CmpE597 Homework 1 Report (Spring 2022)

Emre Girgin - 2021700060

April 12, 2022

1 Part1

1.1 Question 1

The MNIST dataset is downloaded from the repositories of PyTorch framework. The dataset and dataloader classes used, as well. Therefore, to run and to download the dataset, **an internet connection is needed**.

1.2 Question 2

In this section the desired architecture is implemented. It is observed that convolutional layers are the bottleneck, so there are two implementations are present. The first implementation is exactly what it is wanted. The second implementation is the lightweight version of it, where only fully connected layers are utilized. Both implementations work and the user can select which one to run. (Check ReadME) Remember that lightweight version only needs **5** mins to run but the full implementations requires **3 hours**. Check *alternativelib* folder for details.

The hyperparameters used in lightweight version are shown in the Table 1. These parameters achieve **0.97 test set accuracy**. The hyperparameters used in full implementation version are shown in the Table 2. These parameters achieve **0.91 test set accuracy**. Be aware that this experiment is stopped early where only **1500 iterations** are done, which takes half an hour approximately.

1.3 Question 3

Forward Pass:

Convolutional Layer 0:

$$h_0 = X * W_0 + b_0 \tag{1}$$

, where $X \in \Re^{28x28x1}, W_0 \in \Re^{5x5x4}, b_0 \in \Re 24x24x4$

ReLU Layer 0:

$$z_0 = max(0, h_0) \tag{2}$$

, where $h_0 \in \Re^{24x24x4}$

Max Pool Layer 0:

$$a_0 = MaxPool(z_0) \tag{3}$$

, where $z_0 \in \Re^{24x24x4}$

Convolutional Layer 1:

$$h_1 = a_0 * W_1 + b_1 \tag{4}$$

, where $a_0 \in \Re^{12x12x4}, W_1 \in \Re^{5x5x8}, b_1 \in \Re^{8x8x8}$

ReLU Layer 1:

$$z_1 = max(0, h_1) \tag{5}$$

Epochs	Batch Size	Learning Rate	Momentum Coeff.
2	8	0.01	0.9

Table 1: Training hyperparameters for lightweight implementation.

Iteration	Batch Size	Learning Rate	Momentum Coeff.
1000	8	0.01	0.9

Table 2: Training hyperparameters for full implementation.

, where $h_1 \in \Re^{8x8x8}$

Max Pool Layer 1:

$$a_1 = MaxPool(z_1) \tag{6}$$

, where $z_1 \in \Re^{8x8x8}$

Flatten Layer:

$$h_2 = flatten(a_1) \tag{7}$$

, where $a_1 \in \Re^{4x4x8}$

Fully Connected Layer 0:

$$z_2 = W_2 h_2 + b_2 \tag{8}$$

, where $h_2 \in \Re^{128x1}, W_2 \in \Re^{128x128}, b_2 \in \Re^{128x1}$

ReLU Layer 2:

$$a_2 = max(0, z_2) \tag{9}$$

, where $z_2 \in \Re^{128x1}$

Fully Connected Layer 1:

$$z_3 = W_3 a_2 + b_3 \tag{10}$$

, where $a_2 \in \Re^{128x1}, W_3 \in \Re^{10x128}, b_2 \in \Re^{10x1}$

Linear Activation Layer 0:

$$a_3 = z_3 \tag{11}$$

, where $z_3 \in \Re^{10x1}$

Softmax Cross Entropy Loss:

$$SCE_L = \sum_{i=1}^{c} -y_i log(softmax(a_3)_i) \tag{12}$$

, where c is the number of classes, y_i is the *ith* element of target vector and o_i is the *ith* element of softmax output.

Backward Pass w.r.t. W_3**:**

$$\frac{\partial SCE_L}{\partial W_3} = \frac{\partial SCE_L}{\partial a_3} \frac{\partial a_3}{\partial z_3} \frac{\partial z_3}{\partial W_3} \tag{13}$$

$$\delta_L = \frac{\partial SCE_L}{\partial a_3} \odot \frac{\partial a_3}{\partial z_3} = a_3 - y \tag{14}$$

, where $\delta_L, a_3, y \in \Re^{10x1}$

$$\frac{\partial SCE_L}{\partial W_3} = \delta_L a_2^T \tag{15}$$

, where $a_2 \in \Re^{128x1}$

$$\frac{\partial SCE_L}{\partial b_3} = \delta_L \tag{16}$$

Backward Pass w.r.t. W_2**:**

$$\frac{\partial SCE_L}{\partial W_2} = \frac{\partial SCE_L}{\partial a_3} \frac{\partial a_3}{\partial z_3} \frac{\partial z_3}{\partial a_2} \frac{\partial a_2}{\partial z_2} \frac{\partial z_2}{\partial W_2} \tag{17}$$

$$\delta_2 = \frac{\partial SCE_L}{\partial a_3} \frac{\partial a_3}{\partial z_3} \frac{\partial z_3}{\partial a_2} \frac{\partial a_2}{\partial z_2} \tag{18}$$

$$\delta_2 = \delta_L \frac{\partial z_3}{\partial a_2} \frac{\partial a_2}{\partial z_2} = (W_3^T \delta_L) \odot \frac{\partial a_2}{\partial z_2} \tag{19}$$

, where $\delta_L \in \Re^{10x1}, W_3 \in \Re^{10x128}, \delta_2 \in \Re^{128x1}$ and $\frac{\partial a_2}{\partial z_2}$ is the derivative of ReLU function which returns 1 if the element is greater than zero, otherwise returns zero.

$$\frac{\partial SCE_L}{\partial W_2} = \delta_2 h_2^T \tag{20}$$

$$\frac{\partial SCE_L}{\partial b_2} = \delta_2 \tag{21}$$

Backward Pass w.r.t. W_1:

$$\frac{\partial SCE_L}{\partial W_1} = \frac{\partial SCE_L}{\partial a_3} \frac{\partial a_3}{\partial z_3} \frac{\partial z_3}{\partial a_2} \frac{\partial a_2}{\partial z_2} \frac{\partial z_2}{\partial h_2} \frac{\partial h_2}{\partial a_1} \frac{\partial a_1}{\partial z_1} \frac{\partial z_1}{\partial h_1} \frac{\partial h_1}{\partial W_1} \tag{22}$$

$$\delta_t = \frac{\partial SCE_L}{\partial a_3} \frac{\partial a_3}{\partial z_3} \frac{\partial z_3}{\partial a_2} \frac{\partial a_2}{\partial z_2} \frac{\partial z_2}{\partial h_2} = (W_2 \delta_2) \tag{23}$$

, where $\delta_t, \delta_2 \in Re^{128x1}, W_2 \in \Re^{128x128}$,

$$\delta_{Flatten} = \delta_t \frac{\partial h_2}{\partial a_1} \tag{24}$$

, where $\delta_{Flatten} \in Re^{4x4x8}$

$$\delta_{MP_1} = \delta_{Flatten} \frac{\partial a_1}{\partial z_1} \tag{25}$$

, where $\delta^{MP_1} \in \Re^{8x8x8}$ and $\frac{\partial a_1}{\partial z_1}$ is the derivative of MaxPooling function and upsamples the data.

$$\delta_1 = \delta_{MP_1} \odot \frac{\partial z_1}{\partial h_1} \tag{26}$$

, where $\delta_1 \in \Re^{8x8x8}$ and $\frac{\partial z_1}{\partial h_1}$ is the derivative of ReLU function which returns 1 if the element is greater than zero, otherwise returns zero.

$$\frac{\partial SCE_L}{\partial W_1} = a_0 * \delta_1 \tag{27}$$

, where $a_0 \in \Re^{12x12x4}$

$$\frac{\partial SCE_L}{\partial b_1} = \delta^1 \tag{28}$$

Backward Pass w.r.t. W_0:

$$\frac{\partial SCE_L}{\partial W_0} = \frac{\partial SCE_L}{\partial a_3} \frac{\partial a_3}{\partial z_3} \frac{\partial z_3}{\partial a_2} \frac{\partial a_2}{\partial z_2} \frac{\partial z_2}{\partial h_2} \frac{\partial h_2}{\partial a_1} \frac{\partial a_1}{\partial z_1} \frac{\partial z_1}{\partial h_1} \frac{\partial h_1}{\partial a_0} \frac{\partial a_0}{\partial z_0} \frac{\partial z_0}{\partial h_0} \frac{\partial h_0}{\partial W_0} \tag{29}$$

$$\delta_0 = \frac{\partial SCE_L}{\partial a_3} \frac{\partial a_3}{\partial z_3} \frac{\partial z_3}{\partial a_2} \frac{\partial a_2}{\partial z_2} \frac{\partial z_2}{\partial h_2} \frac{\partial h_2}{\partial a_1} \frac{\partial a_1}{\partial z_1} \frac{\partial z_1}{\partial h_1} \frac{\partial h_1}{\partial a_0} \frac{\partial a_0}{\partial z_0} \frac{\partial z_0}{\partial h_0} = \delta_1 \frac{\partial h_1}{\partial a_0} \frac{\partial a_0}{\partial z_0} \frac{\partial z_0}{\partial h_0} \tag{30}$$

, where $\delta_1 \in \Re^{8x8x8}$

$$\delta_{a_0} = \delta_1 \frac{\partial h_1}{\partial a_0} = zero - padding(\delta_1, 4) * flip(W_1) \tag{31}$$

, where $\delta_{a_0} \in \Re^{12x12x4}$ $zero - padding(\delta_0, 4)$ pads the δ_1 with zeros by kernel size-1 times (4 in our case) in both first two dimensions and $flip$ function flips the kernel W_1 both row-wise and column-wise. [JAU]

$$\delta_{MP_0} = \delta_{a_0} \frac{\partial a_0}{\partial z_0} \tag{32}$$

Epochs	Batch Size	Learning Rate	Momentum Coeff.
2	8	0.01	0.9

Table 3: Sanity check training hyperparameters.

, where $\delta^{MP_0} \in \Re^{24x24x4}$ and $\frac{\partial a_0}{\partial z_0}$ is the derivative of MaxPooling function and upsamples the data.

$$\delta_0 = \delta_{MP_0} \odot \frac{\partial z_0}{\partial h_0} \tag{33}$$

, where $\delta_0 \in \Re^{12x12x4}$ and $\frac{\partial z_0}{\partial h_0}$ is the derivative of ReLU function which returns 1 if the element is greater than zero, otherwise returns zero.

$$\frac{\partial SCE_L}{\partial W_0} = X * \delta_0 \tag{34}$$

$$\frac{\partial SCE_L}{\partial b_0} = \delta_0 \tag{35}$$

1.4 Question 4

The MNIST dataset is also trained using PyTorch framework. As optimizer, SGD is adapted. The training parameters can be found in Table 3. After two epochs **98% test accuracy** is obtained. The same initial parameters are **not** used.

1.5 Question 5

ReadME file for each part is provided with the code submission.

2 Part2

2.1 Question 1

The designed neural network architecture can be found in Table 4. As optimizer, SGD is adapted. The training hyper-parameters are shown in Table 5.

FC1	input dim: 2, output dim: 8, activation: ReLU
FC2	input dim: 8, output dim: 16, activation: ReLU
FC3	input dim: 16, output dim: 32, activation: ReLU
FC4	input dim: 32, output dim: 16, activation: ReLU
FC5	input dim: 16, output dim: 8, activation: ReLU
FC6	input dim: 8, output dim: 2, activation: Linear

Table 4: Designed network architecture.

The learned decision boundary (See Figure 1)was able to separate the dataset successfully. The test accuracy after 100 epochs was **0.9**. There was not any learning rate scheduler and early stopping is not adopted.

Epochs	Batch Size	Learning Rate	Momentum Coeff.
100	16	0.1	0.9

Table 5: Part2 Training hyperparameters.

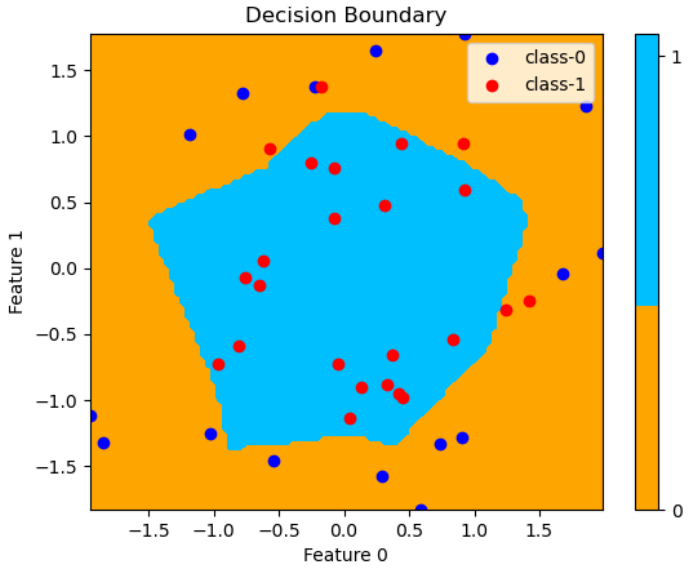

Figure 1: Learned decision boundary and testset.

2.2 Question 2

It is possible to obtain a non-linear decision boundary with ReLU activation function.[Ye]

Intuition: Rectified Linear Unit (ReLU) is a piece-wise linear function that collapses negative values to zero and leaving positive values as they are. On the other side, the fully connected layers are affine functions and can learn affine transformation functions like rotation and translation. Therefore, fully connected layers can rotate and translate the dataset whereas ReLU function collapse some of them.

Consider the following 1-D dataset:

$$X = \begin{pmatrix} 1 \\ 3 \\ 4 \\ 5 \\ 7 \\ 8 \end{pmatrix}, y = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \tag{36}$$

, where $X \in \Re^{5x1}$ is the data and $y \in \Re^{5x1}$ is the labels.

There is not way to separate this sequence with a linear line. However consider the following operations:

$$X_1 = X_0 - 1, where X_0 = X \tag{37}$$

$$X_2 = ReLU(X_1), where X_1 = \begin{pmatrix} 0 \\ 2 \\ 3 \\ 4 \\ 6 \\ 7 \end{pmatrix} \tag{38}$$

$$X_3 = -1 * X_2, where X_2 = \begin{pmatrix} 0 \\ 2 \\ 3 \\ 4 \\ 6 \\ 7 \end{pmatrix} \tag{39}$$

$$X_4 = X_3 + 5, where X_3 = \begin{pmatrix} 0 \\ -2 \\ -3 \\ -4 \\ -6 \\ -7 \end{pmatrix} \tag{40}$$

$$X_5 = ReLU(X_4), where X_4 = \begin{pmatrix} 0 \\ 3 \\ 2 \\ 1 \\ -1 \\ -2 \end{pmatrix} \tag{41}$$

$$X_5 = \begin{pmatrix} 0 \\ 3 \\ 2 \\ 1 \\ 0 \\ 0 \end{pmatrix} \tag{42}$$

$$max(0, (W^T x + b))$$

operation transforms the data and discards the negative values while leaving the remaining the same.

References

[JAU] Pierre JAUMIER. Backpropagation in a convolutional layer. https://towardsdatascience.com/backpropagation-in-a-convolutional-layer-24c8d64d8509. Accessed: 2022-04-12.

[Ye] Andre Ye. Finally, an intuitive explanation of why relu works. https://towardsdatascience.com/if-rectified-linear-units-are-linear-how-do-they-add-nonlinearity-40247d3e4792. Accessed: 2022-04-12.