Assignment 3 - Cs6890 - 2024

MohammadReza EskandariNasab
Department of Computer Science
Utah State University
reza.eskandarinasab@usu.edu

1 Q1

Given the time series of stock prices for Google (Alphabet Inc.) and considering $\beta = 0.9$, we calculate the exponentially weighted averages for each month as follows, without bias correction:

$\theta_1 = 1434.87$, $V_0 = 0$
$V_1 = 0.9V_0 + 0.1\theta_1 = 143.487$
$\theta_2 = 1442$, $V_2 = 0.9V_1 + 0.1\theta_2 = 0.9 \times 143.487 + 144.2 = 273.1383$
$\theta_3 = 1482.76$, $V_3 = 0.9V_2 + 0.1\theta_3 = 0.9 \times 273.1383 + 148.276 = 393.80147$
$\theta_4 = 1655.08$, $V_4 = 0.9V_3 + 0.1\theta_4 = 0.9 \times 393.80147 + 165.508 = 518.929322$
$\theta_5 = 1487.9$, $V_5 = 0.9V_4 + 0.1\theta_5 = 0.9 \times 518.929322 + 148.79 = 615.94559$
$\theta_6 = 1624.32$, $V_6 = 0.9V_5 + 0.1\theta_6 = 0.9 \times 615.94559 + 162.432 = 711.640831$

Therefore, the exponentially weighted averages without bias correction for the stock prices from June to November are V_1, V_2, V_3, V_4, V_5, and V_6 respectively.

2 Q2

Adam optimization algorithm includes several key hyperparameters:

- **Learning Rate** (α or *lr*): The step size for updating the parameters. A typical initial value is 0.001.
- **Beta1** (β_1): Controls the exponential decay rate for the first moment estimates, commonly set to 0.9.
- **Beta2** (β_2): Controls the exponential decay rate for the second moment estimates, commonly set to 0.999.
- **Epsilon** (ϵ): A small number to prevent division by zero in the implementation, often in the range of $1e-7$ to $1e-8$.

Initialization

For the initialization of the Adam optimizer's hyperparameters, we have:

- The learning rate, α, can be one of the set values $\{0.0001, 0.001, 0.01, 0.1, 1\}$ or it can be set as a random value in the logarithmic scale range of $[-4, 0]$. If using a random value, α is set as:
$$\alpha = 10^r$$
where r is a random value between -4 and 0.

- The momentum hyperparameter, β, typically falls within the range of $\{0.9, \ldots, 0.999\}$. Similarly, β can be initialized by setting a random value in the range $[-3, -1]$ for $1 - \beta$ as follows:

$$1 - \beta = 10^r$$
$$\beta = 1 - 10^r$$

where r is a random value between -3 and -1.

3 Q3

Given the initial learning rate $\alpha_0 = 0.5$ and the learning rate decay strategy $\alpha = 0.95^{\text{epochNumber}} \cdot \alpha_0$, the learning rate α for the first ten epochs is calculated as follows:

$$\begin{aligned}
\alpha_1 &= 0.95^1 \cdot 0.5 = 0.475 \\
\alpha_2 &= 0.95^2 \cdot 0.5 \approx 0.45125 \\
\alpha_3 &= 0.95^3 \cdot 0.5 \approx 0.4286875 \\
\alpha_4 &= 0.95^4 \cdot 0.5 \approx 0.407253125 \\
\alpha_5 &= 0.95^5 \cdot 0.5 \approx 0.38689046875 \\
\alpha_6 &= 0.95^6 \cdot 0.5 \approx 0.3675459453125 \\
\alpha_7 &= 0.95^7 \cdot 0.5 \approx 0.349168648046875 \\
\alpha_8 &= 0.95^8 \cdot 0.5 \approx 0.33171021564453125 \\
\alpha_9 &= 0.95^9 \cdot 0.5 \approx 0.3151247048623047 \\
\alpha_{10} &= 0.95^{10} \cdot 0.5 \approx 0.29936846961918944
\end{aligned}$$

To determine the number of epochs needed for the learning rate α to fall below 0.3, we start with the initial learning rate $\alpha_0 = 0.5$ and apply the decay strategy $\alpha = 0.95^{\text{epochNumber}} \times \alpha_0$ at each epoch. Upon calculation, it is observed that after 10 epochs, the learning rate α decreases to approximately 0.2994, which is just below the threshold of 0.3. Therefore, the learning rate α reaches below 0.3 after the completion of 10 epochs.

4 Q4

The softmax function is calculated as follows:

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}$$

For the given vector $z = [3, 2, 5, -4, 0, 0.5, 5.9, 11, -8, -1.2]$, we first compute e^{z_i} for each component:

$$e^z \approx [20.08554, 7.38906, 148.41316, 0.01832, 1, 1.64872, 363.74846, 59874.14172, 0.00034, 0.30119]$$

Then, sum up all the exponentiated values to get $\sum_{j=1}^{n} e^{z_j} \approx 60413.42751$. Now we calculate the softmax:

$$\begin{aligned}
\text{softmax}(z) \approx \Bigg[&\frac{20.08554}{60413.42751}, \frac{7.38906}{60413.42751}, \frac{148.41316}{60413.42751}, \frac{0.01832}{60413.42751}, \\
&\frac{1}{60413.42751}, \frac{1.64872}{60413.42751}, \frac{363.74846}{60413.42751}, \frac{59874.14172}{60413.42751}, \frac{0.00034}{60413.42751}, \frac{0.30119}{60413.42751} \Bigg]
\end{aligned}$$

Rounded to five decimal places:

$$\text{softmax}(z) \approx [0.00033, 0.00012, 0.00246, 0.00000, 0.00002, 0.00003, 0.00602, 0.99100, 0.00000, 0.00001]$$

For the hardmax activation, which sets the maximum value to 1 and all others to 0, we get:

$$\text{hardmax}(z) = [0, 0, 0, 0, 0, 0, 0, 1, 0, 0]$$

5 Q5

Suppose we have an input image of size $128 \times 128 \times 3$. We perform a convolution operation with 5 filters of size $3 \times 3 \times 3$, padding $p = 1$, and stride $s = 2$.

a. Size of Output Tensor

The size of the output tensor can be calculated using the following formula:

$$\text{output height} = \left\lfloor \frac{\text{input height} - \text{filter height} + 2 \times p}{s} \right\rfloor + 1$$

$$\text{output width} = \left\lfloor \frac{\text{input width} - \text{filter width} + 2 \times p}{s} \right\rfloor + 1$$

$$\text{output depth} = \text{number of filters}$$

Applying the formula:

$$\text{output height} = \left\lfloor \frac{128 - 3 + 2 \times 1}{2} \right\rfloor + 1 = 64$$

$$\text{output width} = \left\lfloor \frac{128 - 3 + 2 \times 1}{2} \right\rfloor + 1 = 64$$

The output depth equals the number of filters, which is 5. Therefore, the size of the output tensor after the convolution operation is $64 \times 64 \times 5$.

b. Number of Learnable Parameters

The number of learnable parameters in a convolution layer is given by:

$$(\text{filter height} \times \text{filter width} \times \text{input depth} + 1) \times \text{number of filters}$$

For our filters:

$$(3 \times 3 \times 3 + 1) \times 5 = (27 + 1) \times 5 = 28 \times 5 = 140$$

Thus, there are 140 learnable parameters in the convolutional layer.

c. Learnable Parameters After Max Pooling

Max pooling layers do not add any learnable parameters. Thus, after adding a max pooling layer with a filter size of 2×2 and a stride of $s = 2$, the number of learnable parameters remains 140.