PROJECT REPORT – CMPEN 454

Project 1 - CNN for object recognition

Group Members: Hongshuo Wang, Songyang Mao, Jiafu Chen, Zerui Li

A. Summarize in your own words what you think the project was about. What were the tasks you performed; what did you expect to achieve?

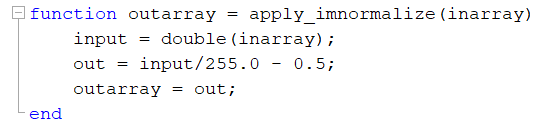
B. Present an outline of the procedural approach along with a flowchart showing the flow of control and subroutine structure of your Matlab code. Explain any design decisions you had to make. Even though the mathematical specification of each part of this project is fairly strict, there are a lot of different ways you could implement each building block in Matlab, ranging from C-like nested for-loop computations, to cleverly vectorized code. Be sure to document any deviations you made from the above project descriptions (and why), or any additional functionality you added to increase robustness or generality of the approach

For this Project 1 which about building a convolutional neural network (CNN) for object recognition in Matlab, we first read the project introduction together and then brainstormed it. Then we reached a consensus that this project could be mainly divide into several parts. First of all is The Basic Operations. In this part the tasks are divided in to six parts, which are Image Normalization, ReLU aka Rectified Linear Unit, Maxpool, Convolution, Fully Connected and Softmax. We found Convolution and Fully Connected are the most challenging part of this project. Since there are four members in our group, the first four tasks are assigned by two of the members and the last two tasks are assigned to the other two members. So this part of the project is the main focus for the first two week, and we complete this division of labor on September 15. Since none of our team members has programming the Matlab before, we are still leaning this new language while we are doing the project, this become a little challenging start for us that we need to read the document of MATLAB to familiar its normal grammar first. The second big part of this project is putting the functions what we wrote before September 25 all together. As we finished each task from previous part, we worked to put them all together to do object recognition to implement an 18-layer CNN that takes a 32 by 32 color image as input and produces 10 object class probabilities as output.

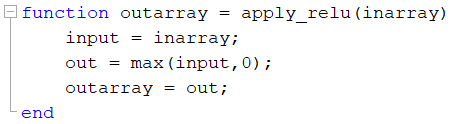
Here is an outline of the procedural approach along with a flowchart showing the flow of control and subroutine structure of Matlab code:

1. The Basic Operations

In this part, we have development 6 MATLAB functions as the equipment components for convolutional neural net, which are: Image Normalization (1.1), ReLU aka Rectified Linear Unit (1.2), Maxpool (1.3), Convolution (1.4), Fully Connected (1.5) and Softmax (1.6).

1.1 Image Normalization

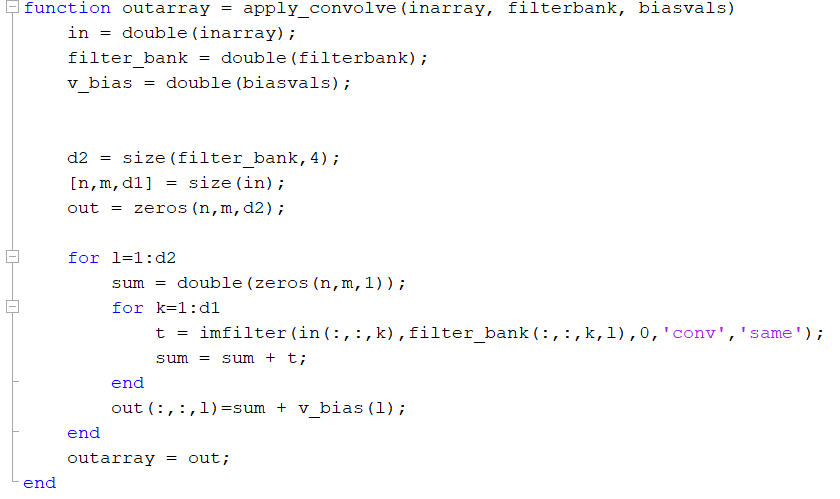
In this function, the input is the array given for the information of the image and we just simply suing the equation that the project description given, which is Out(i,j,k) = In(i,j,k) / 255.0 − 0.5, and convert it to MATLAB code in the function.

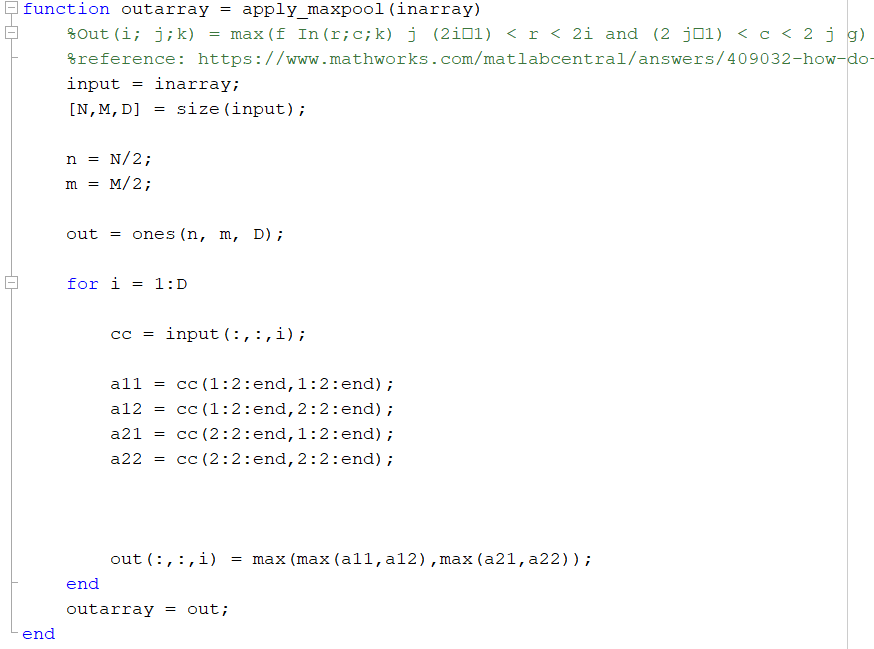
1.2 ReLU

In this function, the input is the array for image information and we just simply suing the equation that the project description given, which is Out(i,j,k) =max(In(i,j,k),0), and convert it to MATLAB code in the function. We using the max() statement in MATLAB to compare the values in input array inarray.

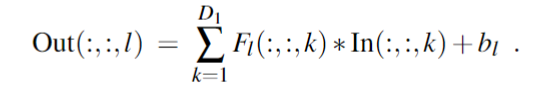
1. 3 Maxpool

The equation that project description given is Out(i, j ,k) =max({In(r, c, k) | ( 2i – 1 ) < r < 2i and ( 2j – 1 ) < c < 2j }), however, as we know MATLAB could not directly achieve this math equation by a statement. So we tried another way to solve this question. We have r in range ( 2i - 1, 2i ) and c in range ( 2j - 1, 2j ), based on this we wrote a loop in MATLAB to find the max value in the pool by comparing near 4 positions, and this is very effective. The output will return the maximum value of the image in a 2\*2 area to reduce the spatial size of the image to a level therefore the neural network could be much easier to manage the input image.

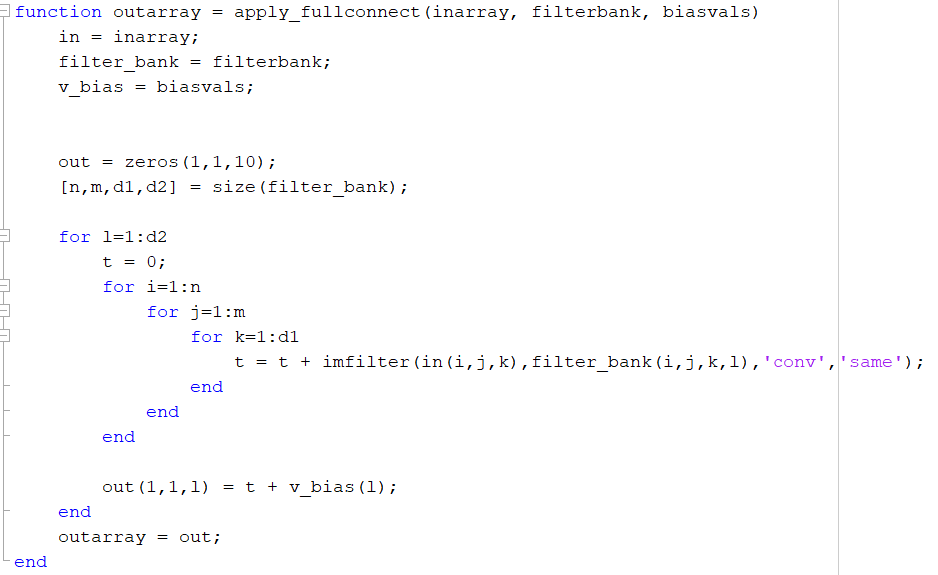


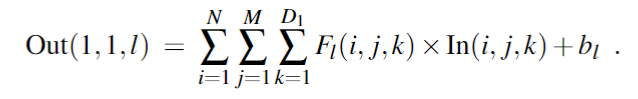
1.4 Convolution

As we have the given equation:



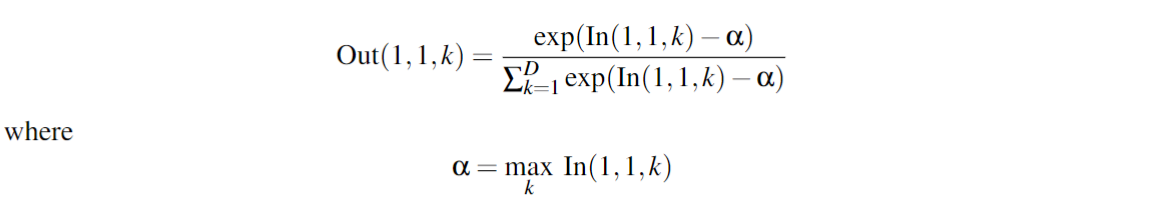
for computing the l-th channel of the output image using summations of 2D convolution, but MATLAB cannot directly process this as we know. So we wrote a loop as K in range(1, D1) process such Fl \* In, and add the bias. Since we have D2 numbers for output, we also need a loop outside the computing loop to add these number together to complete the convolution function.

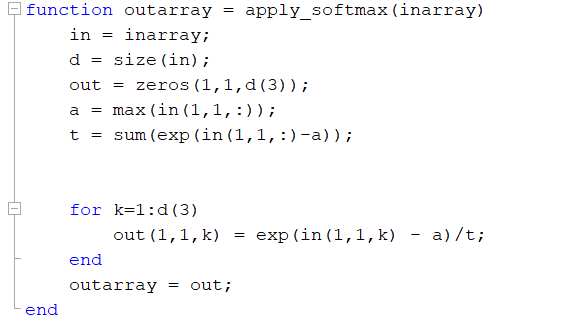
1.5 Fully Connected

Since there has total D2 numbers of filter banks, so we have a loop as from 1 to D2, since the equation is below:

So we have other 3 loops inside the 1 to D2 loop, which are i in range 1-N, j in range 1-M, and k in range 1-D1. Inside the loop, we times the value for Fl\*In together and add bias for each one. Then, we add these values together to get the output array, as form of 1 \* 1 \* D2.

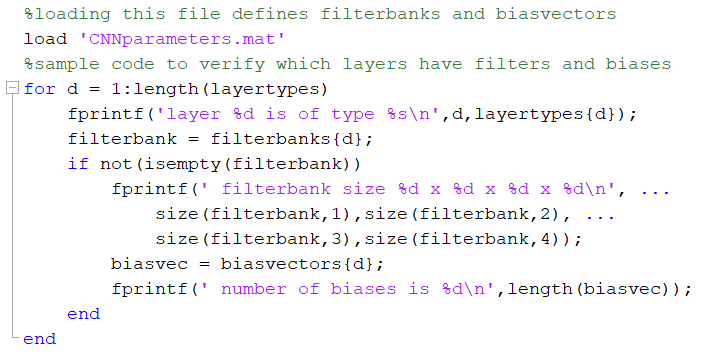
1.6 Softmax

In this function, our main job is to achieve with this equation:

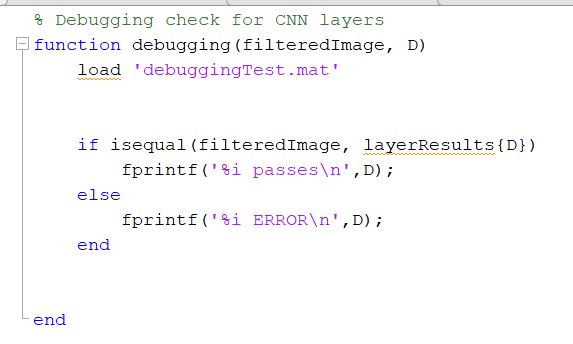
To avoid massive code of calculating this, we first using code to represent the value of a and the sum of exp(In( 1, 1, k) – a), then we using a loop for k in range(1, D) to get the according k for Out(1, 1, k) and return a set of probabilities from 0 to 100% for the prediction as the evaluation neural network for the input array about the information for the image.

2. Putting It All Together

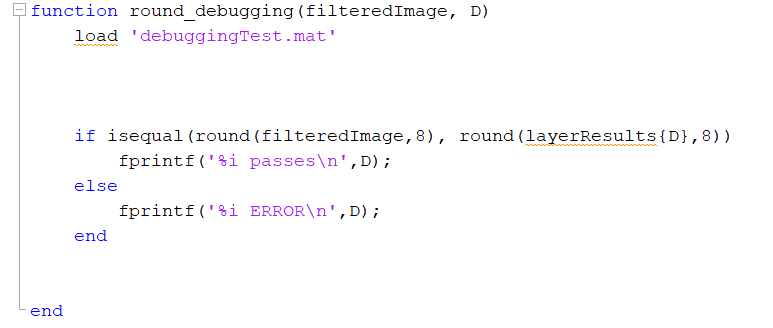
In this part, we have development 3 MATLAB functions for putting 6 functions which are completed form previous section together, which are: Input Parameters (2.1), Debugging (2.2), Round Debugging (2.3), Debugging Main (2.4), and Combination (2.5):

2.1 Input Parameters

What we are mainly do in this function is put parameters into it to load data set as input of the convolutional neural net for object recognition.

2.2 Debugging

This function we design to debugging our code to make sure everything is worked.

2.3 Round Debugging

Little different with function 2.2 debugging, this function is specially design for the debugging of convolution to test and make sure that function would worked.

2.4 Debugging Main

In this function, we debugged the connected three neural network each has six layers: Image Normalization, ReLU aka Rectified Linear Unit, Maxpool, Convolution, Fully Connected and Softmax to test and make sure that the neural network could work

2.5 Combination

In this function we connected three neural network each has six layers: Image Normalization, ReLU aka Rectified Linear Unit, Maxpool, Convolution, Fully Connected and Softmax

3. Quantitative Evaluation

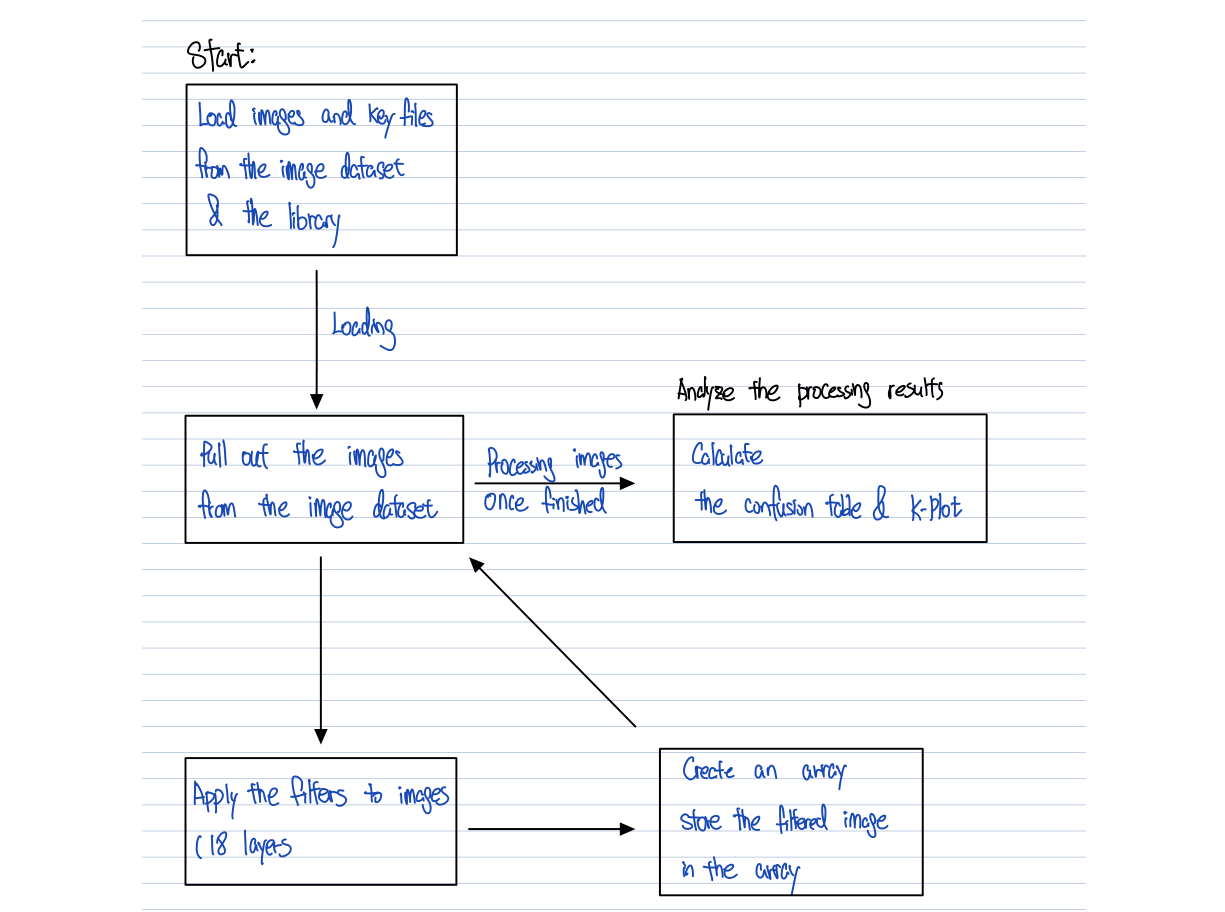
In this part, we have development 2 MATLAB functions for quantitative evaluation, which are: Confusion Matrix (3.1), K-plot (3.2):

3.1 Confusion Matrix

In this function we created a 10 \* 10 table to record the predictor predict object with label to which kinds of classes of the objects for each image in the data set that project is given. The output would be a 10 \* 10 matrix that recorded the specific numbers of the prediction details for each class.

3.2 K-Plot

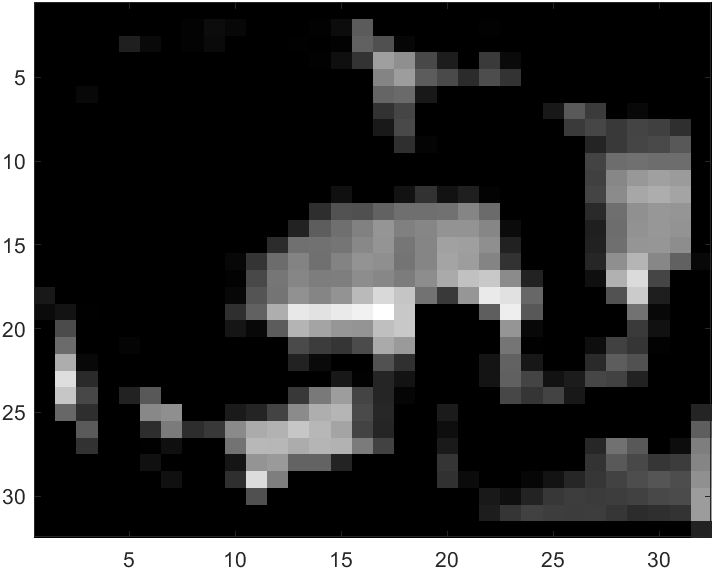
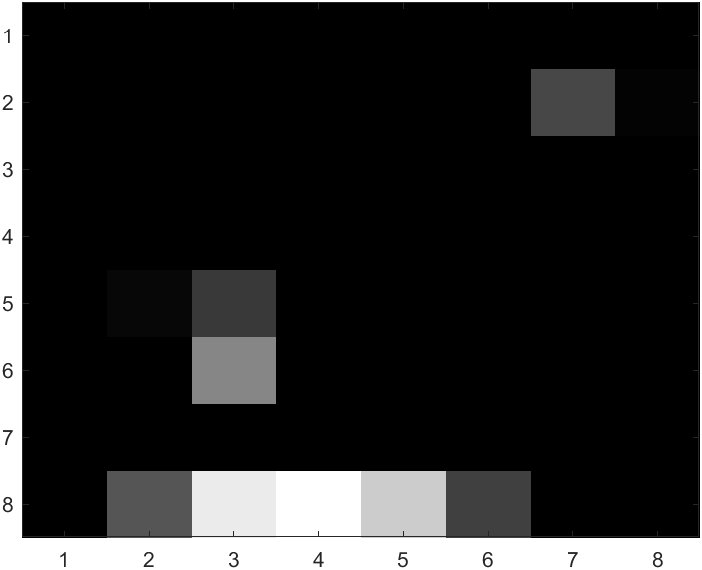
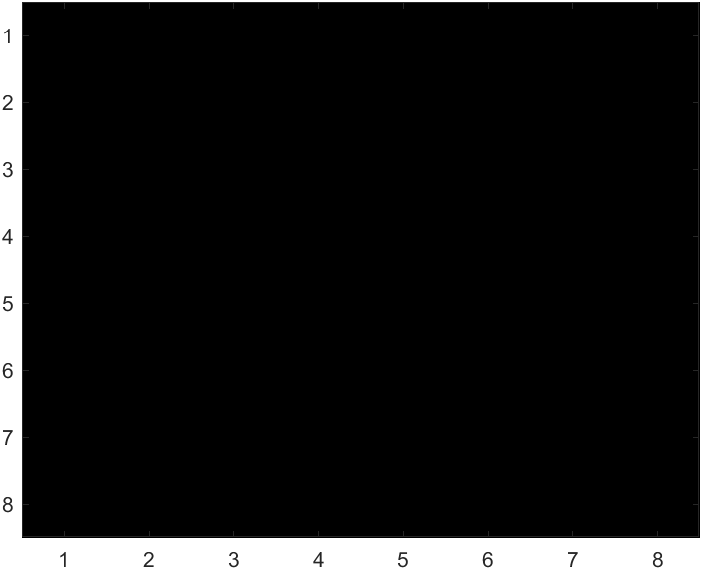
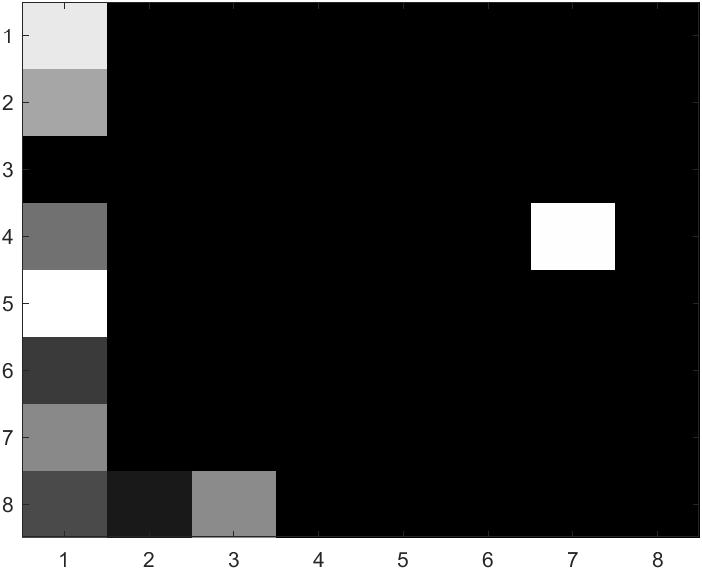
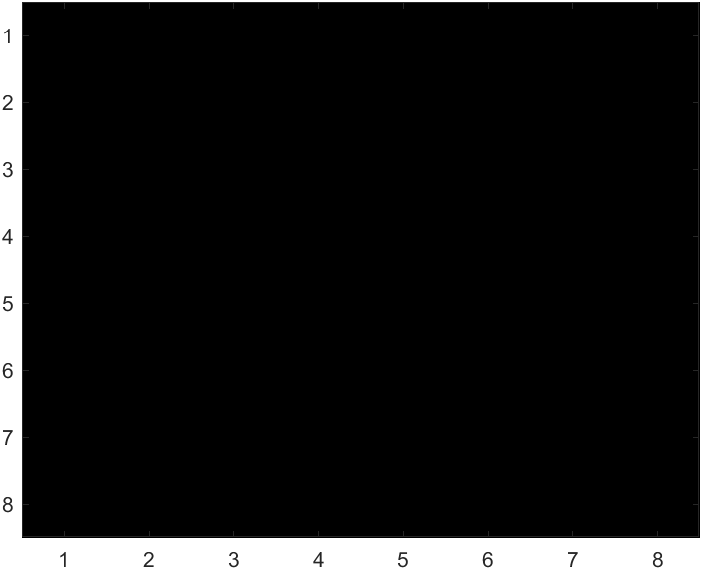
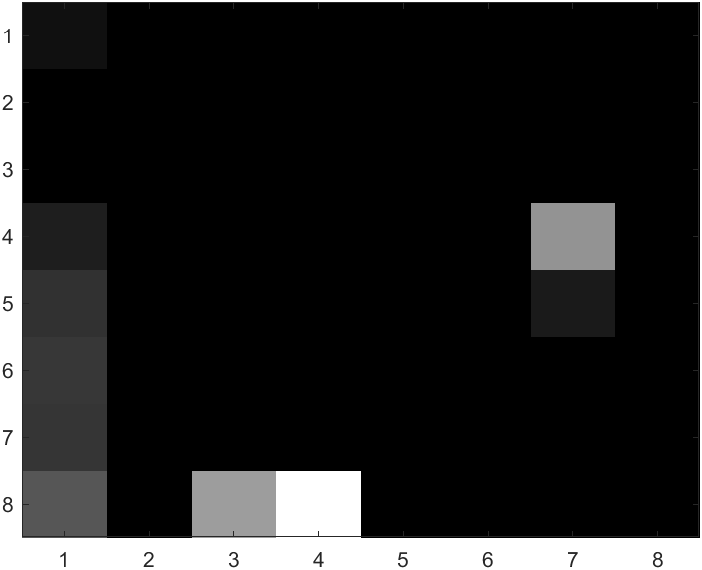
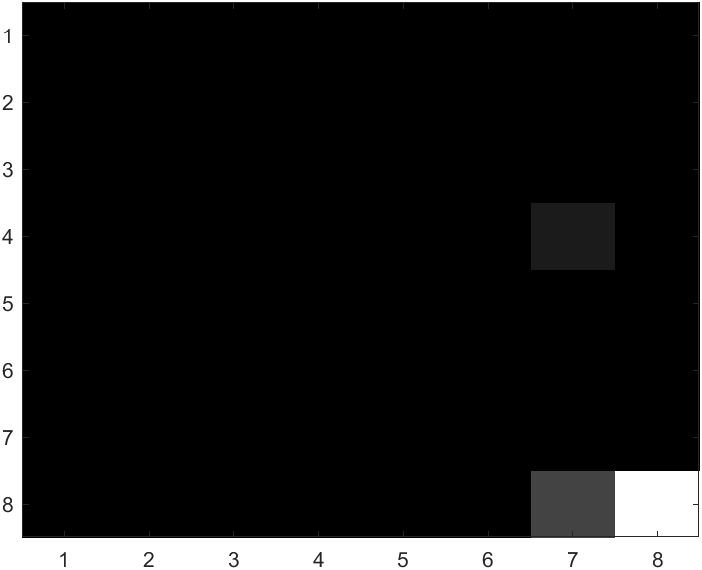
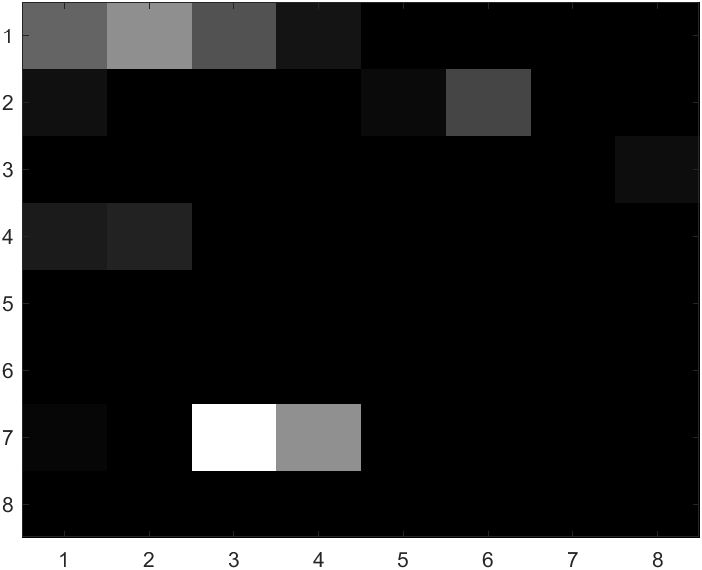
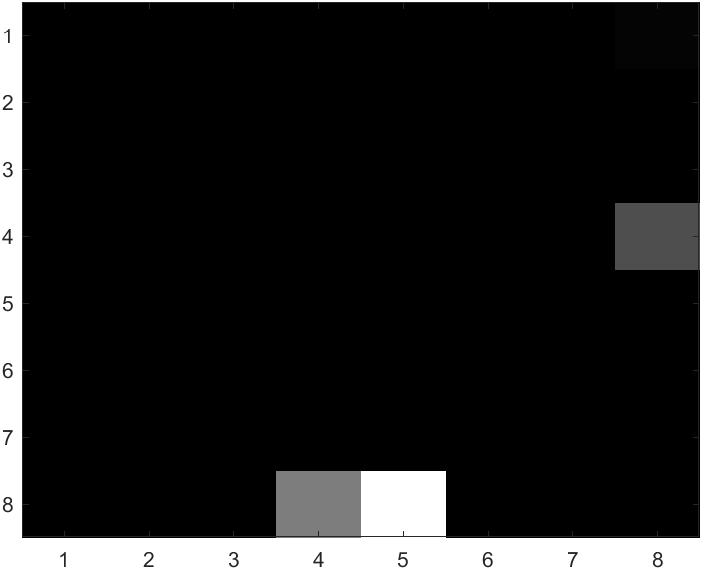
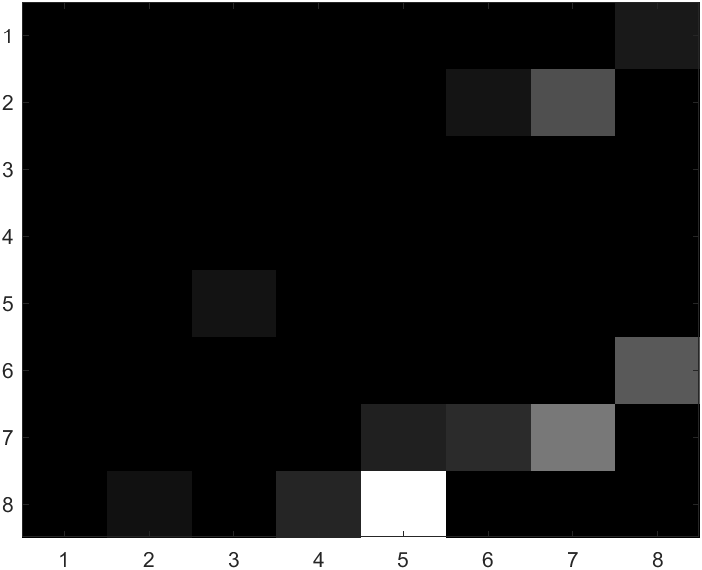
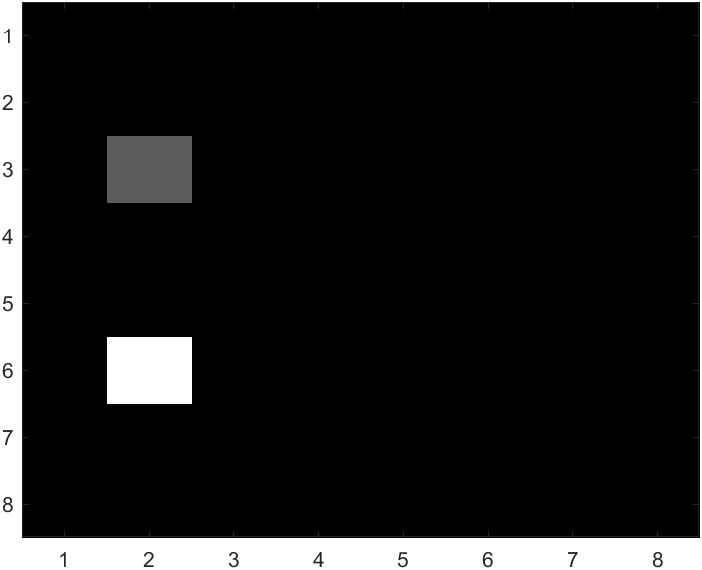
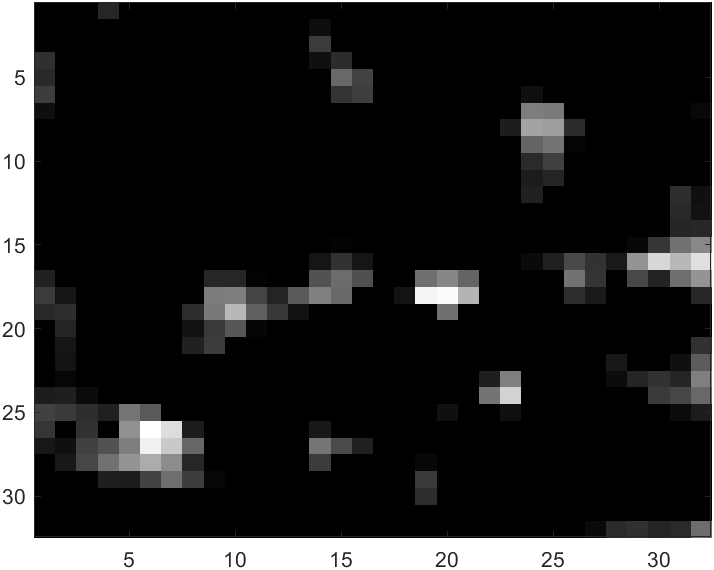
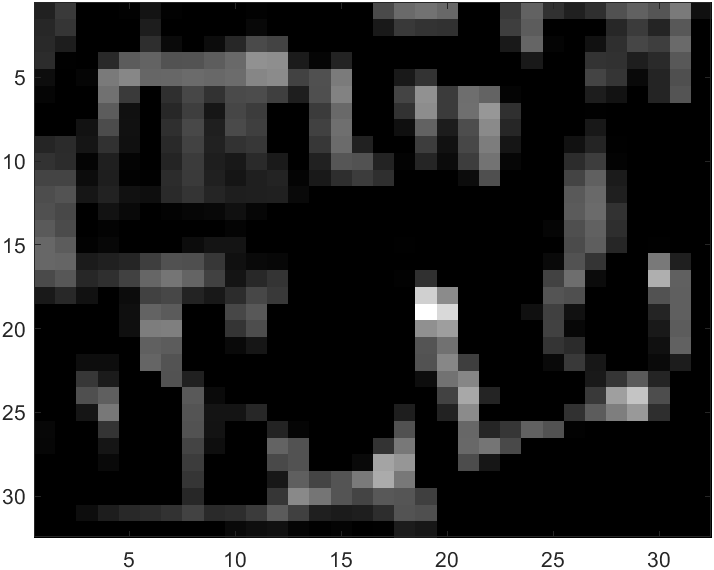
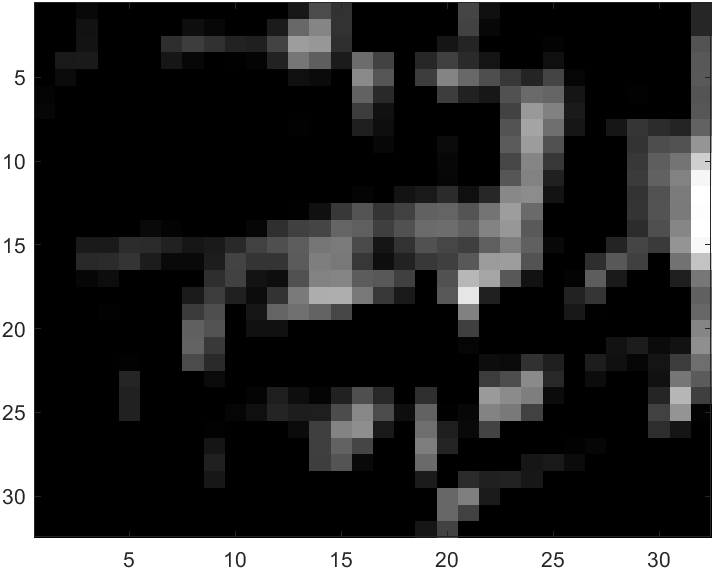
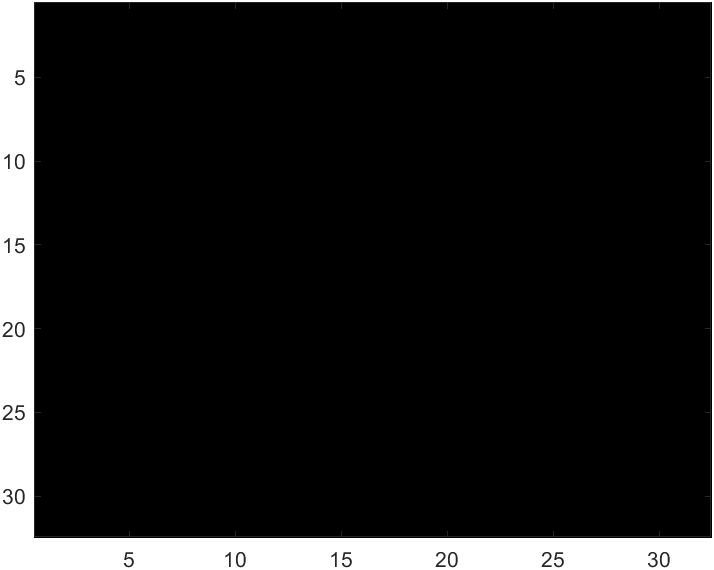
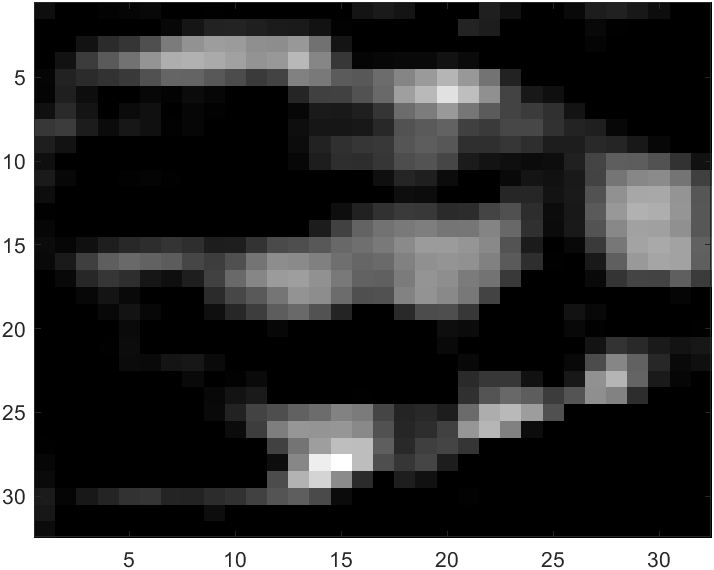
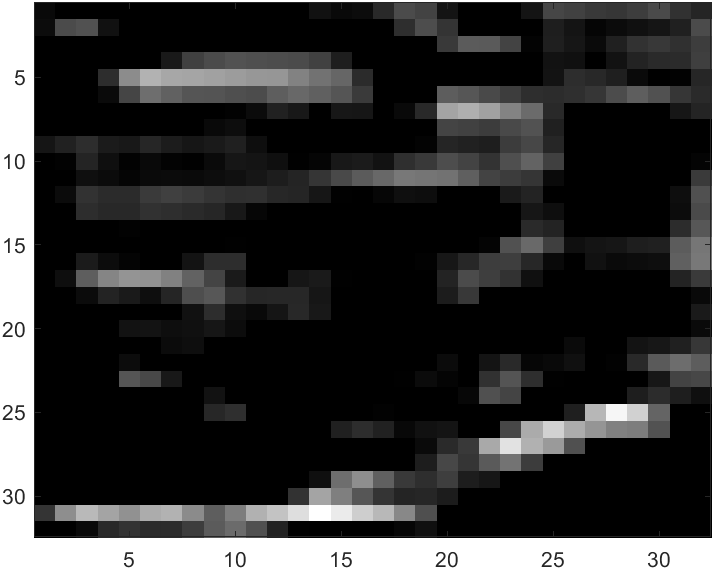
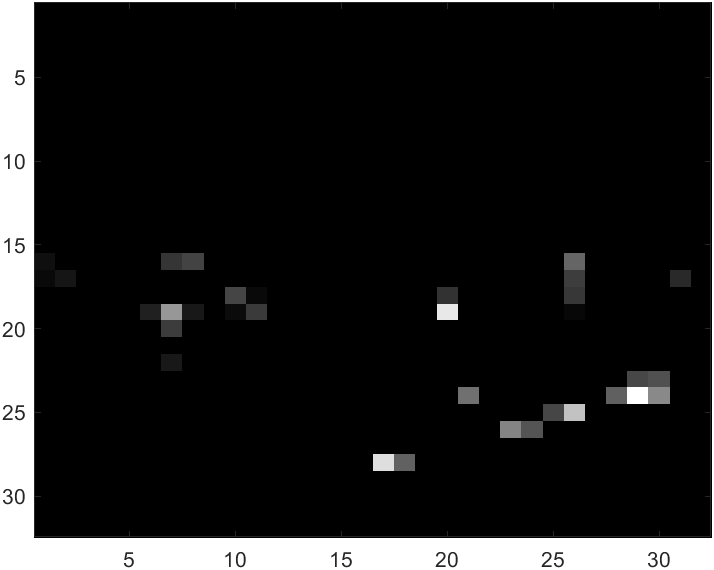
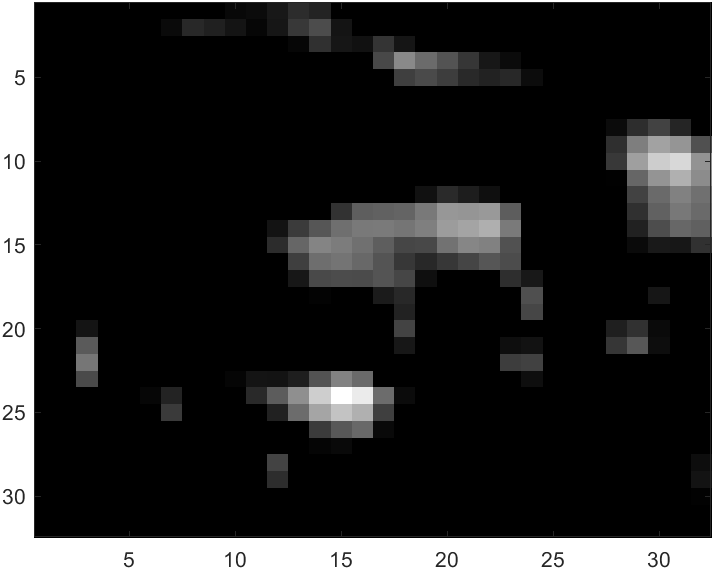
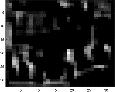
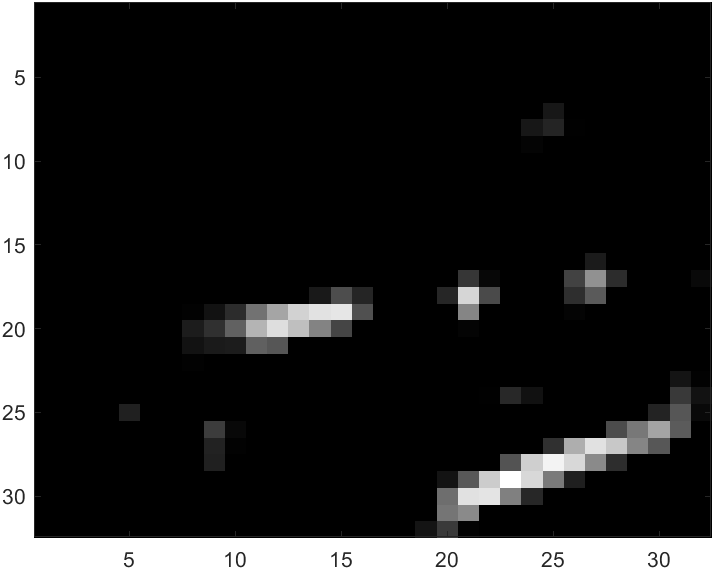
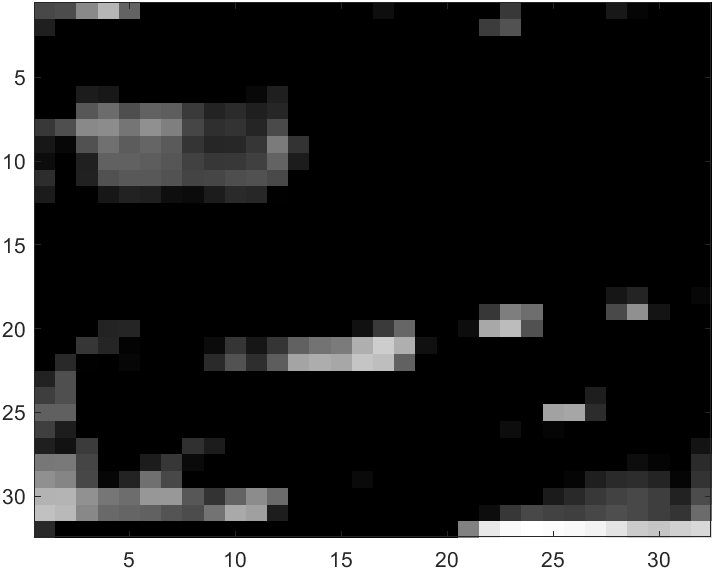
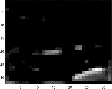
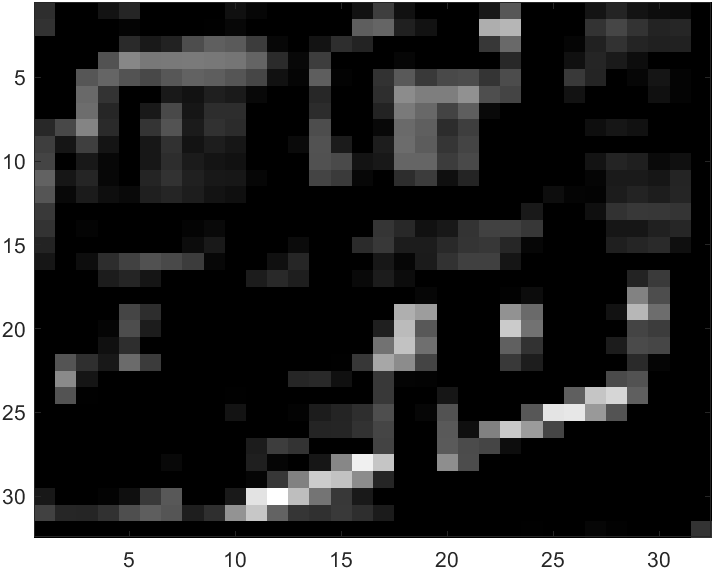
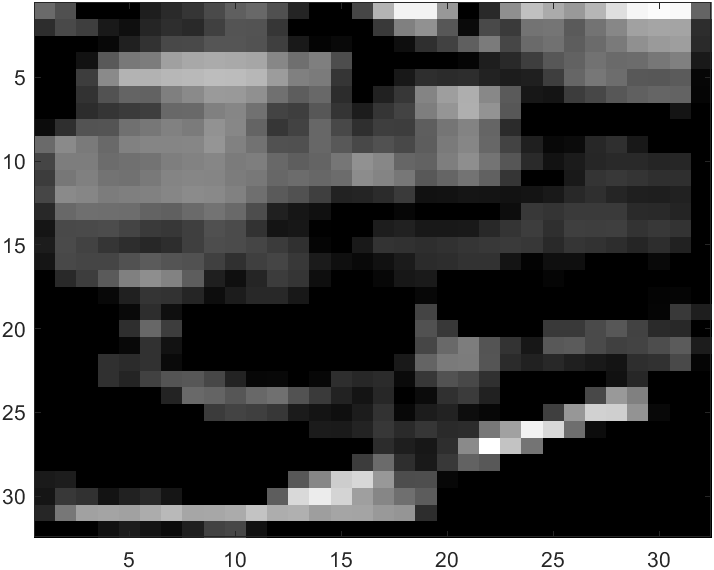
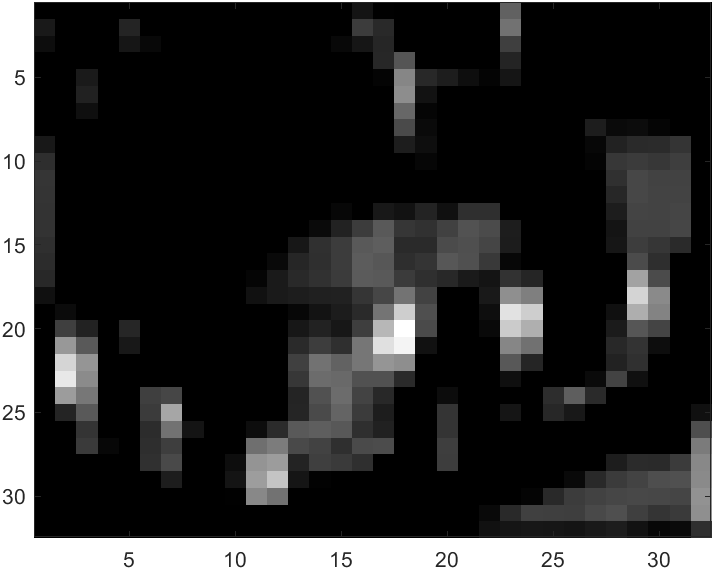
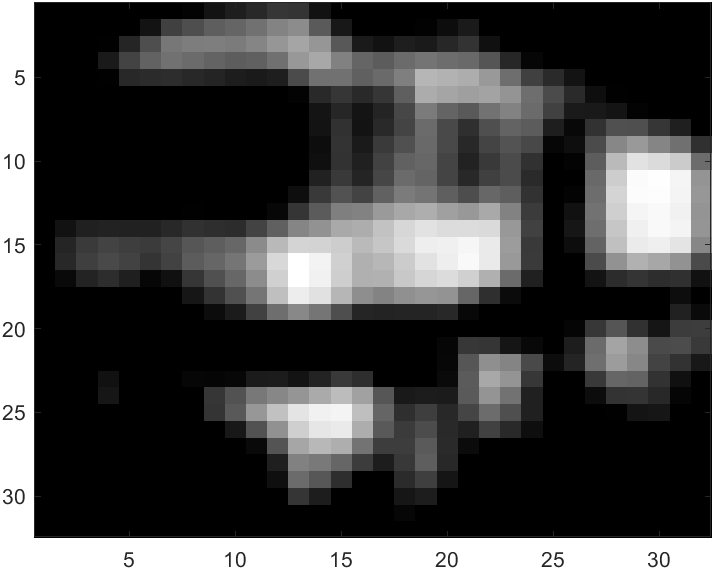
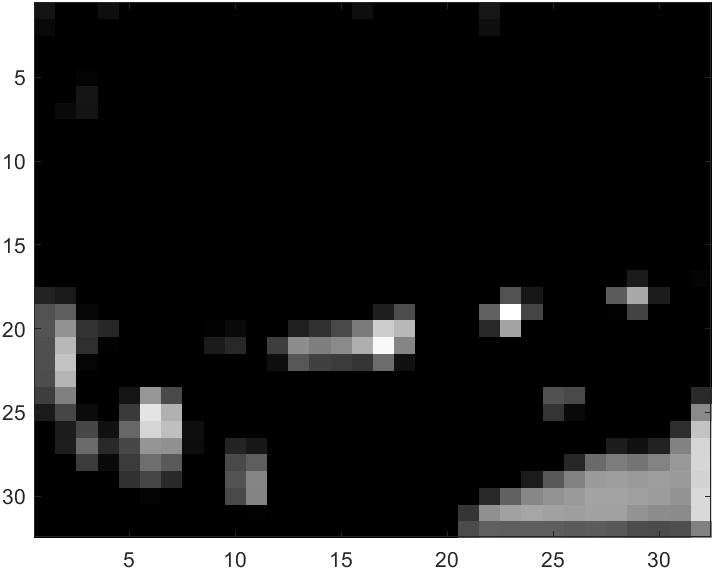
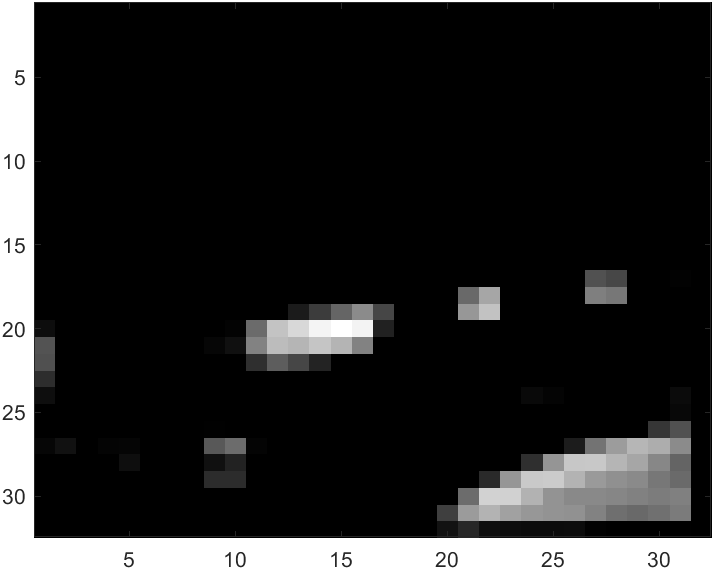
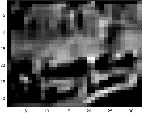
In this function we using the plot graph shows the classification accuracy with respect to the top-k classes. The input is an array with size (1,1,10), by a double layer loop we can have the max accuracy percentage for each of the situation as 1 to 10.

4. Implementation Process

C. Experimental observations. What do you observe about the behavior of your program when you run it? Does it seem to work the way you think it should? Run the different portions of your code and show pictures of intermediate and final results that convince us that the program does what you think it does. Each channel of an intermediate result array in the CNN can be interpreted as a greyscale image. So, for example, the output from layer 2 of the CNN is an array of size 32×32×10, and this could be displayed as 10 small greyscale images, each of size 32×32. CNN practitioners often display these intermediate results and interpret them as images showing the results of learned “feature” detectors. The results from the final softmax layer could be displayed as a bar chart.

Input a truck image

Layer 3:



Layer 5:

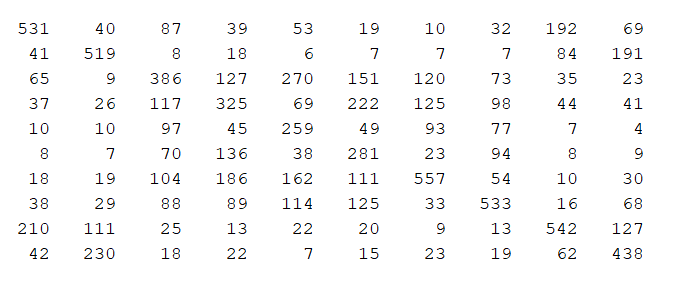
Layer 13:

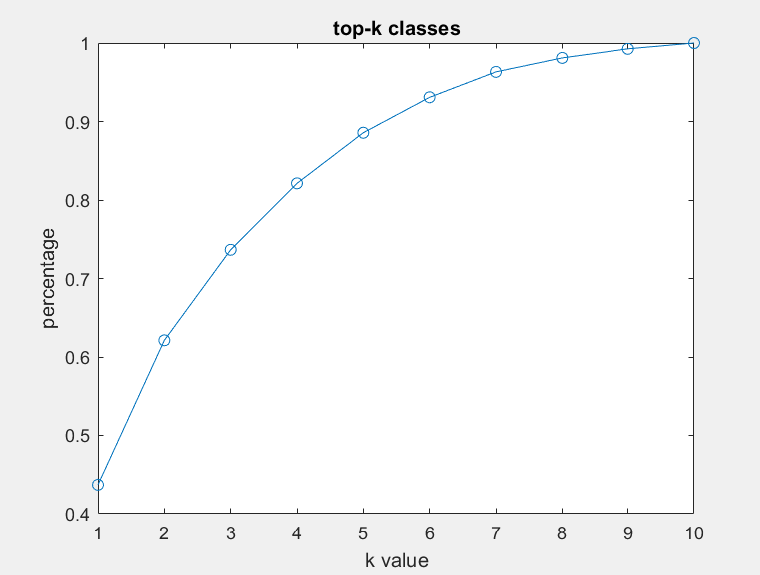
Softmax: airplane 0.0041, automobile 0.7332, bird 0.0011, cat 0.0070, deer 0.0012, dog 0.0021, frog 0.0066, horse 0.0019, ship 0.0128, truck 0.2300

In most cases, the program will work as expected. As we randomly picked an image about a truck and want to check its greyscale image from the processing of layer 3, 5, and 13, we find that as the layer become deeper the features would be much lesser so that it would have more weight in the neural network. Well.. for the prediction result of this image is little bit embarrassing, it mistakenly recognized a truck as an automobile, we believe it might be lots of feature in this graph that similar with automobile rather than a normal truck, so the neural network get cheated.

D. Run performance evaluation experiments as described above to compute the confusion matrix and classification rate. Discuss.

Here is the confusion matrix that we get:



Here is the K-plot for the classification rate that we get:

We have the accuracy for K is 1 to 10: 0.4371, 0.6212, 0.7366, 0.8212, 0.8857, 0.9308, 0.9632, 0.9810, 0.9926, 1.0000.

From the plot, we can have 43.71% as accuracy for just the usual classification accuracy. However, as the plotted value for k = 2, which is the percentage of times that the correct class for an image is one of the 2 classes that have the highest computed probability scores for that image, we have the accuracy increased to 62.12%. As k = 4 the accuracy would higher than 80% and k=6 it has more than 90% for the prediction accuracy.

E. If you are in an exploratory mood, find some images of your own on the web and input them to the CNN to see how well it does on them. To do this, you will need to reduce the image size down to a 32×32×3 color thumbnail image. How would you do this in Matlab? (hint: we discussed generating thumbnail images in one of the lectures). Of course, you will get best results if the images you choose actually contain one of the object classes, and that object should pretty much dominate the image. What happens if you give it an image containing an object that it doesn’t know about? For example, what happens if you input an image of your face? (no matter what the output classification for your face is, I’m sure it will be hilarious). Can you think of a test that you could perform on the output probabilities to extend the classification to include an “unknown” category.

Input airplane:

Output softmax: airplane 0.4665, automobile 0.0233, bird 0.0641, cat 0.1020, deer 0.0358, dog 0.0197, frog 0.0291, horse 0.0715, ship 0.0622, truck 0.1258

For the graph similar with the trained dataset, the model would be accuracy for predicting it.

Input Uncle Sam:

Output SoftMax: airplane 6.3443e-04, automobile 0.0027, bird 0.0068, cat 0.4403, deer 0.0063, dog 0.1911, frog 0.2415, horse 0.0382, ship 5.1766e-04, truck 0.0720

For the graph classification that not similar with the trained dataset, the model would firstly find its largest weight as in the procession from the neural network and try to classify it to one of its trained classification

F. Document what each team member did to contribute to the project. It is OK if you divide up the labor into different tasks (actually it is expected), and it is OK if not everyone contributes precisely equal amounts of time/effort on each project. However, if two people did all the work because the third team member could not be contacted until the day before the project was due, this is where you get to tell us about it, so the grading can reflect the inequity.

Hongshuo Wang: Image Normalization layer, ReLU layer, Putting all together, debugging

Songyang Mao: Softmax layer, Video presentation, Confusion Matrix

Jiafu Chen: Maxpool, Convolution layer, project report part A, C

Zerui Li: Fully Connected layer, K-Plot, project report part B, D, E