CSE803 HW5

Javen W. Zamojcin
zamojci1@msu.edu

2024, November 4

1 Question 1: Fashion-MNIST Classification

For my network architecture, I defined three convolutional layers and two linear layers. Each convolutional layer included a kernel size of 3, a stride value of 1, a padding value of 1, a 2D convolutional module, a ReLU operation, a 2D MaxPool module with kernel size and stride values of 2, and a Dropout module with zero probability. The convolutional layers had input-output dimensions of $(1, 32), (32, 64), (64, 128)$. The linear layers had input-output dimensions of $(4 * 4 * 128, 625), (625, 10)$, with a ReLU operation in-between.

The training hyper-parameters for my best model included a learning rate of 0.001, a weight decay of $1e - 4$, 15 epochs, and a batch size of 64. The final testing accuracy was 0.9189, and the training and validation losses over the epochs may be seen in Figure 1. We can see from the increasing validation losses that the number of training epochs was probably too large and resulted in over-fitting, decreasing the potential testing accuracy.

2 Question 2: Activation Visualization

2.1

The architecture of my model was similar to the previous question's model. The base sequential layer consisted of three 2D convolutional layers with dimensions (1, 32), (32, 64), (64, 128), each followed by a ReLU operation and 2D MaxPool operation with kernel size of 2 and stride value of 2. Each convolution layer has a kernel size of 5, stride value of 1, and padding value of 2. Then the base layer is followed by the average pooling and linear layers for the final output.

The training hyper-parameters used were `CrossEntropyLoss` for the criterion, the `Adam` optimizer with a learning rate of 0.001 and weight decay of $1e-4$, and 6 epochs. The final testing accuracy with this model configuration was 0.803.

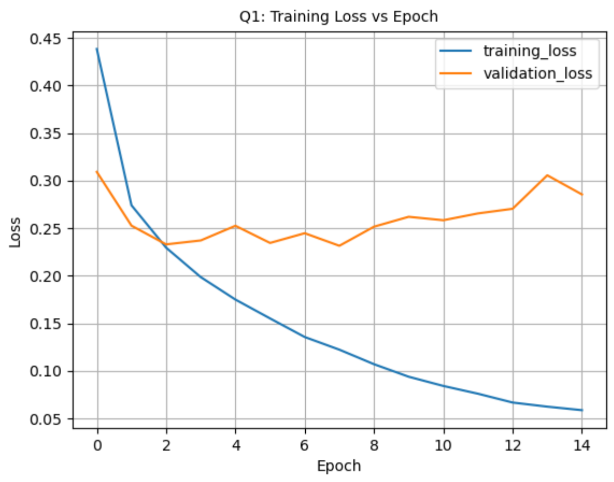


Figure 1: Q1: Training and Validation Losses

2.2

The correctly classified image I used for the activation map visualization was index 3 in the unordered testing dataset. Figure 2 depicts this image as well as its corresponding activation map for each class.

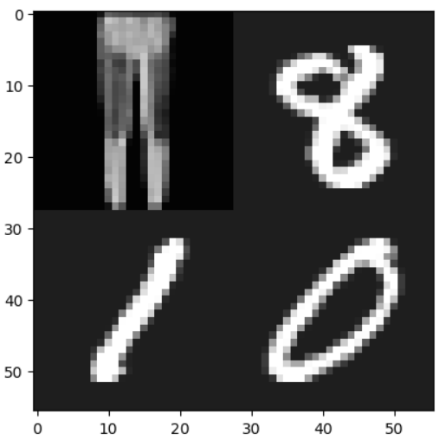
3 Question 3: Semantic Segmentation

3.1

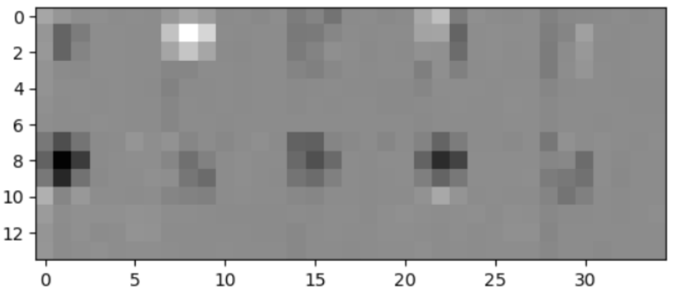
For the architecture of my model, I chose a U-Net implementation. Each layer had a kernel size of 3 and padding value of 1. The model was broken into three subclasses, `UNet`, `Up`, and `Down`, where the latter two are down/up sampling layers, respectively. Each class has a sequential convolutional layer including a 2D convolution, a 2D batch normalization, followed by a ReLU operation.

The `Up` class forward-feeds by upsampling the first input data parameter using a bilinear mode and scaling it to twice its size, followed by the convolution operation. The `Down` class forward-feeds simply by first performing a 2D MaxPool operation (to 2 channels) on the input data, followed by the convolution operation.

The `UNet` class is defined with a (3, 64) convolution layer, three down-sampling layers with input-output ranges of (64, 256), three up-sampling layers with input-output ranges of (512, 32), and finally a (32, 5) convolutional layer. `UNet` forward-feeds by sequentially activating the aforementioned layers, followed by



(a) Correctly Predicted Input Image



(b) Activation Map

Figure 2: Q2: Correct Image Activation Map

a sigmoid activation function for the final output.

3.2

For training hyper-parameters, my best model used 15 epochs, the `Adam` optimizer, `CrossEntropyLoss` for the criterion, a learning rate of 0.0001, a weight decay of 0.000001, a batch size of 32, and the aforementioned U-Net layer dimensions. Additionally, the training dataset used training image pairs (0, 905), the validation dataset used testing images (0, 57), and both the AP dataset and testing dataset used testing images (57, 114). Figure 3 depicts the training and validation losses over time for this model.

3.3

For the average precisions calculated from evaluating my best model on the test/AP dataset, I had the following results:

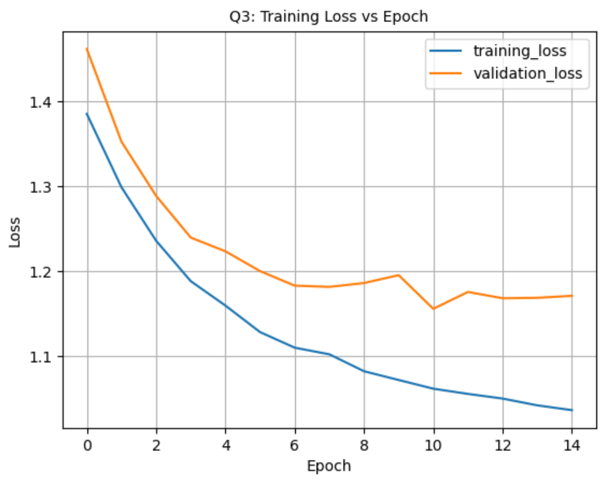


Figure 3: Q3: Training Losses

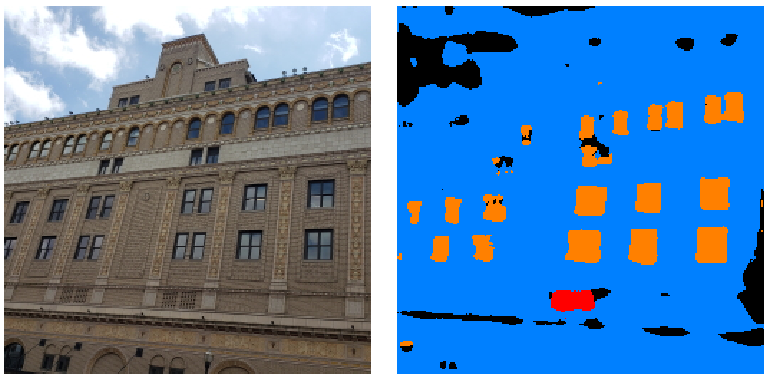
$AP = \{0.64846, 0.76661, 0.06442, 0.85549, 0.63966\}$

As you can see, four out of the five classes had APs far above the minimum required 0.45 AP threshold. Class three, however, only had 0.06442 AP. Why this one class scored so low, I'm not sure. I experimented with many different hyper-parameters and U-Net layer configurations, to no avail. Training for more epochs only marginally increased its score. However, in the homework description, the instructions were unclear whether only one class or all five classes needed an AP of above 0.45.

3.4

Included in Figure 4 are my custom building input image and its predicted label image from my best model. The photo I took was of the side of a brick building, with no porches, patios, balconies, pillars, or doors, and with a variety of window shapes. Additionally, the photo was at a slanted angle to make it more interesting.

The resulting label prediction image did a fairly good job with capturing the windows; however, the group of smaller, clear-colored windows in the top-left it seemed to miss most of. The difference between these ones and the windows in the top-right seems to only be the reflection color. The top-right all had dark tints from the reflection angle, and were all captured much better. Another error was that one of the sets of vents in the building (there were two sets), was classified as a balcony.



(a) Input Image (b) Output Image

Figure 4: Q3: Custom Image Labeling