Computer Vision – HW1

Orr Avrech - 302857065

Opher Bar Nathan - 302188628

**Dry Section**

1. 1. Input: image size 128x128x3.

Layer 1: conv, K(# of filters)=64, F(filter dimensions)= 1x1x3,S(stride)=1 output dimension - 128x128x64

Layer 2: pooling 2x2, output dimension - 64x64x64

Layer 3: conv, K=32, F = 5x5x64, P=0, S=1 output dimension - 60x60x32

For Layers 1 and 3, we use the formula: .

We made an assumption that S=1 .

* 1. Example: Input 2x2x3 , kernel 1x1x3

Stride = 1, P(zero padding) = 0 (no zero padding), K (kernels) = 2 (two filters).



The kernel is a sharpen kernel.

* + 1. Stride = 2, Zero Padding = 1 , output dimension:
    2. Stride = 2, Zero Padding = 0, output dimension:

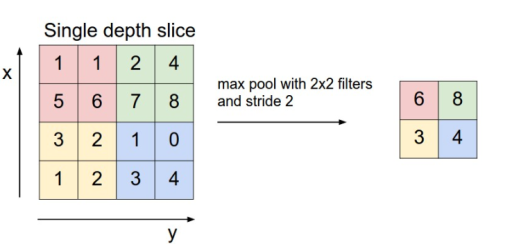
1. 1. Convolution Layer:

Convolution Layer's kernels are learnable, each pixel of input image is a neuron while each pixel of the kernel is a weight. The layer computes the output of neurons that are connected to local regions (which depend on the kernels size - width and height), in the input, each computing a dot product between their weights and a small region they are connected to in the input volume.

Convolution Layer's parameters consist of a set of learnable kernels (filters), in practice the parameters are the network weights and bias.

Convolution Layer's Hyper-parameters control the size of the output volume, they are Depth, Stride, Zero-padding and receptive field (filter size).

* 1. Pooling Layer:

The layer performs a down sampling operation along the spatial dimensions (width, height. For instance, spatial dimensions are 2x2, max pooling will set the maximum value within little 2x2 region in some depth slice for the layer's output.

Average pooling sets the average value within the little region for the layer's output.

Pooling Layer's Hyper-parameters are spatial dimensions and the Stride (in the example above the stride is 2).

* 1. FC Layer: (Fully-Connected)

This layer computes the class scores. Each neuron in this layer is have full connections to all activations in the previous layer.

However, the neurons in the convolution layer are connected only to a local region in the input

**FC layer -> Convolution layer:** F (filter size) = input size, P(zero-padding)=0, S(stride) = 1, K(filter depth) = FC layer #of neurons.

**Convolution layer -> FC layer:** The weight matrix would be a large matrix that is mostly zero except for at certain blocks (due to local connectivity - regions which were the filter spatial dimensions).

1. **Image Classification**: The task in Image Classification is to predict a single label (or a distribution over labels) for a given image, or assigning an input image to one label from a fixed set of categories.

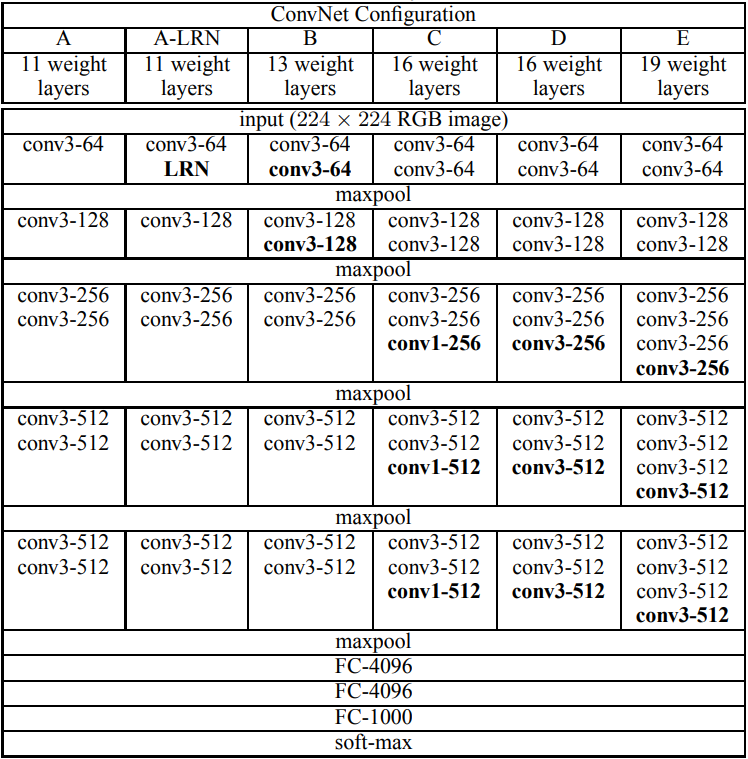
**ImageNet Challenge** full name is ImageNet Large Scale Visual Recognition Challenge (ILSVRC).

It is an annual competition organized by the ImageNet team. During the computation, research teams evaluate their computer vision algorithms in the field of visual recognition tasks (for example Classification, Localization). The training data is a subset of ImageNet with around 1.2 million images belonging to 1000 classes.

1. **VGG16 -** all convolution layers use filters (kernels) with spatial dimensions of 3x3, and stride = 1.

All pooling layers are 2x2 max pooling and stride = 2.

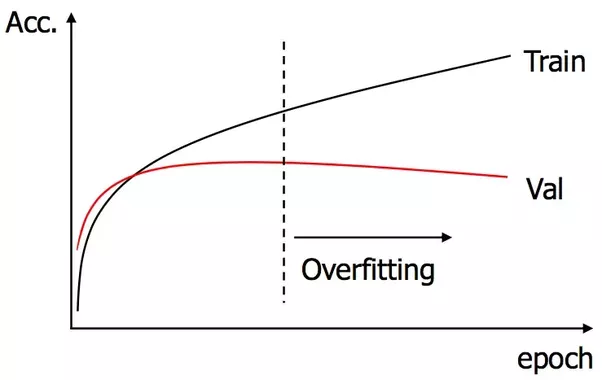
(According to "VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION" by Karen Simonyan & Andrew Zisserman).

****

|  |  |  |  |
| --- | --- | --- | --- |
| Layer | Input | Output | Assumptions |
| Conv. layer 3x3x64 | 224x224x3 | 224x224x64 | Zero padding = 1 |
| Conv. layer 3x3x64 | 224x224x64 | 224x224x64 | Zero padding = 1 |
| 2x2 maxpool | 224x224x64 | 112x112x64 |  |
| Conv. layer 3x3x128 | 112x112x64 | 112x112x128 | Zero padding = 1 |
| Conv. layer 3x3x128 | 112x112x128 | 112x112x128 | Zero padding = 1 |
| Conv. layer 3x3x128 | 112x112x128 | 112x112x128 | Zero padding = 1 |
| 2x2 maxpool | 112x112x128 | 56x56x128 |  |
| Conv. layer 3x3x256 | 56x56x128 | 56x56x256 | Zero padding = 1 |
| Conv. layer 3x3x256 | 56x56x256 | 56x56x256 | Zero padding = 1 |
| Conv. layer 3x3x256 | 56x56x256 | 56x56x256 | Zero padding = 1 |
| 2x2 maxpool | 56x56x256 | 28x28x256 |  |
| Conv. layer 3x3x512 | 28x28x256 | 28x28x512 | Zero padding = 1 |
| Conv. layer 3x3x512 | 28x28x512 | 28x28x512 | Zero padding = 1 |
| Conv. layer 3x3x512 | 28x28x512 | 28x28x512 | Zero padding = 1 |
| 2x2 maxpool | 28x28x512 | 14x14x512 |  |
| Conv. layer 3x3x512 | 14x14x512 | 14x14x512 | Zero padding = 1 |
| Conv. layer 3x3x512 | 14x14x512 | 14x14x512 | Zero padding = 1 |
| Conv. layer 3x3x512 | 14x14x512 | 14x14x512 | Zero padding = 1 |
| 2x2 maxpool | 14x14x512 | 7x7x512 |  |
| FC-4096 | 7x7x512 | 1x1x4096 |  |
| FC-4096 | 1x1x4096 | 1x1x4096 |  |
| FC-1000 | 1x1x4096 | 1x1x1000 |  |
| Soft-max | 1x1x1000 | 1x1x1000 |  |

1. Overfitting is reflected when the error on the training set is driven to a very small value, but when new data is presented to the network, the error is large. The network has memorized the training examples, but it has not learned to generalize to new situations.

One way for overfitting recognition during running is to split the initial train set to subsets, and once in awhile during the training, run one of the subsets as a validation set. Measuring the error/accuracy and compare to the error/accuracy for the other train subsets. If the performance of the network for the validation set is much worth than for the training sets, it can cause by overfitting.



1. The updating for of the network weights (kernels) is executed by backpropagation.

Backpropagation is a way to compute the gradients on the connections of the neural network, with respect to a loss function, while using chain rule.

1. For network training we use data. In general, it is preferred to use big amount of data for better results. The data is split to train and test sets. The train set is the major of this data. The network first run the train set for parameters and weights updating. The test set is used to evaluate the performance of the network over the “real world”.

For example, train set for image classification network can consist hundreds of images for each label, and test set will be one or several images. In addition, the training includes the backpropagation process in contrast to the evaluation. Therefore runtime for train set is much longer than for test set.

**Wet Section**

**Task 1**

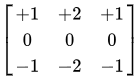
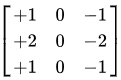
Open CVhw1.ipynb

All explanations are in there.

**Task 2**

A.1.

Sobel Edge Detector - computation of the gradient of image intensity. Using convolution kernels (see images) for each axes (x and y) of the image: and finally combine the two results. For gradient magnitude: For gradient Direction: .

Last step is to choose a Threshold, which each pixel with gradient magnitude above it will be set as an edge pixel.

Laplacian of Gaussian (LoG) Edge Detector - convolution of image with Laplacian (or gradient), may detect noise as an edge. To solve this issue, it possible to pass the image thru Gaussian filter (convolution with Gaussian kernel) and then convolve with Laplacian. Because linearity of convolution operation, there will be same result for convolve the image with Laplacian of Gaussian (LoG). The result of Laplacian of Gaussian has 2 dimension, hence convolution with image will find edges in all directions.

For LoG edge detector there is 2 relevant parameters: The variance of the Guassian kernel (Sigma) and a Threshold which each pixel with value above it will be set as an edge pixel.

Canny Edge Detector - the first step is to convolve (filter) with derivative of Gaussian, which help with the noise and will find the edges similar to the sobel operator.

The next step is using two Thresholds (low and high). Pixels with gradient magnitude above the high threshold will set as edge pixels.

Third step is non-maximum suppression, to find the pixel with maximum value in the gradient direction (Thinning).

Forth step is "walking" form an edge pixel to the gradient vertical direction. Pixels which magnitude above the lower threshold will be set as edge pixels. This step uses the pixels from previous steps to start edge curves, and pixels above low threshold to continue the edge curves.

A.2.

Sobel Edge Detector - Threshold parameter: After finding image gradient by sobel operator, pixels with gradient magnitude above the threshold will be set as edge pixels, and a pixel below threshold will be set as not edge pixel. Increasing the threshold will affect the number the edge pixels. For instance, in noisy image there are many pixels with respectively high gradient magnitude, therefore increasing threshold can help with filter the noisy pixels, but also real edged pixel can be set as "not edge".

Laplacian of Gaussian (LoG) Edge Detector - Threshold parameter: Identical explanation to the Sobel Edge Detector.

Canny Edge Detector - There are two threshold parameters, low and high. After filtering the image with gradient of Gaussian, pixels above the high threshold are set as edge pixels, and in later step, pixels above the low threshold are used to complete edge curves which begin with the above high threshold pixels. Increasing/Decreasing the high threshold will affect the Proportionately on the number of the edge curves. Increasing/Decreasing the low threshold will affect the flattery of the edge curves (noisy pixels damages the flattery) and the connectivity of edge curves - increasing the low threshold will cause short curves and may increase the number of the edge curves (they will not be connected).

Canny and Laplacian of Gaussian (LoG) Edge Detector - Sigma parameter: belongs to the Gaussian kernel (variance), which used to filter noise. Increasing the sigma will filter more noise, but may blur the image edges, means that for noisy images increasing sigma can help, but in case of not so noisy images, high sigma can harm the edge detection by blurring too much the edges.

**Results discussion:** We tuned the parameters for each image, for each edge detector.

We tried to choose one image which "easy" edge detected, one which "medium" and one "hard".

In the case of the "easy" - Pandas image. The three edge detectors provides more or less the same good results for our impression.

In the case of the "medium" - Faces image and "hard" - Man Graffiti. Canny Edge detector provides better results than the two others. The edges curves in Canny are smother and less fragmented.

Sobel and LoG results provide satisfying results for sharp and clearly edges, however for blurred and not sharpen edges the results are not good. As it possible to see in Man Graffiti pants and T-shirt.

In addition, in Sobel and LoG edges images, there are much more dots and line segments which are not connected to a certain edge. It is reflected in Faces image and Man Graffiti.

The images are given below.

Notice: that the images results which provided in matlab were better than the images in the document.

Results:

D:\GoogleDrive\STUDIES\Semester 9\Computer vision\HW\HW1\Q2 A - LoG pandas.emfD:\GoogleDrive\STUDIES\Semester 9\Computer vision\HW\HW1\Q2 A - Canny pandas.emfD:\GoogleDrive\STUDIES\Semester 9\Computer vision\HW\HW1\Q2 A - Sobel  pandas.emfD:\GoogleDrive\STUDIES\Semester 9\Computer vision\HW\HW1\Q2 A - Original pandas.emf

D:\GoogleDrive\STUDIES\Semester 9\Computer vision\HW\HW1\Q2 A - Sobel  faces.emfD:\GoogleDrive\STUDIES\Semester 9\Computer vision\HW\HW1\Q2 A - Original faces.emf

D:\GoogleDrive\STUDIES\Semester 9\Computer vision\HW\HW1\Q2 A - Canny faces.emfD:\GoogleDrive\STUDIES\Semester 9\Computer vision\HW\HW1\Q2 A - LoG faces.emf

D:\GoogleDrive\STUDIES\Semester 9\Computer vision\HW\HW1\Q2 A - Original man graffiti.emf

D:\GoogleDrive\STUDIES\Semester 9\Computer vision\HW\HW1\Q2 A - Sobel  man graffiti.emfD:\GoogleDrive\STUDIES\Semester 9\Computer vision\HW\HW1\Q2 A - LoG man graffiti.emf

D:\GoogleDrive\STUDIES\Semester 9\Computer vision\HW\HW1\Q2 A - Canny man graffiti.emf

B.1.

There is a tradeoff between Precision and Recall. For instance, if many pixels were labeled as an edge, Recall value will be high (almost 1), means that the detector found most of the edged pixels. On the other hand, Precision value will be low, means that the detector labeled many pixels as edged pixels, while they are not a real edge.

Each application demands other values of Precision and Recall. Example, medical applications for defining shapes of cancerous growth demands high Precision value, because the detection has to be accurate without false alarms. While applications, which define if the object which just entered to the room is a human or an animal (dog/cat/wild pig), doesn't need the accurate edges of the object for decision, hence Recall value will be high.

B.2.

D:\GoogleDrive\STUDIES\Semester 9\Computer vision\HW\HW1\Q2 B2.emf