
Deep Learning — Assignment 3

Anirudhan J. Rajagopalan
Department of Computer Science
New York University
New York, NY.
ajr619@nyu.edu

1 General Questions

1.1 Distilling a network into one module

A linear activation function can be defined as

$$a^i = w^{(i)}x + b^i$$

A linear combination of such layers can be expressed as a sum of all the individual activations. For example, given two activation layers

$$\begin{aligned} y &= w^{(2)}x_{out} + b^{(2)} \\ &= w^{(2)}w^{(1)}x_{in} + w^{(2)}b^{(1)} + b^{(2)} \end{aligned}$$

Generalizing this, we can write

$$\begin{aligned} y &= Wx_{in} + b \\ W &= \prod_{i=1}^l w_i \\ b &= \sum_{i=1}^l \left(\prod_{l=i+1}^l w_l \right) b_i + b_n \end{aligned}$$

1.2 Dictionary used in Sparse coding vs Autoencoders

An autoencoder is a model which tries to reconstruct the input using some sort of constraint. Without any constraints an autoencoder can learn just the identity function which will always provide the least reconstruction loss. Thus a dictionary learnt by an autoencoder need not be sparse. Additionally there are different type of encoding techniques, like denoising autoencoders which can be used to remove noise from an input. Sparse coding is a encoding method which can forces the model to find a sparse and overcomplete representation of the input. Thus a dictionary learnt with denoising autoencoder will always be sparse (there will be lot of features with zeros).

2 Softmax regression gradient calculation

2.1 Derive $\frac{\partial l}{\partial W_{i,j}}$

2.2 Loss functions and gradients

2.2.1 When $y_{c1} = 1, \hat{y}_{c2} = 1, c_1 \neq c_2$

2.2.2 Why there is no need to worry about this situation

3 Chain rule

3.1 Calculating gradients using chain rule

3.2 Gradients at $x = 1, y = 0$

4 Variants of Pooling

Pooling is an operation in Convolutional Neural Networks which is performed after a convolutional layer. We decide on a size of the region to do pool our convolutional filters over. Then we divide the convolved features into disjoint regions and take the max, mean or norm of the feature activations depending on the type of pooling operation. Generally the stride length used in pooling is equal to the pooling window size.

4.1 Three types of pooling

There are three types of pooling operations that are used

Max pooling Max pooling takes the maximum of the feature activations. Implemented by SpatialMaxPooling.

Average pooling Average pooling takes the average of the feature activations. Implemented by SpatialAveragePooling.

Lp pooling LpPooling takes the lp norm of the feature activations. Implemented by SpatialLPPooling.

4.2 Mathematical formulas for pooling

Suppose we have a $N \times N$ convolutional layer (filters) and a pooling layer of size $m \times m$ where $m < N$. The pooling operation is generally implemented using a stride of m . Therefore the size of filters after pooling will be $\frac{N}{m} * \frac{N}{m}$ and each of the $m \times m$ pooling operations gives a single value. The single value for a pooling operation can be represented by

$$y = \sigma(x_{ij})$$

where $1 \leq i \leq m$ and $1 \leq j \leq m$.

4.2.1 MaxPooling

Incase of max pooling

$$\sigma = \max(x_{ij}) \quad 1 \leq i \leq m \text{ and } 1 \leq j \leq m$$

4.2.2 Average Pooling

Incase of average pooling

$$\sigma = \frac{\sum_{i=1}^m \sum_{j=1}^m x_{ij}}{m * m}$$

4.2.3 LPPooling

Incase of LP pooling where $p \geq 2$.

$$\sigma = \left(\sum_{i=1}^m \sum_{j=1}^m x_{ij}^p \right)^{\frac{1}{p}}$$

4.3 Reason for using in deep learning

Pooling is used for a number of reasons.

1. Computational tractability. The number of features after convolution increases by $numFilters \times (N - m + 1) \times (N - m + 1)$. Generally it will be computationally intractable to perform learning on this huge data.
2. Overfitting. Since the number of features are high, this will lead to overfitting. Pooling helps in reducing the number of features.
3. Stationary property. Generally images have stationary property. This means that features that are useful in one region will be useful in other regions also. Pooling operation helps in aggregating the features at various locations.

We generally use Max-pooling for images as the performance of Max pooling has been empirically shown to be better than Average or Lp pooling for image workflows. This can be explained by the stationary property of the image as the most important activation in a small patch can be found by using max pooling.

5 Convolution

5.1 Number of values after forward propagation

When we apply a convolution on a $N \times N$ image with a kernel of size $m \times m$ we get $(N - m + 1) \times (N - m + 1)$ output values after the convolution.

Here, $N = 5$, $m = 3$. Therefore we will get $(5 - 3 + 1) \times (5 - 3 + 1) = 9$ values.

5.2 Values after forward propagation

The resulting filter obtained by applying the convolutional kernel is:

109	92	72
108	85	74
110	74	79

5.3 Value of gradient

4	7	10	6	3
9	17	25	16	8
11	23	34	23	11
7	16	24	17	8
2	6	9	7	3

6 Optimization

6.1 Mathematical formula for reconstruction loss

6.2 Gradient of the loss with respect to the loss

6.3 Gradient descent step

6.4 Gradient step with momentum

7 Top-k error

7.1 Definition

Top-k error is an error metric used in ranking problems. This is the number of samples, x_i with correct labels y_i that doesn't appear in the top-k predicted results of the model in the order of the probability measure or score used for ranking. According to LVSRC 2014 by Imagenet [1], the error is defined as

$$e = \frac{1}{n} \cdot \sum_k \min_i d(c_i, C_k)$$

where C_k is the ground truth of the image with $k = 1, \dots, n$ labels. and c_i is the labels generated by the classification algorithm with $i \in 1, \dots, 5$.

Given this formulation, using the top-5 error helps us not to penalize a classification algorithm if it finds an object in an image which is not present in ground truth.

Using top-1 error ensures how close the object identified in the image by an algorithm is with respect to the original image in imagenet database.

8 t-SNE

8.1 Crowding problem

8.2 Derive $\frac{\partial C}{\partial y_i}$

9 Proximal gradient descent

References

- [1] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.