Deep Learning for Visual Recognition - Assignment 6

Mayara E. Bonani, Guillaume Rouvarel, Arash Safavi, Vardeep Singh, Cüneyt Erem

January 22, 2021

Theoretical Part

a) Backpropagation through convolution and pooling

1)

To calculate derivative of E(o, y) where o = w * x, dimension w X w;

$$\nabla_w E(o,y) = \frac{\partial E}{\partial w} = \frac{\partial E}{\partial y}\frac{\partial y}{\partial w} \tag{1}$$

$$= \frac{-2(o-y)}{n^2} * rotate(x) \tag{2}$$

where (n-m+1) X (n-m+1) for s=1;

$$\frac{\partial E}{\partial y} = \frac{-2(o-y)}{n^2} \tag{3}$$

$$\nabla_b E(o,y) = \frac{\partial E}{\partial b} = \frac{\partial E}{\partial y}\frac{\partial y}{\partial b} \tag{4}$$

$$= \frac{-2(o-y)}{n^2} \tag{5}$$

where activation function is considered as giving only convolution operator directly without using sigmoid or any other functions.

2) To calculate partial derivative for 2-max-pooling, we need to calculate maximum values of 2 values;

$$E(o',y') = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(o'-y)^2 \tag{6}$$

dimension w X w;

$$\nabla_w E(o',y) = \frac{\partial E}{\partial w} = \frac{\partial E}{\partial y}\frac{\partial y}{\partial w} \tag{7}$$

$$= \frac{-2(o' - y)}{n^2} * rotate(maximum(x_1, x_2, x_3, x_4)) \tag{8}$$

where

$$o' = maximum(x_{ij}(1), x_{ij}(2), x_{ij}(3), x_{ij}(4)) \tag{9}$$

$$\nabla_b E(o', y) = \frac{\partial E}{\partial b} = \frac{\partial E}{\partial y}\frac{\partial y}{\partial b} \tag{10}$$

$$= \frac{-2(o' - y)}{n^2} \tag{11}$$

3) stride is gamma

$$\nabla_w E(o, y) = \frac{\partial E}{\partial w} = \frac{\partial E}{\partial y}\frac{\partial y}{\partial w} \tag{12}$$

$$= \frac{-2(o - y)}{n^2} * rotate(x_{gamma}) \tag{13}$$

where dimention is w ij x w ij and i and j increases by X gamma times faster but dimension for partial b is ((n-m)/s +1) X ((n-m)/s + 1) for s = gamma

$$\nabla_b E(o, y) = \frac{\partial E}{\partial b} = \frac{\partial E}{\partial y}\frac{\partial y}{\partial b} \tag{14}$$

$$= \frac{-2(o - y)}{n^2} \tag{15}$$

and stride gamma X faster for each value

b) Receptive field and output sizes

Each layer l's spatial configuration is parameterized by 4 variables:
k_l: kernel size (positive integer)
s_l : stride (positive integer)
p_i : padding applied to the left side of the input feature map (non-negative integer) 1
For two sequential convolutional layers f_2 and f_1 with kernel size k, stride s, receptive field :

$$r_1 = s_2 \times r_2 + (k_2 - s_2) \tag{16}$$

Or in a more general form:

$$r_{(i-1)} = s_i \times r_i + (k_i - s_i) \tag{17}$$

This equation can be used in a recursive algorithm to compute the receptive field size of the network, r_0. The solution in terms of ks and ss is given by the equation below.

$$r_0 = \sum_{l=1}^{L} \left((k_l - 1) \prod_{i=1}^{l-1} s_i \right) + 1 \tag{18}$$

Reference:

https://shawnleezx.github.io/blog/2017/02/11/calculating-receptive-field-of-cnn/ - Acessed on 22.01.2020

https://distill.pub/2019/computing-receptive-fields/- Acessed on 22.01.2020

https://theaisummer.com/receptive-field/ - Acessed on 22.01.2020

https://stanford.edu/ shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks - Acessed on 22.01.2020

2. pooling on the other hand, increases the receptive field size multiplicatively.