EE569 – HW6

Mozhdeh Rouhsedaghat

2726554211 rouhseda@usc.edu

Abstract and Motivation:

Although Convolutional Neural Network is the most widely used tool for image analysis it has some fundamental problems. For example, by manipulating the input data a little bit we can make the CNN fail miserably. (adversarial attack). Besides CNN is a black box and we can't completely explain what it does as the feature go through the CNN nodes. SAAB transform (**S**ubspace **a**pproximation with **a**djusted **b**ias) is introduced to solve these issues.

Approach and Procedures:

PCA is a dimension reduction technique which basically keeps k dimensions which carry most of the data and discards dimensions with less information. It uses SVD and keeps k singular vector with bigger singular values.

We want to construct a feed forward CNN by Saab transform. Here is the training process:

In every convolution layer of Saab transform, first we divide the input M*M*D (in the first layer D is 1 of 3 depending on the input image dimension) into overlapping or non-overlapping patches depending on a given stride and a given size w *w* D and then flatten each patch. After this step we must compute and reduce the dc term from each vector and then apply PCA to reduce its dimension to a given lower dimension k. The DC anchor vector is $\mathbf{a}_0 = \frac{1}{\sqrt{N}}(1, \cdots, 1)^T$.

Then we ca apply max pooling. As the output of each node should be positive to avoid sign confusion problem so we add a bias term to each result. We set two constraints for the bias:

1-it needs to be equal for each output in a layer of Saab.

2-it should be big enough to make all the output of the node positive.

To find such a bias term we choose absolute value of the biggest negative output or a bigger value and add this term to all outputs of the layer.

Now we have a k+1 feature vector for each patch, then we reconstruct a cuboid with spectral dimension of k+1 and spatial dimension of (M/w) * (M/w) (in the case of non-overlapping patches)

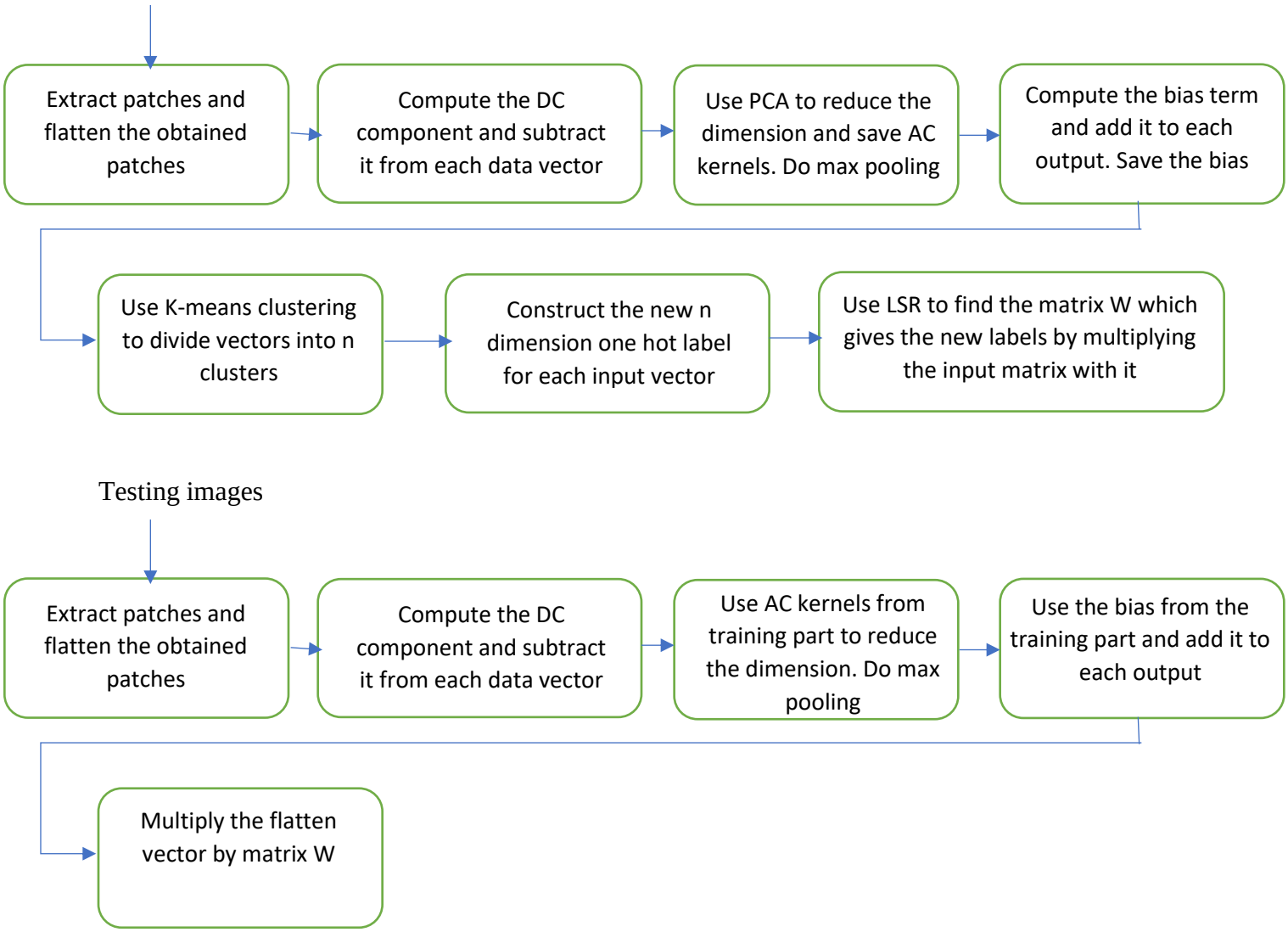
In every fully connected layer of Saab transform with input dimension of N*m and output dimension of N*n we use k means algorithm to generate n clusters out of m and then assign a n dimension one hot vector as the label of each cluster. We can determine the corresponding initial cluster of each output cluster by finding to which initial cluster the max vectors which fall in the new cluster belong. Then we use LSR to compute the weight matrix W which we need to

multiply by the input matrix of dimension N*m to generate the matrix of labels with dimension N*n.

In the testing we use the dc and ac anchor vectors of PCA of each layer obtained in the training step to project the data into a lower dimension in each conv layer. In each FC layer we just multiply the flatten data by the obtained matrix in the training process W.

Results:

1)The training and testing flow chart of a FF-CNN with one convolution layer and one FC layer. (symbols are used according my explanations in the approach part):



The first row in each diagram is the conv layer of FFCNN and the second row represents FC layer. I have explained Saab in the procedure part.

Saab and CNN are both data driven learning tools. CNN uses an end to end framework to do both feature extraction and classification while Saab transform does an unsupervised feature extraction and after that do the classification part using some existing classification methods like K-means clustering and SVM. CNN is like a black box and every thong is done by back propagation while Saab blocks are completely explainable and its is easy to see what each output is generating. we are not able to break CNN into smaller modules while we can easily separate Saab into modules with explainable functions. According to some studies Saab is also more robust to noise. Fundamentally CNN solve a non-convex optimization problem and after convergence fall into a local minimum which is not very different from absolute min while Saab uses PCA and k means which linear algebra and statistics tools to do the classification.

2)

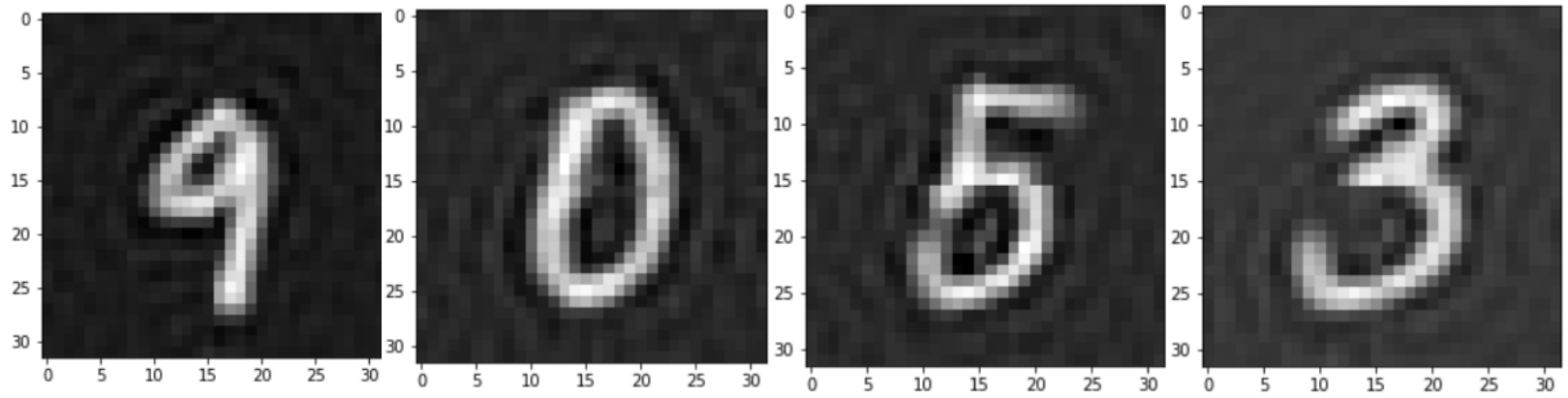


Fig1: reconstructed images after applying Saab with first and second kernel number of 10 and 60 (PSNR=24.26, 21.98, 21.8,23.38 from left to right)

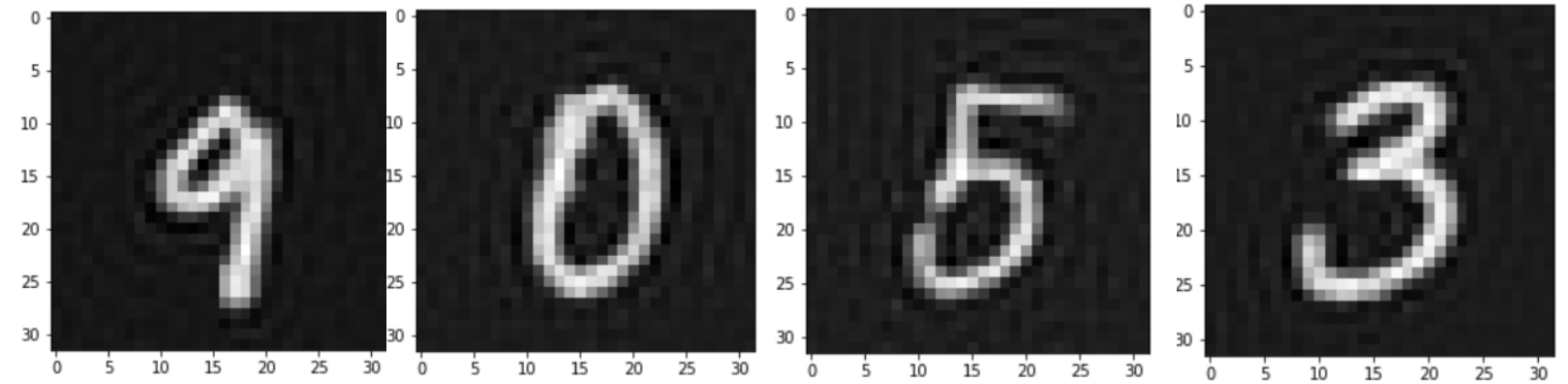


Fig 2: reconstructed images after applying Saab with first and second kernel number of 10 and 100 (PSNR=28.24, 25.64, 25.60, 27.19 from left to right)

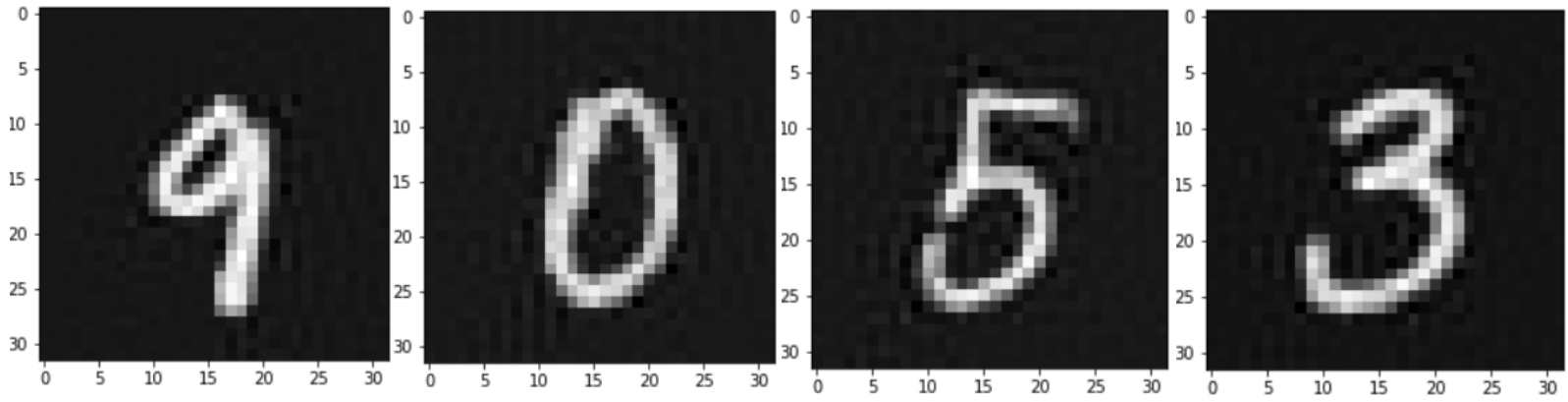


Fig 3: reconstructed images after applying Saab with first and second kernel number of 10 and 140 (PSNR=31.55, 28.43, 28.31, 28.80 from left to right)

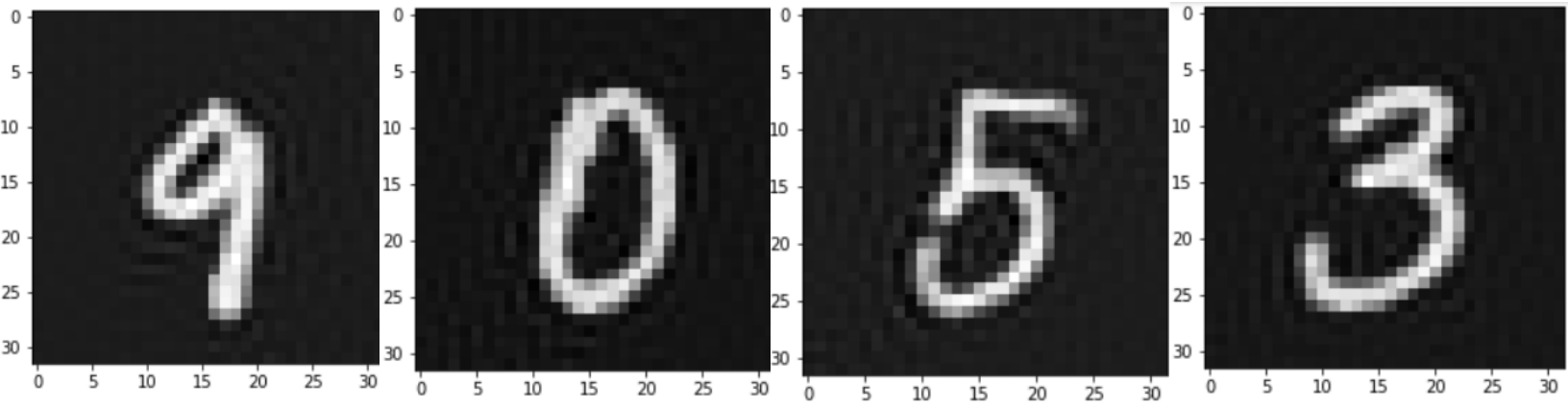


Fig 4: reconstructed images after applying Saab with first and second kernel number of 12 and 150 (PSNR=32.26, 30.31, 29.77, 30.42 from left to right)

As you see by increasing the kernel number in both first and second stage PSNR increases. The reason is by increasing the kernel number we increase the number of ac anchor vectors (remaining PCA components) and the output has more information of the input, so we can reconstruct the input with less information loss.

3) training accuracy is 0.9704 and testing accuracy is 0.9638

For the ensemble of networks, I selected the below 10 setting:

(num kernels =6,16 and stride =1 is common between all setting)

1-kernel sizes=5,5 and input = training images

2= kernel sizes=5,3 and input = training images

3- kernel sizes=3,5 and input = training images

4- kernel sizes=3,3 and input = training images

5- kernel sizes=5,5 and input = the result of applying $E3L3^T$ to training images

6- kernel sizes=5,3 and input = the result of applying $E3L3^T$ to training images

7- kernel sizes=3,5 and input = the result of applying $E3L3^T$ to training images

8- kernel sizes=3,3 and input = the result of applying $E3L3^T$ to training images

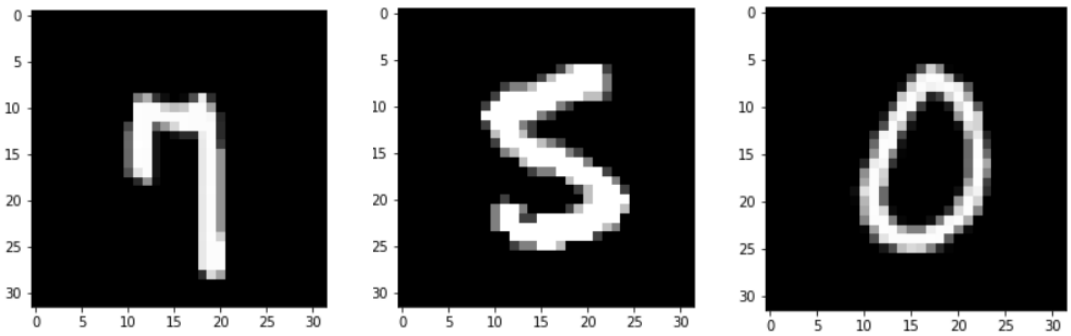
9- kernel sizes=5,5 and input = the result of applying $L3E3^T$ to training images

10- kernel sizes=5,3 and input = the result of applying $L3E3^T$ to training images

Then I use PCA to decrease the dimension of features from 100 to 60 and then applied SYM classifier with default setting to predict the labels. The final training accuracy was **0.9825** and the final testing accuracy was **0.9832**. The reason why it gives a better performance is that by using different variation of inputs data and kernel sizes we let the network compute the label using different setting and although some of them may fail to predict the output correctly, we can still predict the correct label using other architectures and representations of input data.

The best training and testing accuracy in BPCNN are 0.9847 and 0.9913 respectively, so the accuracy is higher is BPCNN in comparison with FFCNN.

About 31.52 percent of error cases of FFCNN and 62.69 percent of error cases of BPCNN are common error between these methods. The reason is that this two methods are fundamentally different so we don't expect one of them to certainly fail for a case when the other fails for that case but we can expect to see some common error cases as both of these methods are data driven and if a test data is very different from the training data we expect both of them to fail with a high probability. Here are some cases which makes both methods fail:



One way to improve FF-CNN is to define a deeper CNN architecture, as more parameters can fit better to training data and yields a better testing accuracy. We may also try different optimizers and change their parameters and train the model using all these cases to reach a better accuracy. Another method is data augmentation for example we can use different representation of data by using laws filter to increase the data variation which is used to train the model.

To improve the ensemble of networks we can try different representation of input with different kernel sizes for example we can also use edge detection techniques to create a new representation of inputs. Then we can use choose 10 networks with highest training accuracy and train the SVM by their output.

Discussion:

Saak transform is the previous version of Saab with a main difference: it uses sign to position transformation to make the output of a layer positive, while Saab uses bias: a fixed value in each layer.

The method of Saak needs the dimension to be doubled in each layer for this transformation. It is a disadvantage as the after several layers of saak the dimension of the whole data dets large while saab maintains the information without enlarging the dimension and by saving only a fixed value in each layer.