Digit Recognition with Convolutional Neural Networks

CMPT 412 - Computational Vision

Fitzpatrick Laddaran

January 17, 2023

1 Part 1 - Forward Pass

This section pertains to Part 1 of the project handout.

1.1 Inner Product Layer - Question 1.1

Figure 1 shows the results of calling `test_inner_1()` in `test_components.m`.

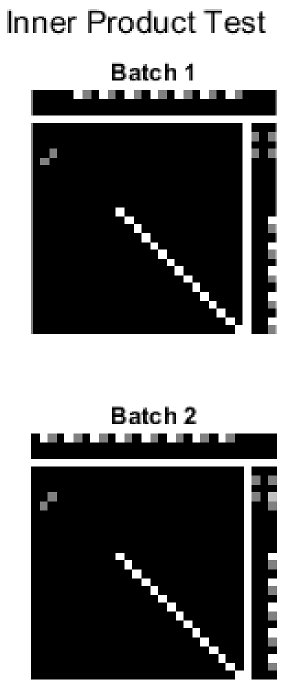


Figure 1: Visualization resulting from `test_inner_1()`.

1.2 Pooling Layer - Question 1.2

Figure 2 shows the results of calling `test_pooling_1()` in `test_components.m`.

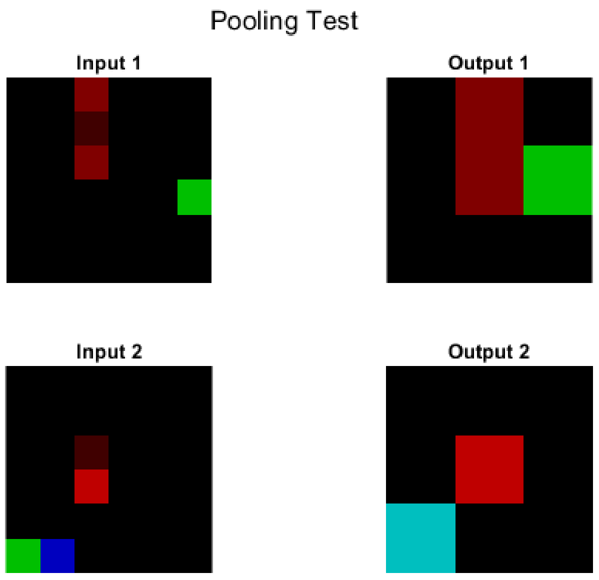


Figure 2: Visualization resulting from `test_pooling_1()`.

1.3 Convolution Layer - Question 1.3

Figure 3 shows the results of calling `test_conv_1()` in `test_components.m`.

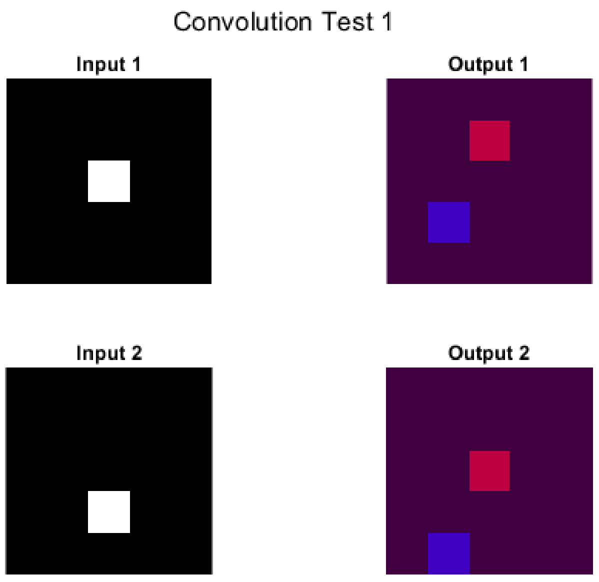


Figure 3: Visualization resulting from `test_conv_1()`.

Figure 4 shows the results of calling `test_conv_2()` in `test_components.m`.

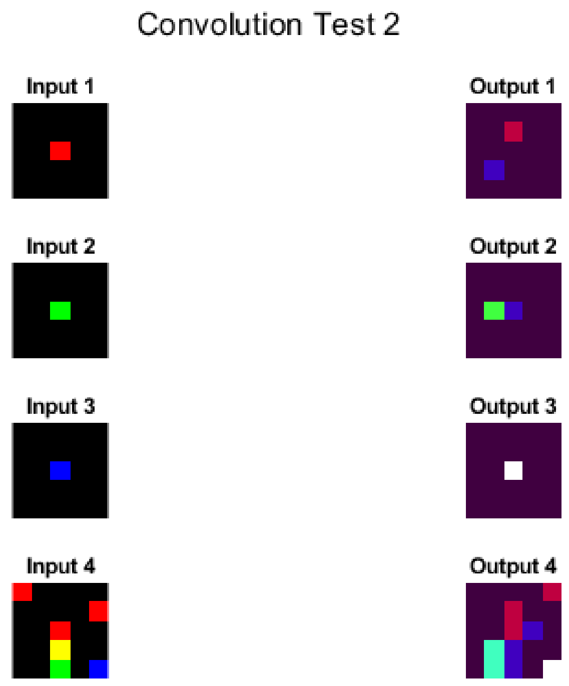


Figure 4: Visualization resulting from `test_conv_2()`.

2 Part 2 - Back Propagation

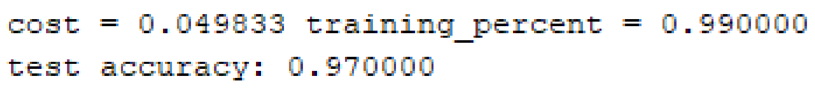
This section pertains to Part 2 of the project handout. There is nothing to include in the report.

3 Part 3 - Training

This section pertains to Part 3 of the project handout.

3.1 Training - Question 3.1

After running `train_lenet.m` for 3000 iterations, the test accuracy is 97%. Figure 5 illustrates the system output from running `train_lenet.m` for 3000 iterations.



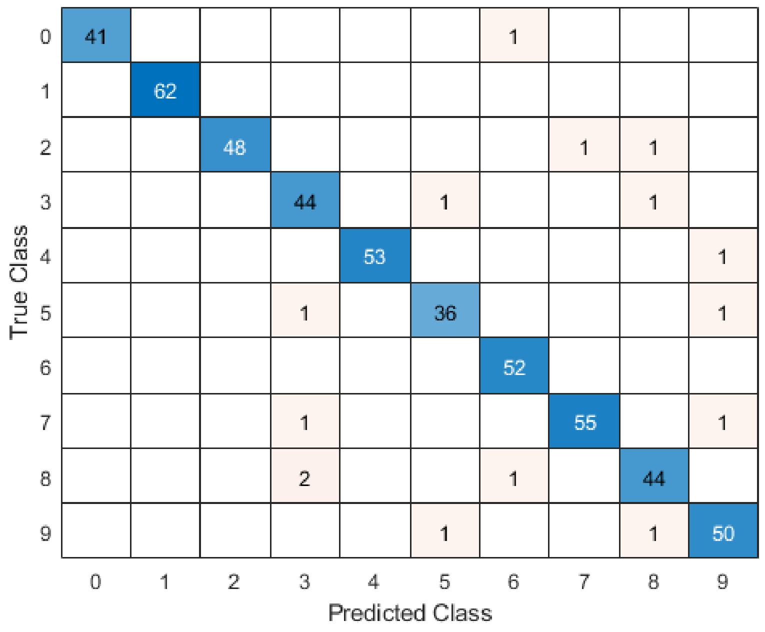
```
cost = 0.049833 training_percent = 0.990000
test accuracy: 0.970000
```

Figure 5: System output after running `train_lenet.m` for 3000 iterations.

3.2 Test the network - Question 3.2

Figure 6 displays the confusion matrix resulting from a modified script of `test_network.m`. The confusion matrix compares the predicted output of the network, and the actual number displayed on the image.

Evidently, the top most confused pair is number 8 and 3. Two 8s have been predicted to be 3. This is likely because of the two loops that compose the number 8. It seems like the network thinks that the top part and the bottom part of the number 8 creates the incomplete loops that create the number 3. Perhaps in the sample input, there are 8s that have incomplete loops.



True Class \ Predicted Class	0	1	2	3	4	5	6	7	8	9
0	41						1			
1		62								
2			48					1	1	
3				44		1			1	
4					53					1
5				1		36				1
6							52			
7				1				55		1
8				2			1		44	
9						1			1	50

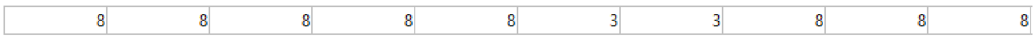
Figure 6: Confusion matrix outputted by `test_network.m`.

Arguably, the remainder of the incorrect predictions are in equal place; there is no second top most confused pair. However, it does seem like that the network are predicting more 8s and 9s than the other numbers. I suspect that it is because of the loop-like pattern that causes ambiguity in image samples where the writing of the numbers are cursive in structure.

3.3 Real-world testing - Question 3.3

I have manually drawn 15 images. The first five examples are 1000×1000 images, and the last ten examples are 28×28 images. Respectively, the numbers are: 8, 6, 2, 3, 9, 8, 6, 2, 3, 9, 8, 6, 2, 3, 9. The images have been included in the folder `../images/` titled as `ex*`, `*` representing integers from 1 to 15. The script is saved in `../matlab/` titled `test_real_images.m`.

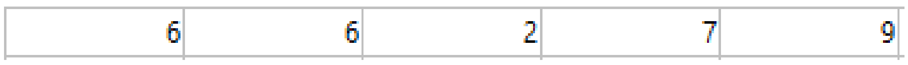
The result of my network on these examples are quite poor. In fact, in the first ten images, only the first 8 was predicted correctly. Figure 7 shows the predicted numbers of my network for the first ten images. Note that these images have white backgrounds.



8	8	8	8	8	3	3	8	8	8

Figure 7: Network predictions from using hand-written images on white background.

For the first 5 images, the poor predictions is likely because of the resulting poor quality images after reducing from 1000×1000 to 28×28; however, I was quite surprised by the results of the next five 28×28 examples. Perhaps it is because of the white background; hence I tried another five 28×28 images on black backgrounds. Figure 8 shows the results.



6	6	2	7	9

Figure 8: Network predictions from using hand-written images on black background.

The system performed better on this last set of five images, predicting 3 out of 5 numbers correctly: 6, 2, and 9. I suspect that this performed better because of my experience in the last part of this project, Part 5 - Image Classification. The bounding boxes were not picking up the connected components in the input images, until I converted the pixel values from white to black, and vice versa. The same intuition applies to these examples.

4 Part 4 - Visualization

This section pertains to Part 4 of the project handout.

4.1 Question 4.1

Figure 9 is a visualization of the output of the second layer—that is, a convolutional layer—of the network.

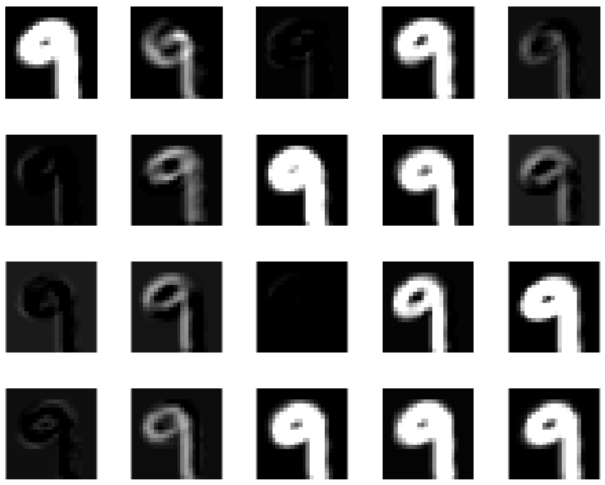


Figure 9: Visualization of the output of the second layer of the network.

Figure 10 is a visualization of the output of the third layer—that is, a ReLU layer—of the network.

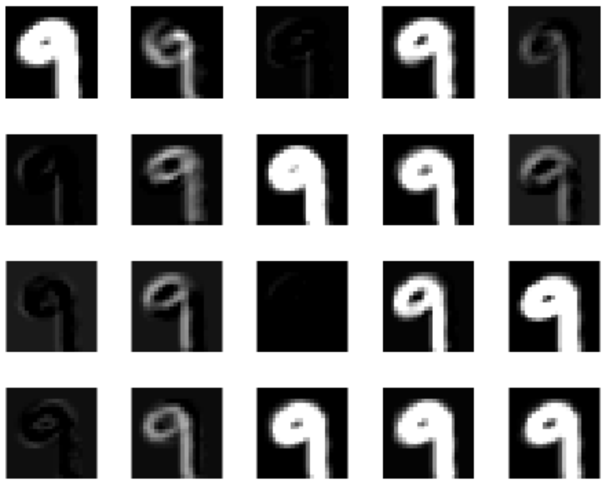


Figure 10: Visualization of the output of the third layer of the network.

4.2 Question 4.2

Comparing the feature maps to the original image, it seems that sub-images in Figure 9 are quite blurry compared to the original image. This is because it is a convolutional layer, where information about the original image is convolved. Figure 10 is identical to Figure 9 because the output (object) of the convolutional layer mostly consists of positive numbers. Therefore, the ReLU layer only changes a few values from negative to 0. Both figures are also different compared to the original input by contrast; it is clear that the number in the original image is harder to identify in some of the sub-images in both figures.

5 Part 5 - Image Classification

This section pertains to Part 5 of the project handout.

Following the approach presented in the lab, the result of my system is shown in the confusion matrix presented in Figure 11.

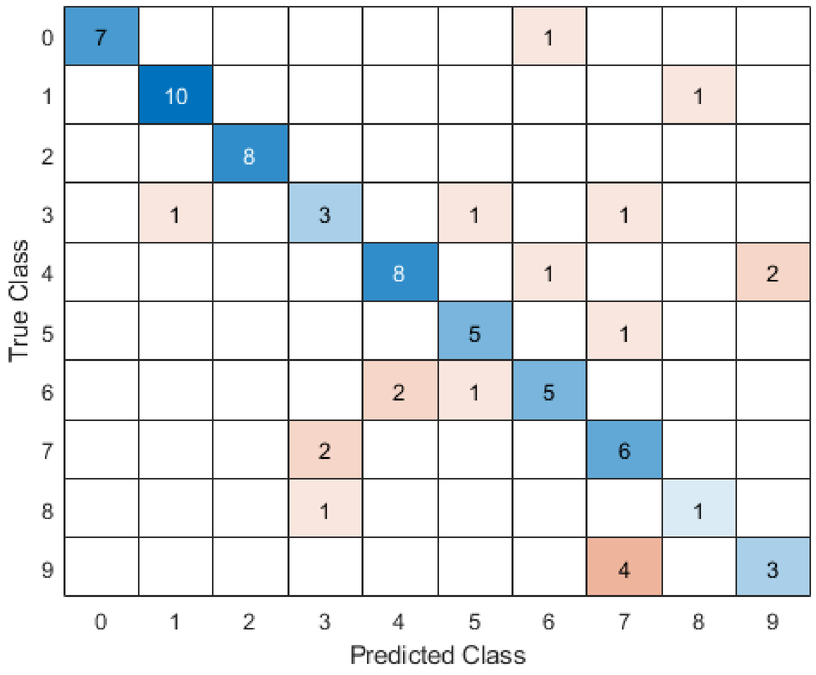


Figure 11: Confusion matrix outputted by `ec.m` using the provided real-world images.

My system performed quite decently, predicting 56 of the 75 input images correctly. The score is about 75%. It seems like a lot of 9s are predicted incorrectly, perhaps due to the loop-like pattern. Since I processed the images by removing isolated pixels and merging connected components, some images may have been morphed to look like 7 and 9 as the loop in the number 9 was filled.