24-789: Deep Learning for Engineers Assignment 1

Angelos Mavrogiannis

February 2020

1 Gradient Descent

a)

Based on our objective function:

$$f(x) = GELU(x) = x\sigma(1.702x) \tag{1}$$

we calculate its derivative using the product derivative rule:

$$\frac{df(x)}{dx} = \frac{dGELU}{dx} = \sigma(1.702x) + 1.702x\sigma(1.702x)[1 - \sigma(1.702x)] \tag{2}$$

Calculating x_i for $i = 1, 2, 3$:

$$\begin{aligned} x_1 &= x_0 - \eta\nabla f_{x_0} = -0.05 \\ x_2 &= x_1 - \eta\nabla f_{x_1} = -0.0958 \\ x_3 &= x_2 - \eta\nabla f_{x_2} = -0.1376 \end{aligned} \tag{3}$$

Calculating $GELU(x_i)$ for $i = 1, 2, 3$ based on our previous calculations of x_i:

$$\begin{aligned} GELU(x_1) &= GELU(-0.05) = -0.0239 \\ GELU(x_2) &= GELU(-0.0958) = -0.0440 \\ GELU(x_3) &= GELU(-0.1376) = -0.0608 \end{aligned} \tag{4}$$

b)

For a learning rate $\eta = 1$ we get:

$$\begin{aligned} x_1 &= -0.5 \\ x_2 &= -0.6208 \\ x_3 &= -0.6765 \end{aligned} \tag{5}$$

and the corresponding $GELU$ values:

$$\begin{aligned} GELU(x_1) &= GELU(-0.5) = -0.1496 \\ GELU(x_2) &= GELU(-0.6208) = -0.1601 \\ GELU(x_3) &= GELU(-0.6765) = -0.1625 \end{aligned} \tag{6}$$

c)

For a learning rate $\eta = 0.1$ we get:

$$\begin{aligned} x_1 &= -2.9975 \\ x_2 &= -2.9951 \\ x_3 &= -2.9926 \end{aligned} \tag{7}$$

and the corresponding $GELU$ values:

$$\begin{aligned} GELU(x_1) &= GELU(-2.9975) = -0.0181 \\ GELU(x_2) &= GELU(-2.9951) = -0.0182 \\ GELU(x_3) &= GELU(-2.9926) = -0.0183 \end{aligned} \tag{8}$$

Using Momentum, we first initialize the velocity:

$$\upsilon_0 = \nabla f_{x_0} = -0.0245 \tag{9}$$

and then perform the following calculations:

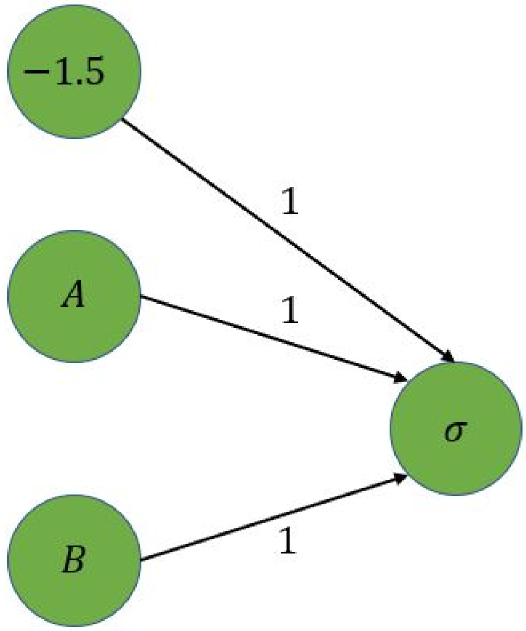
$$\begin{aligned} x_1 &= x_0 - \eta\upsilon_0 = -2.9975 \\ \upsilon_1 &= \beta\upsilon_0 + (1-\beta)\nabla f_{x_0} = -0.0245 \\ x_2 &= x_1 - \eta\upsilon_1 = -2.9951 \\ \upsilon_2 &= \beta\upsilon_1 + (1-\beta)\nabla f_{x_1} = -0.0246 \\ x_3 &= x_2 - \eta\upsilon_2 = -2.9926 \end{aligned} \tag{10}$$

We observe that we get the exact same values with our previous calculations, when we did not use Momentum.
The corresponding $GELU$ values are again:

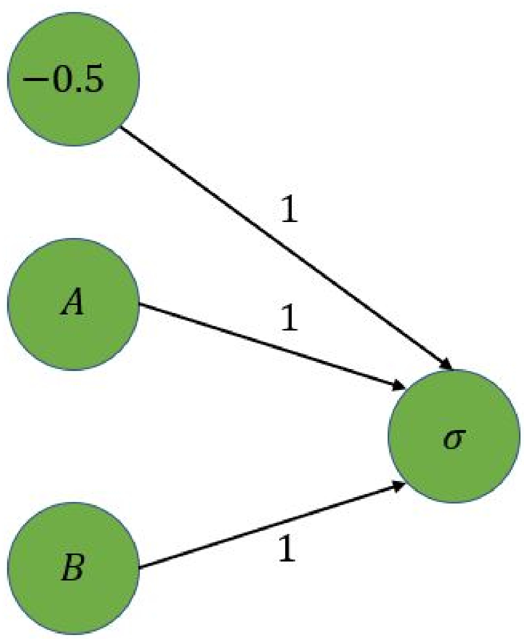
$$\begin{aligned} GELU(x_1) &= GELU(-2.9975) = -0.0181 \\ GELU(x_2) &= GELU(-2.9951) = -0.0182 \\ GELU(x_3) &= GELU(-2.9926) = -0.0183 \end{aligned} \tag{11}$$

2 Neural Network Concepts

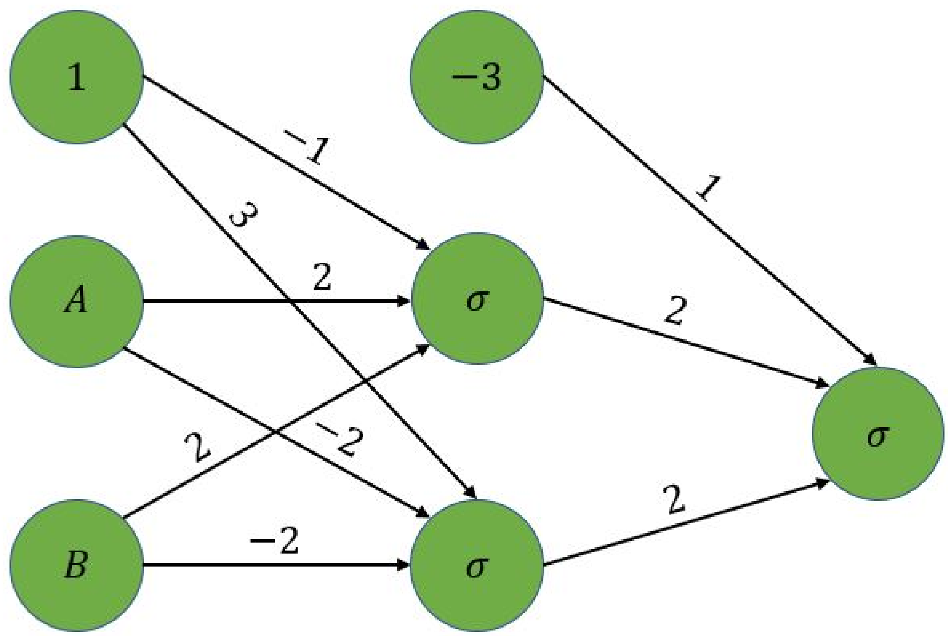
AND



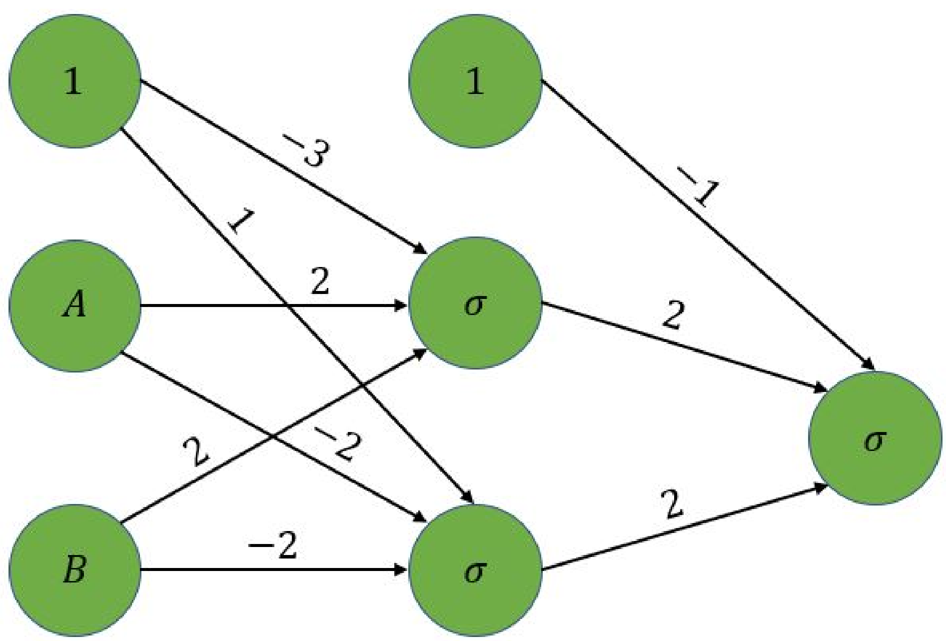
OR



XOR



XNOR



3 Backpropagation

Forward Pass:

$$\begin{cases} f_1 = xW_1 + b_1 \\ \alpha = \sigma(f_1) \\ f_2 = \alpha W_2 + b2 \\ o = S(f_2) \end{cases} \tag{12}$$

where S is the $softmax$ function:

$$S(x_k) = p_k = \frac{e^{x_k}}{\sum_j e^{x_j}} \tag{13}$$

and our loss function is cross-entropy:

$$E(o) = -\sum_1^K y_i \log o_i \tag{14}$$

a)

Derivative of $softmax$ when $k = i$:

$$\frac{\partial p_{k=i}}{\partial x_i} = \frac{e^{x_{k=i}} \sum e^{x_j} - e^{x_{k=i}} e^{x_i}}{(\sum e^{x_j})^2} = p_{k=i}(1 - p_i) \tag{15}$$

Derivative of $softmax$ when $k \neq i$:

$$\frac{\partial p_{k\neq i}}{\partial x_i} = \frac{0 \sum e^{x_j} - e^{x_{k\neq i}} e^{x_i}}{(\sum e^{x_j})^2} = -p_{k\neq i} p_i \tag{16}$$

Overall:

$$\frac{\partial p_k}{\partial x_i} = p_k(1\{k = i\} - p_i) \tag{17}$$

where $1k = i$ is equal to 1 if $k = i$ and 0 otherwise.

Derivative of cross-entropy taking into account the derivative of $softmax$ we just calculated:

$$\begin{aligned} \frac{\partial E}{\partial x_i} = -\sum_{k=1}^K y_k \frac{\partial \log p_k}{\partial p_k} \frac{\partial p_k}{\partial x_i} = -\sum_{k=1}^K \frac{y_k}{p_k} p_k(1\{k = i\} - p_i) = \\ -\sum_{k=1}^K y_k(1\{k = i\} - p_i) = -\sum_{k=1}^K y_k p_i - -\sum_{k=1}^K y_k 1\{k = i\} = p_i - y_i \end{aligned} \tag{18}$$

Therefore, in our case:

$$\frac{\partial E}{\partial f_{2i}} = \frac{\partial E}{\partial o_i} \frac{\partial o_i}{\partial f_{2i}} = o_i - y_i \tag{19}$$

which means that the derivative is equal to the output of the $softmax$ function minus the labels in one-hot vector encoding notation.

b)

$$\frac{\partial E}{\partial x} = \frac{\partial E}{\partial o} \frac{\partial o}{\partial f_2} \frac{\partial f_2}{\partial \alpha} \frac{\partial \alpha}{\partial f_1} \frac{\partial f_1}{\partial x} \tag{20}$$

where:

$$
\begin{cases}
\frac{\partial E}{\partial o_i}\frac{\partial o_i}{\partial f_{2i}} = o_i - y_i \\
\frac{\partial f_2}{\partial \alpha} = W_2 \\
\frac{\partial \alpha}{\partial f_1} = \frac{d\sigma(f_1)}{df_1} = \sigma(f_1)\big[1-\sigma(f_1)\big] \\
\frac{\partial f_1}{\partial x} = W_1
\end{cases}
\tag{21}
$$

Hence, overall we get:

$$
\frac{\partial E}{\partial x} = (o-y)W_2\sigma(f_1)\big[1-\sigma(f_1)\big]W_1 \tag{22}
$$

4 Convolutional Neural Network

a)

With stride $s = 1$, we need to use a padding size of $p = \frac{f-s}{2} = \frac{3-1}{2} = 1$, where f is the size of the filter. Using size 1 padding, the feature map F becomes:

$$
F = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 3 & 5 & 2 & 3 & 0 \\
0 & 9 & 1 & 8 & 4 & 0 \\
0 & 6 & 4 & 3 & 7 & 0 \\
0 & 7 & 0 & 2 & 4 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix} \tag{23}
$$

Doing the calculations in the following manner:

$$
\begin{aligned}
F'(1,1) &= 0*(-1)+0*0.5+0*(-2)+0*2+3*0+5*1+0*0+9*1+1*1.5 = 15.5 \\
F'(1,2) &= 0*(-1)+0*0.5+0*(-2)+3*2+5*0+2*1+9*0+1*1+8*1.5 = 21 \\
F'(1,3) &= 0*(-1)+0*0.5+0*(-2)+5*2+2*0+3*1+1*0+8*1+4*1.5 = 27 \\
F'(1,4) &= 0*(-1)+0*0.5+0*(-2)+2*2+3*0+0*1+8*0+4*1+0*1.5 = 8 \\
F'(2,1) &= 0*(-1)+3*0.5+5*(-2)+0*2+9*0+1*1+0*0+6*1+4*1.5 = 4.5 \\
F'(2,2) &= 3*(-1)+5*0.5+2*(-2)+9*2+1*0+8*1+6*0+4*1+3*1.5 = 30 \\
F'(2,3) &= 5*(-1)+2*0.5+3*(-2)+1*2+8*0+4*1+4*0+3*1+7*1.5 = 9.5 \\
F'(2,4) &= 2*(-1)+3*0.5+0*(-2)+8*2+4*0+0*1+3*0+7*1+0*1.5 = 22.5 \\
F'(3,1) &= 0*(-1)+9*0.5+1*(-2)+0*2+6*0+4*1+0*0+7*1+0*1.5 = 13.5 \\
F'(3,2) &= 9*(-1)+1*0.5+8*(-2)+6*2+4*0+3*1+7*0+0*1+2*1.5 = -6.5 \\
F'(3,3) &= 1*(-1)+8*0.5+4*(-2)+4*2+3*0+7*1+0*0+2*1+4*1.5 = 18 \\
F'(3,4) &= 8*(-1)+4*0.5+0*(-2)+3*2+7*0+0*1+2*0+4*1+0*1.5 = 4 \\
F'(4,1) &= 0*(-1)+6*0.5+4*(-2)+0*2+7*0+0*1+0*0+0*1+0*1.5 = -5 \\
F'(4,2) &= 6*(-1)+4*0.5+3*(-2)+7*2+0*0+2*1+0*0+0*1+0*1.5 = 6 \\
F'(4,3) &= 4*(-1)+3*0.5+7*(-2)+0*2+2*0+4*1+0*0+0*1+0*1.5 = -12.5 \\
F'(4,4) &= 3*(-1)+7*0.5+0*(-2)+2*2+4*0+0*1+0*0+0*1+0*1.5 = 4.5
\end{aligned} \tag{24}
$$

we get the new feature map:

$$
F' = \begin{pmatrix}
15.5 & 21 & 27 & 8 \\
4.5 & 30 & 9.5 & 22.5 \\
13.5 & -6.5 & 18 & 4 \\
-5 & 6 & -12.5 & 4.5
\end{pmatrix} \tag{25}
$$

which is $4x4$, maintaining the size of the original feature map F.

b)

Using Filter2 with padding size $p = 1$ we observe that we cannot get a new feature map with the same size as the original (4x4). That is because Filter2 has dimensions 2x2, which means that with stride $s = 1$ it will output a 5x5 feature map, whereas with stride $s = 2$, it will give us a 3x3 feature map. Hence the required output size cannot be achieved in this case.
This can also be proved by the following formula for calculating the spatial size S of the output volume:

$$S = k\frac{w - f + 2p}{s} + 1 \rightarrow 4 = 1 \cdot \frac{4 - 2 + 2 \cdot 1}{s} + 1 \rightarrow s = \frac{4}{3} \tag{26}$$

It is not possible for the stride to be equal to a fractional number.

c

Performing average pooling on the feature map F' we derived on question a), we form 2x2 patches and take the average of their values:

$$\begin{aligned}
F''(1,1) &= \frac{15.5 + 21 + 4.5 + 30}{4} = 17.75 \\
F''(1,2) &= \frac{27 + 8 + 9.5 + 22.5}{4} = 16.75 \\
F''(2,1) &= \frac{13.5 - 6.5 - 5 + 6}{4} = 2 \\
F''(2,2) &= \frac{18 + 4 - 12.5 + 4.5}{4} = 3.5
\end{aligned} \tag{27}$$

We eventually get the following feature map:

$$F'' = \begin{pmatrix} 17.75 & 16.75 \\ 2 & 3.5 \end{pmatrix} \tag{28}$$

d)

The required filter needs to transform the 4x4 feature map F' to a 2x2 one, achieving the same result as the average pooling operation on question c). Hence, we need a 2x2 filter with a padding size of $p = 0$ and stride $s = 2$.
Solving the following system of equations:

$$\begin{aligned}
&15.5 \cdot a + 21 \cdot b + 4.5 \cdot c + 30 \cdot d = 17.75 \\
&27 \cdot a + 8 \cdot b + 9.5 \cdot c + 22.5 \cdot d = 16.75 \\
&13.5 \cdot a - 6.5 \cdot b - 5 \cdot c + 6 \cdot d = 2 \\
&18 \cdot a + 4 \cdot b - 12.5 \cdot c + 4.5 \cdot d = 3.5
\end{aligned} \tag{29}$$

we get the desired filter:

$$\begin{pmatrix} 0.25 & 0.25 \\ 0.25 & 0.25 \end{pmatrix} \tag{30}$$