 9/16/15.
 */
public class LenetMnistExample {
    private static final Logger log =
LoggerFactory.getLogger(LenetMnistExample.class);


    public static void main(String[] args) throws Exception {
        int nChannels = 1; // Number of input channels
        int outputNum = 10; // The number of possible outcomes
        int batchSize = 64; // Test batch size
        int nEpochs = 1; // Number of training epochs
        int seed = 123; //

        /*
            Create an iterator using the batch size for one
iteration
         */
        log.info("Load data....");
        DataSetIterator mnistTrain = new
MnistDataSetIterator(batchSize,true,12345);
        DataSetIterator mnistTest = new
MnistDataSetIterator(batchSize,false,12345);


        /*
            Construct the neural network
         */
        log.info("Build model....");

        MultiLayerConfiguration conf = new
NeuralNetConfiguration.Builder()
                .seed(seed)
                .l2(0.0005)
                .weightInit(WeightInit.XAVIER)
                .updater(new Adam(1e-3))
                .list()
                .layer(new ConvolutionLayer.Builder(5, 5)
                        //nIn and nOut specify depth. nIn here is
the nChannels and nOut is the number of filters to be applied
                        .nIn(nChannels)
                        .stride(1,1)
                        .nOut(20)
                        .activation(Activation.IDENTITY)
                        .build())
                .layer(new
SubsamplingLayer.Builder(PoolingType.MAX)
                        .kernelSize(2,2)
                        .stride(2,2)
```

```
                        .build())
                .layer(new ConvolutionLayer.Builder(5, 5)
                        //Note that nIn need not be specified in
later layers
                        .stride(1,1)
                        .nOut(50)
                        .activation(Activation.IDENTITY)
                        .build())
                .layer(new
SubsamplingLayer.Builder(PoolingType.MAX)
                        .kernelSize(2,2)
                        .stride(2,2)
                        .build())
                .layer(new
DenseLayer.Builder().activation(Activation.RELU)
                        .nOut(500).build())
                .layer(new
OutputLayer.Builder(LossFunctions.LossFunction.NEGATIVELOGLIKELIHO
                        .nOut(outputNum)
                        .activation(Activation.SOFTMAX)
                        .build())

.setInputType(InputType.convolutionalFlat(28,28,1)) //See note
below
                .build();

        /*
        Regarding the
.setInputType(InputType.convolutionalFlat(28,28,1)) line: This
does a few things.
        (a) It adds preprocessors, which handle things like the
transition between the convolutional/subsampling layers
            and the dense layer
        (b) Does some additional configuration validation
        (c) Where necessary, sets the nIn (number of input
neurons, or input depth in the case of CNNs) values for each
            layer based on the size of the previous layer (but it
won't override values manually set by the user)

        InputTypes can be used with other layer types too (RNNs,
MLPs etc) not just CNNs.
        For normal images (when using ImageRecordReader) use
InputType.convolutional(height,width,depth).
        MNIST record reader is a special case, that outputs 28x28
pixel grayscale (nChannels=1) images, in a "flattened"
        row vector format (i.e., 1x784 vectors), hence the
"convolutionalFlat" input type used here.
        */

        MultiLayerNetwork model = new MultiLayerNetwork(conf);
```

```
        model.init();


        log.info("Train model...");
        model.setListeners(new ScoreIterationListener(10), new
EvaluativeListener(mnistTest, 1, InvocationType.EPOCH_END));
//Print score every 10 iterations and evaluate on test set every
epoch
        model.fit(mnistTrain, nEpochs);
```

This Gist brought to you by gist-it. view raw

java/org/deeplearning4j/examples/convolution/LenetMnistExample.java

All Deeplearning4j examples of convolutional networks are available here.

Other Resources

To see DL4J convolutional neural networks in action, please run our examples after following the instructions on the Quickstart page.

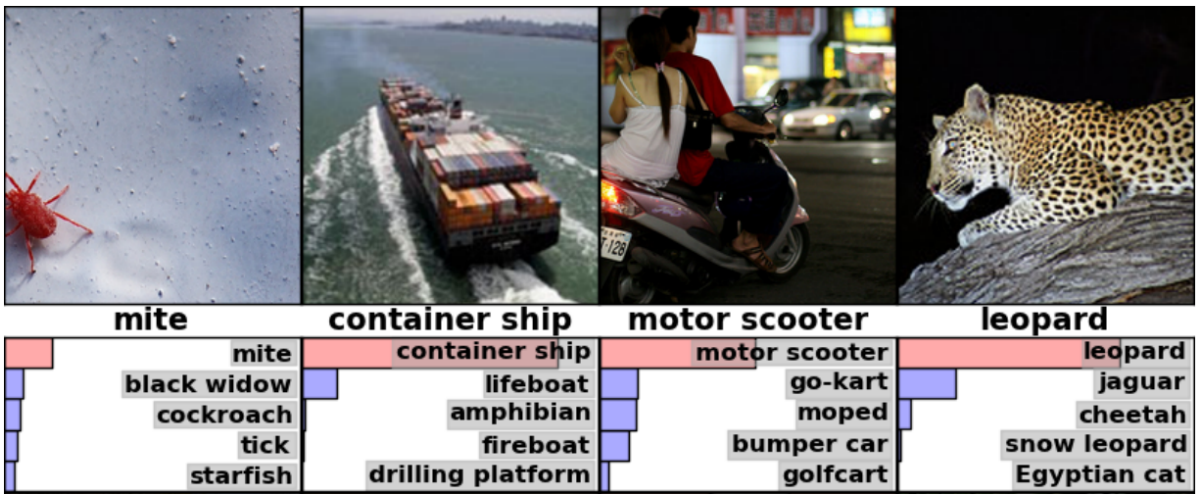
Skymind wraps NVIDIA's cuDNN and integrates with OpenCV. Our convolutional nets run on distributed GPUs using Spark, making them among the fastest in the world. You can learn how to build a image recognition web app with VGG16 here and how to deploy CNNs to Android here.

Resources Table of Contents

    - Framework
    - Applications
- Tutorials
- Blogs

Papers

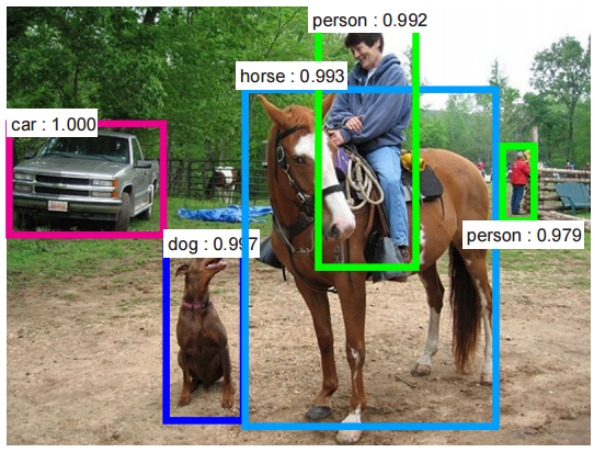
ImageNet Classification

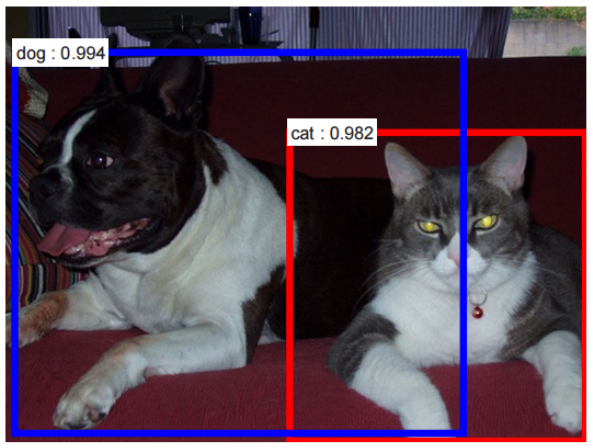


(from Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, NIPS, 2012.)

- Microsoft (Deep Residual Learning) [Paper][Slide]
    - Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Deep Residual Learning for Image Recognition, arXiv:1512.03385.
- Microsoft (PReLu/Weight Initialization) [Paper]
    - Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification, arXiv:1502.01852.
- Batch Normalization [Paper]
    - Sergey Ioffe, Christian Szegedy, Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, arXiv:1502.03167.
- GoogLeNet [Paper]
    - Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, CVPR, 2015.
- VGG-Net [Web] [Paper]
    - Karen Simonyan and Andrew Zisserman, Very Deep Convolutional Networks for Large-Scale Visual Recognition, ICLR, 2015.
- AlexNet [Paper]
    - Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, NIPS, 2012.

Object Detection





(from Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, arXiv:1506.01497.)

- PVANET [Paper] [Code]
  - Kye-Hyeon Kim, Sanghoon Hong, Byungseok Roh, Yeongjae Cheon, Minje Park, PVANET: Deep but Lightweight Neural Networks for Real-time Object Detection, arXiv:1608.08021
- OverFeat, NYU [Paper]
  - OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks, ICLR, 2014.
- R-CNN, UC Berkeley [Paper-CVPR14] [Paper-arXiv14]
  - Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, CVPR, 2014.
- SPP, Microsoft Research [Paper]
  - Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition, ECCV, 2014.
- Fast R-CNN, Microsoft Research [Paper]
  - Ross Girshick, Fast R-CNN, arXiv:1504.08083.
- Faster R-CNN, Microsoft Research [Paper]
  - Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, arXiv:1506.01497.
- R-CNN minus R, Oxford [Paper]
  - Karel Lenc, Andrea Vedaldi, R-CNN minus R, arXiv:1506.06981.
- End-to-end people detection in crowded scenes [Paper]
  - Russell Stewart, Mykhaylo Andriluka, End-to-end people detection in crowded scenes, arXiv:1506.04878.
- You Only Look Once: Unified, Real-Time Object Detection [Paper], [Paper Version 2], [C Code], [Tensorflow Code]
  - Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, You Only Look Once: Unified, Real-Time Object Detection, arXiv:1506.02640
  - Joseph Redmon, Ali Farhadi (Version 2)
- Inside-Outside Net [Paper]
  - Sean Bell, C. Lawrence Zitnick, Kavita Bala, Ross Girshick, Inside-Outside Net: Detecting Objects in Context with Skip Pooling and Recurrent Neural Networks
- Deep Residual Network (Current State-of-the-Art) [Paper]
  - Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Deep Residual Learning for Image Recognition

- Weakly Supervised Object Localization with Multi-fold Multiple Instance Learning [Paper]
- R-FCN [Paper] [Code]
  - Jifeng Dai, Yi Li, Kaiming He, Jian Sun, R-FCN: Object Detection via Region-based Fully Convolutional Networks
- SSD [Paper] [Code]
  - Wei Liu1, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg, SSD: Single Shot MultiBox Detector, arXiv:1512.02325
- Speed/accuracy trade-offs for modern convolutional object detectors [Paper]
  - Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, Kevin Murphy, Google Research, arXiv:1611.10012

Video Classification

- Nicolas Ballas, Li Yao, Pal Chris, Aaron Courville, “Delving Deeper into Convolutional Networks for Learning Video Representations”, ICLR 2016. [Paper]
- Michael Mathieu, camille couprie, Yann Lecun, “Deep Multi Scale Video Prediction Beyond Mean Square Error”, ICLR 2016. [Paper]

Object Tracking

- Seunghoon Hong, Tackgeun You, Suha Kwak, Bohyung Han, Online Tracking by Learning Discriminative Saliency Map with Convolutional Neural Network, arXiv:1502.06796. [Paper]
- Hanxi Li, Yi Li and Fatih Porikli, DeepTrack: Learning Discriminative Feature Representations by Convolutional Neural Networks for Visual Tracking, BMVC, 2014. [Paper]
- N Wang, DY Yeung, Learning a Deep Compact Image Representation for Visual Tracking, NIPS, 2013. [Paper]
- Chao Ma, Jia-Bin Huang, Xiaokang Yang and Ming-Hsuan Yang, Hierarchical Convolutional Features for Visual Tracking, ICCV 2015 [Paper] [Code]
- Lijun Wang, Wanli Ouyang, Xiaogang Wang, and Huchuan Lu, Visual Tracking with fully Convolutional Networks, ICCV 2015 [Paper] [Code]
- Hyeonseob Namand Bohyung Han, Learning Multi-Domain Convolutional Neural Networks for Visual Tracking, [Paper] [Code] [Project Page]

Low-Level Vision

Super-Resolution

- Iterative Image Reconstruction
  - Sven Behnke: Learning Iterative Image Reconstruction. IJCAI, 2001. [Paper]
  - Sven Behnke: Learning Iterative Image Reconstruction in the Neural Abstraction Pyramid. International Journal of

Computational Intelligence and Applications, vol. 1, no. 4, pp. 427-438, 2001. [Paper]

- Super-Resolution (SRCNN) [Web] [Paper-ECCV14] [Paper-arXiv15]
  - Chao Dong, Chen Change Loy, Kaiming He, Xiaoou Tang, Learning a Deep Convolutional Network for Image Super-Resolution, ECCV, 2014.
  - Chao Dong, Chen Change Loy, Kaiming He, Xiaoou Tang, Image Super-Resolution Using Deep Convolutional Networks, arXiv:1501.00092.
- Very Deep Super-Resolution
  - Jiwon Kim, Jung Kwon Lee, Kyoung Mu Lee, Accurate Image Super-Resolution Using Very Deep Convolutional Networks, arXiv:1511.04587, 2015. [Paper]
- Deeply-Recursive Convolutional Network
  - Jiwon Kim, Jung Kwon Lee, Kyoung Mu Lee, Deeply-Recursive Convolutional Network for Image Super-Resolution, arXiv:1511.04491, 2015. [Paper]
- Casade-Sparse-Coding-Network
  - Zhaowen Wang, Ding Liu, Wei Han, Jianchao Yang and Thomas S. Huang, Deep Networks for Image Super-Resolution with Sparse Prior. ICCV, 2015. [Paper] [Code]
- Perceptual Losses for Super-Resolution
  - Justin Johnson, Alexandre Alahi, Li Fei-Fei, Perceptual Losses for Real-Time Style Transfer and Super-Resolution, arXiv:1603.08155, 2016. [Paper] [Supplementary]
- SRGAN
  - Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, Wenzhe Shi, Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network, arXiv:1609.04802v3, 2016. [Paper]
- Others
  - Osendorfer, Christian, Hubert Soyer, and Patrick van der Smagt, Image Super-Resolution with Fast Approximate Convolutional Sparse Coding, ICONIP, 2014. [Paper ICONIP-2014]

Other Applications

- Optical Flow (FlowNet) [Paper]
  - Philipp Fischer, Alexey Dosovitskiy, Eddy Ilg, Philip Häusser, Caner Hazırbaş, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, Thomas Brox, FlowNet: Learning Optical Flow with Convolutional Networks, arXiv:1504.06852.
- Compression Artifacts Reduction [Paper-arXiv15]
  - Chao Dong, Yubin Deng, Chen Change Loy, Xiaoou Tang, Compression Artifacts Reduction by a Deep Convolutional Network, arXiv:1504.06993.
- Blur Removal

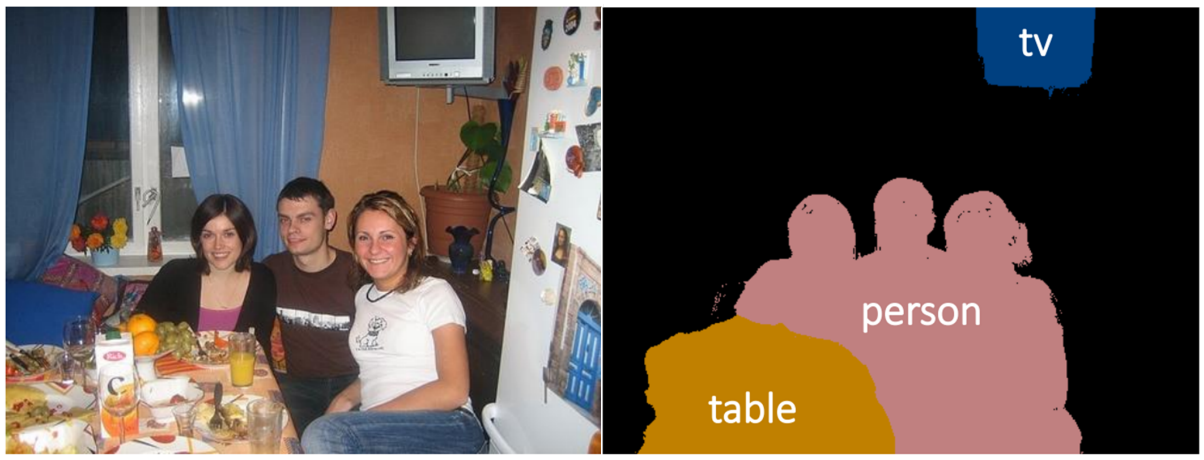
    - Christian J. Schuler, Michael Hirsch, Stefan Harmeling, Bernhard Schölkopf, Learning to Deblur, arXiv:1406.7444 [Paper]
    - Jian Sun, Wenfei Cao, Zongben Xu, Jean Ponce, Learning a Convolutional Neural Network for Non-uniform Motion Blur Removal, CVPR, 2015 [Paper]
- Image Deconvolution [Web] [Paper]
    - Li Xu, Jimmy SJ. Ren, Ce Liu, Jiaya Jia, Deep Convolutional Neural Network for Image Deconvolution, NIPS, 2014.
- Deep Edge-Aware Filter [Paper]
    - Li Xu, Jimmy SJ. Ren, Qiong Yan, Renjie Liao, Jiaya Jia, Deep Edge-Aware Filters, ICML, 2015.
- Computing the Stereo Matching Cost with a Convolutional Neural Network [Paper]
    - Jure Žbontar, Yann LeCun, Computing the Stereo Matching Cost with a Convolutional Neural Network, CVPR, 2015.
- Colorful Image Colorization Richard Zhang, Phillip Isola, Alexei A. Efros, ECCV, 2016 [Paper], [Code]
- Ryan Dahl, [Blog]
- Feature Learning by Inpainting[Paper][Code]
    - Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, Alexei A. Efros, Context Encoders: Feature Learning by Inpainting, CVPR, 2016

Edge Detection

(from Gedas Bertasius, Jianbo Shi, Lorenzo Torresani, DeepEdge: A Multi-Scale Bifurcated Deep Network for Top-Down Contour Detection, CVPR, 2015.)

- Holistically-Nested Edge Detection [Paper] [Code]
    - Saining Xie, Zhuowen Tu, Holistically-Nested Edge Detection, arXiv:1504.06375.
- DeepEdge [Paper]
    - Gedas Bertasius, Jianbo Shi, Lorenzo Torresani, DeepEdge: A Multi-Scale Bifurcated Deep Network for Top-Down Contour Detection, CVPR, 2015.
- DeepContour [Paper]
    - Wei Shen, Xinggang Wang, Yan Wang, Xiang Bai, Zhijiang Zhang, DeepContour: A Deep Convolutional Feature Learned by Positive-Sharing Loss for Contour Detection, CVPR, 2015.

Semantic Segmentation



(from Jifeng Dai, Kaiming He, Jian Sun, BoxSup: Exploiting Bounding Boxes to Supervise Convolutional Networks for Semantic Segmentation, arXiv:1503.01640.)

- PASCAL VOC2012 Challenge Leaderboard (01 Sep. 2016)

	mean	submission date
SegModel [?]	81.8	23-Aug-2016
RRR-ResNet152-COCO [?]	81.7	25-Aug-2016
CentraleSupelec Deep G-CRF [?]	80.2	12-Aug-2016
CMT-FCN-ResNet-CRF [?]	80.0	02-Aug-2016
DeepLabv2-CRF [?]	79.7	06-Jun-2016
CASIA_SegResNet_CRF_COCO [?]	79.3	03-Jun-2016
LRR_4x_ResNet_COCO [?]	79.3	18-Jul-2016
Adelaide_VeryDeep_FCN_VOC [?]	79.1	13-May-2016
LRR_4x_COCO [?]	78.7	16-Jun-2016
CASIA_IVA_OASeg [?]	78.3	21-May-2016
Oxford_TVG_HO_CRF [?]	77.9	16-Mar-2016
Adelaide_Context_CNN_CRF_COCO [?]	77.8	06-Nov-2015
CUHK_DPN_COCO [?]	77.5	22-Sep-2015
Adelaide_Context_CNN_CRF_COCO [?]	77.2	13-Aug-2015
DeepLab-CRF-Attention-DT [?]	76.3	03-Feb-2016
CentraleSuperBoundaries++ [?]	76.0	13-Jan-2016
LRR_4x_de_pyramid_VOC [?]	75.9	07-Jun-2016
DeepLab-CRF-Attention [?]	75.7	03-Feb-2016
Adelaide_Context_CNN_CRF_VOC [?]	75.3	30-Aug-2015
MSRA_BoxSup [?]	75.2	18-May-2015
MERL_UMD_Deep_GCRF_COCO [?]	74.8	15-Jan-2016
POSTECH_DeconvNet_CRF_VOC [?]	74.8	18-Aug-2015
Oxford_TVG_CRF_RNN_COCO [?]	74.7	22-Apr-2015

(from PASCAL VOC2012 leaderboards)

- SEC: Seed, Expand and Constrain
  - Alexander Kolesnikov, Christoph Lampert, Seed, Expand and Constrain: Three Principles for Weakly-Supervised Image Segmentation, ECCV, 2016. [Paper] [Code]
- Adelaide
  - Guosheng Lin, Chunhua Shen, Ian Reid, Anton van dan Hengel, Efficient piecewise training of deep structured models for semantic segmentation, arXiv:1504.01013. [Paper] (1st ranked in VOC2012)
  - Guosheng Lin, Chunhua Shen, Ian Reid, Anton van den Hengel, Deeply Learning the Messages in Message Passing Inference, arXiv:1508.02108. [Paper] (4th ranked in VOC2012)
- Deep Parsing Network (DPN)

    - Ziwei Liu, Xiaoxiao Li, Ping Luo, Chen Change Loy, Xiaoou Tang, Semantic Image Segmentation via Deep Parsing Network, arXiv:1509.02634 / ICCV 2015 [Paper] (2nd ranked in VOC 2012)
- CentraleSuperBoundaries, INRIA [Paper]
    - Iasonas Kokkinos, Surpassing Humans in Boundary Detection using Deep Learning, arXiv:1411.07386 (4th ranked in VOC 2012)
- BoxSup [Paper]
    - Jifeng Dai, Kaiming He, Jian Sun, BoxSup: Exploiting Bounding Boxes to Supervise Convolutional Networks for Semantic Segmentation, arXiv:1503.01640. (6th ranked in VOC2012)
- POSTECH
    - Hyeonwoo Noh, Seunghoon Hong, Bohyung Han, Learning Deconvolution Network for Semantic Segmentation, arXiv:1505.04366. [Paper] (7th ranked in VOC2012)
    - Seunghoon Hong, Hyeonwoo Noh, Bohyung Han, Decoupled Deep Neural Network for Semi-supervised Semantic Segmentation, arXiv:1506.04924. [Paper]
    - Seunghoon Hong,Junhyuk Oh, Bohyung Han, and Honglak Lee, Learning Transferrable Knowledge for Semantic Segmentation with Deep Convolutional Neural Network, arXiv:1512.07928 [Paper] [Project Page]
- Conditional Random Fields as Recurrent Neural Networks [Paper]
    - Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, Philip H. S. Torr, Conditional Random Fields as Recurrent Neural Networks, arXiv:1502.03240. (8th ranked in VOC2012)
- DeepLab
    - Liang-Chieh Chen, George Papandreou, Kevin Murphy, Alan L. Yuille, Weakly-and semi-supervised learning of a DCNN for semantic image segmentation, arXiv:1502.02734. [Paper] (9th ranked in VOC2012)
- Zoom-out [Paper]
    - Mohammadreza Mostajabi, Payman Yadollahpour, Gregory Shakhnarovich, Feedforward Semantic Segmentation With Zoom-Out Features, CVPR, 2015
- Joint Calibration [Paper]
    - Holger Caesar, Jasper Uijlings, Vittorio Ferrari, Joint Calibration for Semantic Segmentation, arXiv:1507.01581.
- Fully Convolutional Networks for Semantic Segmentation [Paper-CVPR15] [Paper-arXiv15]
    - Jonathan Long, Evan Shelhamer, Trevor Darrell, Fully Convolutional Networks for Semantic Segmentation, CVPR, 2015.
- Hypercolumn [Paper]
    - Bharath Hariharan, Pablo Arbelaez, Ross Girshick, Jitendra Malik, Hypercolumns for Object Segmentation and Fine-Grained Localization, CVPR, 2015.

- Deep Hierarchical Parsing
  - Abhishek Sharma, Oncel Tuzel, David W. Jacobs, Deep Hierarchical Parsing for Semantic Segmentation, CVPR, 2015. [Paper]
- Learning Hierarchical Features for Scene Labeling [Paper-ICML12] [Paper-PAMI13]
  - Clement Farabet, Camille Couprie, Laurent Najman, Yann LeCun, Scene Parsing with Multiscale Feature Learning, Purity Trees, and Optimal Covers, ICML, 2012.
  - Clement Farabet, Camille Couprie, Laurent Najman, Yann LeCun, Learning Hierarchical Features for Scene Labeling, PAMI, 2013.
- University of Cambridge [Web]
  - Vijay Badrinarayanan, Alex Kendall and Roberto Cipolla "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation." arXiv preprint arXiv:1511.00561, 2015. [Paper]
- Alex Kendall, Vijay Badrinarayanan and Roberto Cipolla "Bayesian SegNet: Model Uncertainty in Deep Convolutional Encoder-Decoder Architectures for Scene Understanding." arXiv preprint arXiv:1511.02680, 2015. [Paper]
- Princeton
  - Fisher Yu, Vladlen Koltun, "Multi-Scale Context Aggregation by Dilated Convolutions", ICLR 2016, [Paper]
- Univ. of Washington, Allen AI
  - Hamid Izadinia, Fereshteh Sadeghi, Santosh Kumar Divvala, Yejin Choi, Ali Farhadi, "Segment-Phrase Table for Semantic Segmentation, Visual Entailment and Paraphrasing", ICCV, 2015, [Paper]
- INRIA
  - Iasonas Kokkinos, "Pusing the Boundaries of Boundary Detection Using deep Learning", ICLR 2016, [Paper]
- UCSB
  - Niloufar Pourian, S. Karthikeyan, and B.S. Manjunath, "Weakly supervised graph based semantic segmentation by learning communities of image-parts", ICCV, 2015, [Paper]

Visual Attention and Saliency

(from Nian Liu, Junwei Han, Dingwen Zhang, Shifeng Wen, Tianming Liu, Predicting Eye Fixations using Convolutional Neural Networks, CVPR, 2015.)

- Mr-CNN [Paper]
  - Nian Liu, Junwei Han, Dingwen Zhang, Shifeng Wen, Tianming Liu, Predicting Eye Fixations using Convolutional Neural Networks, CVPR, 2015.

- Learning a Sequential Search for Landmarks [Paper]
  - Saurabh Singh, Derek Hoiem, David Forsyth, Learning a Sequential Search for Landmarks, CVPR, 2015.
- Multiple Object Recognition with Visual Attention [Paper]
  - Jimmy Lei Ba, Volodymyr Mnih, Koray Kavukcuoglu, Multiple Object Recognition with Visual Attention, ICLR, 2015.
- Recurrent Models of Visual Attention [Paper]
  - Volodymyr Mnih, Nicolas Heess, Alex Graves, Koray Kavukcuoglu, Recurrent Models of Visual Attention, NIPS, 2014.

Object Recognition

- Weakly-supervised learning with convolutional neural networks [Paper]
  - Maxime Oquab, Leon Bottou, Ivan Laptev, Josef Sivic, Is object localization for free? – Weakly-supervised learning with convolutional neural networks, CVPR, 2015.
- FV-CNN [Paper]
  - Mircea Cimpoi, Subhransu Maji, Andrea Vedaldi, Deep Filter Banks for Texture Recognition and Segmentation, CVPR, 2015.

Human Pose Estimation

- Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh, Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields, CVPR, 2017.
- Leonid Pishchulin, Eldar Insafutdinov, Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, Peter Gehler, and Bernt Schiele, Deepcut: Joint subset partition and labeling for multi person pose estimation, CVPR, 2016.
- Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh, Convolutional pose machines, CVPR, 2016.
- Alejandro Newell, Kaiyu Yang, and Jia Deng, Stacked hourglass networks for human pose estimation, ECCV, 2016.
- Tomas Pfister, James Charles, and Andrew Zisserman, Flowing convnets for human pose estimation in videos, ICCV, 2015.
- Jonathan J. Tompson, Arjun Jain, Yann LeCun, Christoph Bregler, Joint training of a convolutional network and a graphical model for human pose estimation, NIPS, 2014.

Understanding CNN

(from Aravindh Mahendran, Andrea Vedaldi, Understanding Deep Image Representations by Inverting Them, CVPR, 2015.)

- Karel Lenc, Andrea Vedaldi, Understanding image representations by measuring their equivariance and equivalence, CVPR, 2015. [Paper]
- Anh Nguyen, Jason Yosinski, Jeff Clune, Deep Neural Networks are Easily Fooled:High Confidence Predictions for Unrecognizable Images, CVPR, 2015. [Paper]
- Aravindh Mahendran, Andrea Vedaldi, Understanding Deep Image Representations by Inverting Them, CVPR, 2015. [Paper]
- Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, Antonio Torralba, Object Detectors Emerge in Deep Scene CNNs, ICLR, 2015. [arXiv Paper]
- Alexey Dosovitskiy, Thomas Brox, Inverting Visual Representations with Convolutional Networks, arXiv, 2015. [Paper]
- Matthrew Zeiler, Rob Fergus, Visualizing and Understanding Convolutional Networks, ECCV, 2014. [Paper]

Image and Language

Image Captioning

(from Andrej Karpathy, Li Fei-Fei, Deep Visual-Semantic Alignments for Generating Image Description, CVPR, 2015.)

- Pay Less Attention with Lightweight and Dynamic Convolutions
- UCLA / Baidu [Paper]
  - Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Alan L. Yuille, Explain Images with Multimodal Recurrent Neural Networks, arXiv:1410.1090.
- Toronto [Paper]
  - Ryan Kiros, Ruslan Salakhutdinov, Richard S. Zemel, Unifying Visual-Semantic Embeddings with Multimodal Neural Language Models, arXiv:1411.2539.
- Berkeley [Paper]
  - Jeff Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, Trevor Darrell, Long-term Recurrent Convolutional Networks for Visual Recognition and Description, arXiv:1411.4389.
- Google [Paper]
  - Oriol Vinyals, Alexander Toshev, Samy Bengio, Dumitru Erhan, Show and Tell: A Neural Image Caption Generator, arXiv:1411.4555.
- Stanford [Web] [Paper]
  - Andrej Karpathy, Li Fei-Fei, Deep Visual-Semantic Alignments for Generating Image Description, CVPR, 2015.
- UML / UT [Paper]
  - Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, Kate Saenko, Translating Videos

to Natural Language Using Deep Recurrent Neural Networks, NAACL-HLT, 2015.

- CMU / Microsoft [Paper-arXiv] [Paper-CVPR]
  - Xinlei Chen, C. Lawrence Zitnick, Learning a Recurrent Visual Representation for Image Caption Generation, arXiv:1411.5654.
  - Xinlei Chen, C. Lawrence Zitnick, Mind's Eye: A Recurrent Visual Representation for Image Caption Generation, CVPR 2015
- Microsoft [Paper]
  - Hao Fang, Saurabh Gupta, Forrest Iandola, Rupesh Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C. Platt, C. Lawrence Zitnick, Geoffrey Zweig, From Captions to Visual Concepts and Back, CVPR, 2015.
- Univ. Montreal / Univ. Toronto [Web] [Paper]
  - Kelvin Xu, Jimmy Lei Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S. Zemel, Yoshua Bengio, Show, Attend, and Tell: Neural Image Caption Generation with Visual Attention, arXiv:1502.03044 / ICML 2015
- Idiap / EPFL / Facebook [Paper]
  - Remi Lebret, Pedro O. Pinheiro, Ronan Collobert, Phrase-based Image Captioning, arXiv:1502.03671 / ICML 2015
- UCLA / Baidu [Paper]
  - Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Zhiheng Huang, Alan L. Yuille, Learning like a Child: Fast Novel Visual Concept Learning from Sentence Descriptions of Images, arXiv:1504.06692
- MS + Berkeley
  - Jacob Devlin, Saurabh Gupta, Ross Girshick, Margaret Mitchell, C. Lawrence Zitnick, Exploring Nearest Neighbor Approaches for Image Captioning, arXiv:1505.04467 [Paper]
  - Jacob Devlin, Hao Cheng, Hao Fang, Saurabh Gupta, Li Deng, Xiaodong He, Geoffrey Zweig, Margaret Mitchell, Language Models for Image Captioning: The Quirks and What Works, arXiv:1505.01809 [Paper]
- Adelaide [Paper]
  - Qi Wu, Chunhua Shen, Anton van den Hengel, Lingqiao Liu, Anthony Dick, Image Captioning with an Intermediate Attributes Layer, arXiv:1506.01144
- Tilburg [Paper]
  - Grzegorz Chrupala, Akos Kadar, Afra Alishahi, Learning language through pictures, arXiv:1506.03694
- Univ. Montreal [Paper]
  - Kyunghyun Cho, Aaron Courville, Yoshua Bengio, Describing Multimedia Content using Attention-based Encoder-Decoder Networks, arXiv:1507.01053
- Cornell [Paper]
  - Jack Hessel, Nicolas Savva, Michael J. Wilber, Image Representations and New Domains in Neural Image Captioning, arXiv:1508.02091

- MS + City Univ. of HongKong [Paper]
    - Ting Yao, Tao Mei, and Chong-Wah Ngo, “Learning Query and Image Similarities with Ranking Canonical Correlation Analysis”, ICCV, 2015

Video Captioning

- Berkeley [Web] [Paper]
    - Jeff Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, Trevor Darrell, Long-term Recurrent Convolutional Networks for Visual Recognition and Description, CVPR, 2015.
- UT / UML / Berkeley [Paper]
    - Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, Kate Saenko, Translating Videos to Natural Language Using Deep Recurrent Neural Networks, arXiv:1412.4729.
- Microsoft [Paper]
    - Yingwei Pan, Tao Mei, Ting Yao, Houqiang Li, Yong Rui, Joint Modeling Embedding and Translation to Bridge Video and Language, arXiv:1505.01861.
- UT / Berkeley / UML [Paper]
    - Subhashini Venugopalan, Marcus Rohrbach, Jeff Donahue, Raymond Mooney, Trevor Darrell, Kate Saenko, Sequence to Sequence–Video to Text, arXiv:1505.00487.
- Univ. Montreal / Univ. Sherbrooke [Paper]
    - Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, Aaron Courville, Describing Videos by Exploiting Temporal Structure, arXiv:1502.08029
- MPI / Berkeley [Paper]
    - Anna Rohrbach, Marcus Rohrbach, Bernt Schiele, The Long-Short Story of Movie Description, arXiv:1506.01698
- Univ. Toronto / MIT [Paper]
    - Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, Sanja Fidler, Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books, arXiv:1506.06724
- Univ. Montreal [Paper]
    - Kyunghyun Cho, Aaron Courville, Yoshua Bengio, Describing Multimedia Content using Attention-based Encoder-Decoder Networks, arXiv:1507.01053
- TAU / USC [paper]
    - Dotan Kaufman, Gil Levi, Tal Hassner, Lior Wolf, Temporal Tessellation for Video Annotation and Summarization, arXiv:1612.06950.

Question Answering

What kind of store is this?	bakery bakery pastry	art supplies grocery grocery
Is the display case as full as it could be?	no no no	no yes yes

How many bikes are there?	2 2 2	3 4 12
What number is the bus?	48 48 48	4 46 number 6

(from Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, Devi Parikh, VQA: Visual Question Answering, CVPR, 2015 SUNw:Scene Understanding workshop)

- Virginia Tech / MSR [Web] [Paper]
  - Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, Devi Parikh, VQA: Visual Question Answering, CVPR, 2015 SUNw:Scene Understanding workshop.
- MPI / Berkeley [Web] [Paper]
  - Mateusz Malinowski, Marcus Rohrbach, Mario Fritz, Ask Your Neurons: A Neural-based Approach to Answering Questions about Images, arXiv:1505.01121.
- Toronto [Paper] [Dataset]
  - Mengye Ren, Ryan Kiros, Richard Zemel, Image Question Answering: A Visual Semantic Embedding Model and a New Dataset, arXiv:1505.02074 / ICML 2015 deep learning workshop.
- Baidu / UCLA [Paper] [Dataset]
  - Hauyuan Gao, Junhua Mao, Jie Zhou, Zhiheng Huang, Lei Wang, Wei Xu, Are You Talking to a Machine? Dataset and Methods for Multilingual Image Question Answering, arXiv:1505.05612.
- POSTECH [Paper] [Project Page]
  - Hyeonwoo Noh, Paul Hongsuck Seo, and Bohyung Han, Image Question Answering using Convolutional Neural Network with Dynamic Parameter Prediction, arXiv:1511.05765
- CMU / Microsoft Research [Paper]
  - Yang, Z., He, X., Gao, J., Deng, L., & Smola, A. (2015). Stacked Attention Networks for Image Question Answering. arXiv:1511.02274.
- MetaMind [Paper]
  - Xiong, Caiming, Stephen Merity, and Richard Socher. "Dynamic Memory Networks for Visual and Textual Question Answering." arXiv:1603.01417 (2016).
- SNU + NAVER [Paper]
  - Jin-Hwa Kim, Sang-Woo Lee, Dong-Hyun Kwak, Min-Oh Heo, Jeonghee Kim, Jung-Woo Ha, Byoung-Tak Zhang, *Multimodal Residual Learning for Visual QA*, arXiv:1606:01455
- UC Berkeley + Sony [Paper]

    - Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach, *Multimodal Compact Bilinear Pooling for Visual Question Answering and Visual Grounding*, arXiv:1606.01847
- Postech [Paper]
    - Hyeonwoo Noh and Bohyung Han, *Training Recurrent Answering Units with Joint Loss Minimization for VQA*, arXiv:1606.03647
- SNU + NAVER [Paper]
    - Jin-Hwa Kim, Kyoung Woon On, Jeonghee Kim, Jung-Woo Ha, Byoung-Tak Zhang, *Hadamard Product for Low-rank Bilinear Pooling*, arXiv:1610.04325.

Image Generation

- Convolutional / Recurrent Networks
    - Aäron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, Koray Kavukcuoglu. “Conditional Image Generation with PixelCNN Decoders”[Paper][Code]
    - Alexey Dosovitskiy, Jost Tobias Springenberg, Thomas Brox, “Learning to Generate Chairs with Convolutional Neural Networks”, CVPR, 2015. [Paper]
    - Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, Daan Wierstra, “DRAW: A Recurrent Neural Network For Image Generation”, ICML, 2015. [Paper]
- Adversarial Networks
    - Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, Generative Adversarial Networks, NIPS, 2014. [Paper]
    - Emily Denton, Soumith Chintala, Arthur Szlam, Rob Fergus, Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks, NIPS, 2015. [Paper]
    - Lucas Theis, Aäron van den Oord, Matthias Bethge, “A note on the evaluation of generative models”, ICLR 2016. [Paper]
    - Zhenwen Dai, Andreas Damianou, Javier Gonzalez, Neil Lawrence, “Variationally Auto-Encoded Deep Gaussian Processes”, ICLR 2016. [Paper]
    - Elman Mansimov, Emilio Parisotto, Jimmy Ba, Ruslan Salakhutdinov, “Generating Images from Captions with Attention”, ICLR 2016, [Paper]
    - Jost Tobias Springenberg, “Unsupervised and Semi-supervised Learning with Categorical Generative Adversarial Networks”, ICLR 2016, [Paper]
    - Harrison Edwards, Amos Storkey, “Censoring Representations with an Adversary”, ICLR 2016, [Paper]
    - Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae, Shin Ishii, “Distributional Smoothing with Virtual Adversarial Training”, ICLR 2016, [Paper]
    - Jun-Yan Zhu, Philipp Krahenbuhl, Eli Shechtman, and Alexei A. Efros, “Generative Visual Manipulation on the Natural Image Manifold”, ECCV 2016. [Paper] [Code] [Video]

- Mixing Convolutional and Adversarial Networks
  - Alec Radford, Luke Metz, Soumith Chintala, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”, ICLR 2016. [Paper]

Other Topics

- Visual Analogy [Paper]
  - Scott Reed, Yi Zhang, Yuting Zhang, Honglak Lee, Deep Visual Analogy Making, NIPS, 2015
- Surface Normal Estimation [Paper]
  - Xiaolong Wang, David F. Fouhey, Abhinav Gupta, Designing Deep Networks for Surface Normal Estimation, CVPR, 2015.
- Action Detection [Paper]
  - Georgia Gkioxari, Jitendra Malik, Finding Action Tubes, CVPR, 2015.
- Crowd Counting [Paper]
  - Cong Zhang, Hongsheng Li, Xiaogang Wang, Xiaokang Yang, Cross-scene Crowd Counting via Deep Convolutional Neural Networks, CVPR, 2015.
- 3D Shape Retrieval [Paper]
  - Fang Wang, Le Kang, Yi Li, Sketch-based 3D Shape Retrieval using Convolutional Neural Networks, CVPR, 2015.
- Weakly-supervised Classification
  - Samaneh Azadi, Jiashi Feng, Stefanie Jegelka, Trevor Darrell, “Auxiliary Image Regularization for Deep CNNs with Noisy Labels”, ICLR 2016, [Paper]
- Artistic Style [Paper] [Code]
  - Leon A. Gatys, Alexander S. Ecker, Matthias Bethge, A Neural Algorithm of Artistic Style.
- Human Gaze Estimation
  - Xucong Zhang, Yusuke Sugano, Mario Fritz, Andreas Bulling, Appearance-Based Gaze Estimation in the Wild, CVPR, 2015. [Paper] [Website]
- Face Recognition
  - Yaniv Taigman, Ming Yang, Marc’Aurelio Ranzato, Lior Wolf, DeepFace: Closing the Gap to Human-Level Performance in Face Verification, CVPR, 2014. [Paper]
  - Yi Sun, Ding Liang, Xiaogang Wang, Xiaoou Tang, DeepID3: Face Recognition with Very Deep Neural Networks, 2015. [Paper]
  - Florian Schroff, Dmitry Kalenichenko, James Philbin, FaceNet: A Unified Embedding for Face Recognition and Clustering, CVPR, 2015. [Paper]
- Facial Landmark Detection
  - Yue Wu, Tal Hassner, KangGeon Kim, Gerard Medioni, Prem Natarajan, Facial Landmark Detection with Tweaked Convolutional Neural Networks, 2015. [Paper] [Project]

Courses

- Deep Vision

    - [Stanford] CS231n: Convolutional Neural Networks for Visual Recognition
    - [CUHK] ELEG 5040: Advanced Topics in Signal Processing(Introduction to Deep Learning)
- More Deep Learning
    - [Stanford] CS224d: Deep Learning for Natural Language Processing
    - [Oxford] Deep Learning by Prof. Nando de Freitas
    - [NYU] Deep Learning by Prof. Yann LeCun

Books

- Free Online Books
    - Deep Learning by Ian Goodfellow, Yoshua Bengio, and Aaron Courville
    - Neural Networks and Deep Learning by Michael Nielsen
    - Deep Learning Tutorial by LISA lab, University of Montreal

Videos

- Talks
    - Deep Learning, Self-Taught Learning and Unsupervised Feature Learning By Andrew Ng
    - Recent Developments in Deep Learning By Geoff Hinton
    - The Unreasonable Effectiveness of Deep Learning by Yann LeCun
    - Deep Learning of Representations by Yoshua bengio

Software

Framework

- Tensorflow: An open source software library for numerical computation using data flow graph by Google [Web]
- Torch7: Deep learning library in Lua, used by Facebook and Google Deepmind [Web]
    - Torch-based deep learning libraries: [torchnet],
- Caffe: Deep learning framework by the BVLC [Web]
- Theano: Mathematical library in Python, maintained by LISA lab [Web]
    - Theano-based deep learning libraries: [Pylearn2], [Blocks], [Keras], [Lasagne]
- MatConvNet: CNNs for MATLAB [Web]
- MXNet: A flexible and efficient deep learning library for heterogeneous distributed systems with multi-language support [Web]
- Deepgaze: A computer vision library for human-computer interaction based on CNNs [Web]

Applications

- Adversarial Training

    - Code and hyperparameters for the paper “Generative Adversarial Networks” [Web]
- Understanding and Visualizing
    - Source code for “Understanding Deep Image Representations by Inverting Them,” CVPR, 2015. [Web]
- Semantic Segmentation
    - Source code for the paper “Rich feature hierarchies for accurate object detection and semantic segmentation,” CVPR, 2014. [Web]
    - Source code for the paper “Fully Convolutional Networks for Semantic Segmentation,” CVPR, 2015. [Web]
- Super-Resolution
    - Image Super-Resolution for Anime-Style-Art [Web]
- Edge Detection
    - Source code for the paper “DeepContour: A Deep Convolutional Feature Learned by Positive-Sharing Loss for Contour Detection,” CVPR, 2015. [Web]
    - Source code for the paper “Holistically-Nested Edge Detection”, ICCV 2015. [Web]

Tutorials

- [CVPR 2014] Tutorial on Deep Learning in Computer Vision
- [CVPR 2015] Applied Deep Learning for Computer Vision with Torch

Blogs

- Deep down the rabbit hole: CVPR 2015 and beyond@Tombone’s Computer Vision Blog
- CVPR recap and where we’re going@Zoya Bylinskii (MIT PhD Student)’s Blog
- Facebook’s AI Painting@Wired
- Inceptionism: Going Deeper into Neural Networks@Google Research
- Implementing Neural networks