1. Theoretical Tasks

Often enough, Convolutional Neural Networks (CNN) are built by stacking blocks composed of a convolutional layer followed by a nonlinearity and by a max pooling layer. At the end, it is common to have a fully connected layer to classify the extracted features. Finally softmax operation is performed. In the exercise below you have to calculate by hand each one of the steps.

Task 1.1 Convolution

Theoretical Background

Read the following two blog posts:

- What are convolutions?
- Convolutions and Neural Networks

Practice

Now consider the following image I, represented as a matrix I:

$$I = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & -3 & -4 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

And the following kernel k, represented as a matrix k:

$$k = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Calculate a *same* convolution $I * k$ as described in `Convolutions and Neural Networks` above. Use zero padding for handling the margins. Since its a *same* convolution, use $(1, 1)$ stride. DO NOT use a computer to compute this operation, do it by hand - yes, I know, I know... but it takes you 5 minutes and it's good to do it at least *once* in life.

Task 1.2 Non Linearity

Theoretical Background

In general, Convolutional layers are followed by a nonlinearity i.e. a non linear function is applied to their output. Without the nonlinearity all layers could be collapsed to a single matrix multiplication (non desirable). These functions play a major role affecting the learning (or non learning!) of neural networks. Common ones are sigmoid-like (sigmoid, tanh, softsign, . . .) and ReLU-like (ReLU, LeakyReLU, . . .).

Practice

Apply the ReLU activation function to the result of previous task.

Task 1.3 Max Pooling

Theoretical Background

Max Pooling divides the input image in several sections of a given size. This size is often referred to as *subsample size*, or *filter size*. Then, for each section, we only return the biggest value present in that section. For example in Figure 10 here the sections are of size $(2, 2)$. It is easy to notice that the output image will have a smaller size than the input. In this case, because the filter size is $(2, 2)$, the final size of the image half of the original image. If it were $(3, 3)$, then the final size would be one third of the original image.

Practice

Apply *valid* Max Pooling with a filter size of $(2, 2)$ on the result of previous task. Consider a stride of $(2, 2)$. Notice that often enough if the stride is not provided it means it equal to the filter size (bad practice!).

Task 1.4 Flattening

Theoretical Background

Flattening is the process of reshaping a matrix into a one dimensional vector e.g by putting all the rows of the image in one same line. For example, the image below:

$$I = \begin{bmatrix} 0 & 1 \\ 2 & 3 \end{bmatrix}$$

will become:

$$I_{flattened} = [0, 1, 2, 3]^T$$

Practice

Flatten the result of previous task.

Task 1.5 Fully Connected Layer

Theoretical Background

After extracting the features a fully connected layer is used to perform classification. This is consisting in a simple matrix multiplication where there weight matrix of the layer has are as many rows as classes and as many columns as features. For this reason the flattening operation the we just performed is very handy.

Practice

Calculate the output of a Fully Connected layer. Use the following W:

$$W = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{bmatrix}$$

Task 1.6 SoftMax

Theoretical Background

Finally, a softmax operation is performed to transform the raw output of the network into probabilities. Eventually, the highest number after the softmax operation will be selected as output class.

Practice

Apply a softmax to the output of previous task and indicate which is the output class (1st or 2nd).