EE569 Project 6

Name: Shreyas Dinesh Patil
Email: shreyasp@usc.edu

Feedforward CNN Design and Its Application to the MNIST Dataset

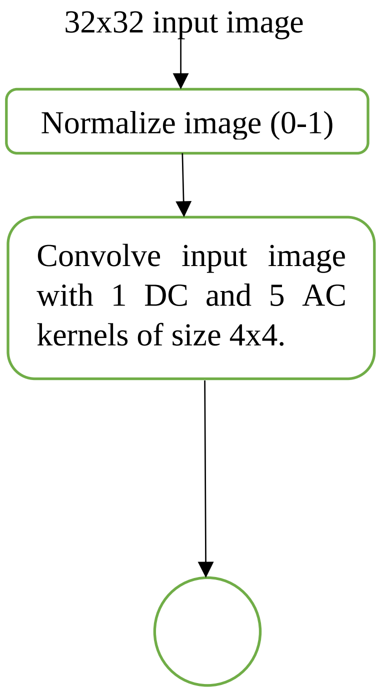
Problem 1: Understanding of feedforward-designed convolutional neural networks (FF-CNNs)

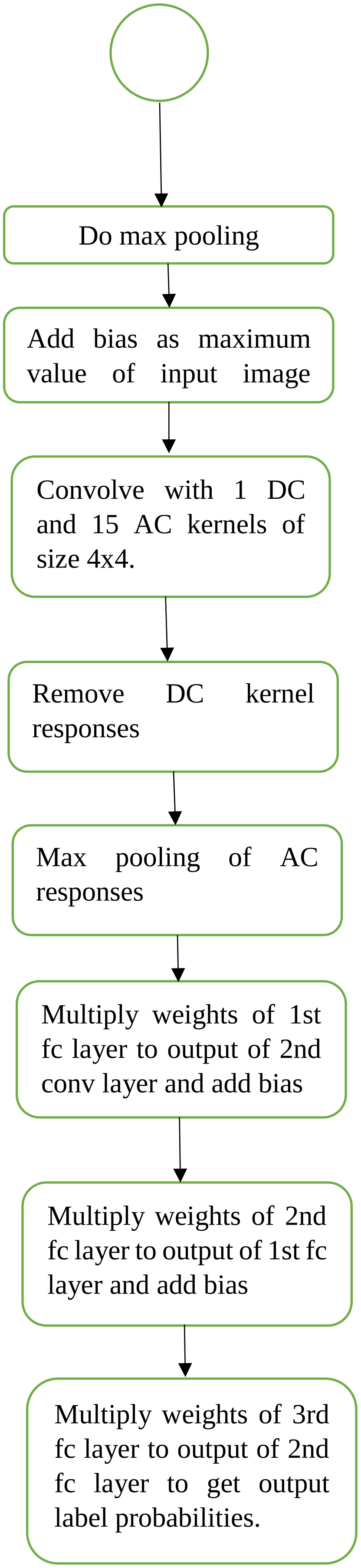
I. Motivation and Abstract

In convolutional neural networks the model parameters are calculated using non-convex optimization method along with backpropagation technique to calculate gradients. The end-to-end interpretability of CNN is a challenge. In this problem interpretable feedforward CNN developed by Dr.C.-C. Jay Kuo has been explained. In this approach the model parameters of current layer are calculated from data statistics of output of previous layer in one-pass. New signal transform (Subspace Approximation with Adjusted Bias) has been developed to construct layers of CNN. Saab is a variant of PCA. Fully connected part of CNN is implemented using cascade of multiple least squared regressors.

II. Approach:

Flowchart (Testing Phase)


32x32 input image
Normalize image (0-1)
Convolve input image with 1 DC and 5 AC kernels of size 4x4.


Do max pooling
Add bias as maximum value of input image
Convolve with 1 DC and 15 AC kernels of size 4x4.
Remove DC kernel responses
Max pooling of AC responses
Multiply weights of 1st fc layer to output of 2nd conv layer and add bias
Multiply weights of 2nd fc layer to output of 1st fc layer and add bias
Multiply weights of 3rd fc layer to output of 2nd fc layer to get output label probabilities.

Training phase (Convolution layers) – Learning the kernels

For each conv layer

- For each input image to conv layer, kernel size non-overlapping or overlapping patches are created.
- Feature mean is subtracted (for input feature normalization) from each feature for all patches. Patch mean is subtracted from each flattened patch.
- PCA is applied on input data matrix (number of patches for all training images vs flattened size of patches) to get AC kernels.
- DC kernel is found as $\frac{1}{\sqrt{size\ of\ kernel}} * (1, 1, 1, \ldots\ldots, 1)^T$
- DC kernel is augmented to AC kernels to get kernels matrix.
- Input for next layer (Saab coefficients) is obtained by multiplying input data matrix and kernels matrix.
- Add bias which is the maximum L2 norm of patches, to the result obtained in previous step.
- Applying max pooling operation to result from previous step to get image for next iteration.

Training phase (Fully connected layers)

For each fully connected layer

- Input to the current layer (for 1st layer features from last conv layer) is clustered into specific number of clusters (number of output nodes) for each class using k-means clustering.
- Then K pseudolabels are created by combining the class and cluster labels.
- Using linear least squared regression method weights are found for current layer.

Explanation for both training and testing:

1) Conv layers provide sequence of spatial-spectral filtering. In this process the spatial resolutions become gradually low. To compensate this spatial loss spectral components are increased.
2) Overlapping stride of filters provides a richer set of features for selection.
3) Patch mean is subtracted from each patch because it is a condition required before applying principal component analysis to input data matrix (number of patches for all training images vs flattened size of patches).
4) The local spatial-spectral cuboids are projected on PCA-based kernels. This is done do enhance discriminability of some dimensions and dimension reduction.

5) Adding bias which is maximum L2 norm of patches acts as a substitute for ReLu function. Adding such a bias at the output of each conv layer ensures the output for each conv layer is positive. It removes the sign confusion problem.
6) Max pooling helps in getting the most important features spatially and also dimension reduction.
7) Multistage cascade of conv layers is used as they can generate rich set of image features. As we move deep into conv layers rich image patterns are detected.
8) The output of each fc layer is one hot vector. Cluster labels for hidden layers are generated using k means clustering on the input to the respective hidden layers. K is the number of output nodes for present input layer. This new cluster labels are combined with original labels to create K pseudo labels in the present layer. These pseudo sub classes accommodates intra-class variability. Linear least squared regressor (weights and biases) is found by setting linear equations relating input to the output. For next fc layer same similar procedure is repeated with pseudo labels of previous layer as features for the next fc layer.
9) Non linear activation ReLu is applied to the output of each hidden output.

Similarities between FF-CNN and BP-CNN:

- Both approaches are data centric/data driven. BP-CNN uses labeled dataset to train its network with optimization of output end cost function. FF-CNN uses covariance matrix statistic to decide the sequence of spatial spectral filtering in conv layers. The purpose of this is to extract discriminant features and dimension reduction.
- The fully connected layer part of both approaches makes use of labels for finding its weights and biases. In BP-CNN gradient descent procedures using back propagation is used for correcting errors in output and adjust weights and biases. FF-CNN makes use of labeled data in clustering data for generating pseudo labels and also for output layer with original labels in fully connected layers.

Difference between FF-CNN and BP-CNN:

- BP-CNN is end-to-end optimization while FF-CNN is divided into 2 cascaded stages each handled separately.
- Training BP-CNN is slow compared to FF-CNN. BP-CNN is iterative optimization procedure. For each epoch entire training dataset is used and this is done for many epochs. Thus it requires more training time.
- BP-CNN training is completely dependent on training samples labels. FF-CNN uses labels only for fully connected layers, the parameters of convolutional layers are independent of training sample labels. Only statistics of training data is used for finding kernels in conv layers.
- FF-CNN are mathematically more transparent than BP-CNN. In BP-CNN full explanation of network is very difficult. FF-CNN can explain the significance of

ReLu activation, max pooling and multistage cascading transparently using mathematical tools.

- BP-CNN has lots of parameters in comparison to FF-CNN.
- Extracted features in FF-CNN are less discriminant than those of BP-CNN.

Problem 2: Image reconstructions from Saab coefficients

I. Approach:

The 4 sample images are input to the conv part of the network to generate saab coefficients after each layer for them. These saab coefficients are then used to reconstruct the respective images. The reconstructed results are shown for four different settings (different number of kernels in each stage). PSNR score is used to evaluate the quality of reconstructed images.

Architecture setting:

There are 2 convolution layers. The kernel size for each conv layer is 4x4. The stride for each conv layer is 4 (non-overlapping). There are 6 (5 AC, 1 DC) and 16 (15 AC, 1 DC) kernels in 1st conv and 2nd conv layers respectively. No max pooling operation is used in each conv layer.

Procedure: Generating saab coefficients

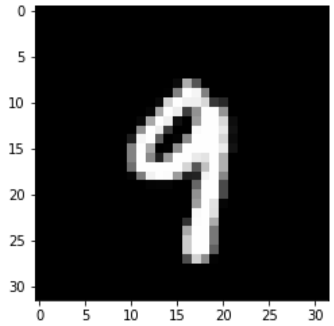
1) Train conv layers as shown in Part 1 to get kernels in each conv layer using 10000 train images.
2) Create 4x4 patches for given images of size 32x32 to generate input data matrix.
3) Subtract patch mean from all patches of the given input images.
4) Multiply input data matrix to 1st conv layer kernels to get saab coefficients for 1st conv layer.
5) Generate 4x4 patches for 2nd layer input from 4). Subtract patch mean from all patches.
6) Add bias term to 5).
7) Multiply 6) to kernels in 2nd conv layer.
8) Subtract bias from 7) to get saab coefficients of 2nd conv layer.

Procedure: Reconstruction algorithm

1) Add bias of 2nd conv layer to saab coefficients of 2nd conv layer.
2) Multiply result of 1) to inverse of kernels transpose to get image patches.
3) Add feature mean and patch mean to result of 2)
4) Reverse the window process applied to image patches to get saab coefficents of 1st conv layer.
5) Repeat above steps until you get patches of input image.
6) Merge the input patches from 5) to get reconstructed image.
7) Calculate PSNR values for given 4 images under different parameter settings of conv layers.

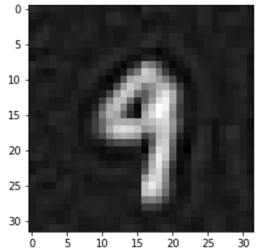
II. Experimental results

Setting 1: Conv1= 12 AC filters, Conv2 = 40 AC filters
Setting 2: Conv1= 12 AC filters, Conv2 = 100 AC filters
Setting 3: Conv1= 15 AC filters, Conv2 = 100 AC filters
Setting 4: Conv1= 10 AC filters, Conv2 = 100 AC filters

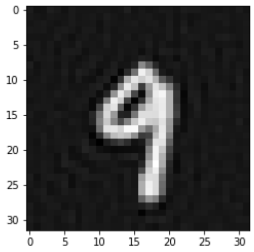


Original image

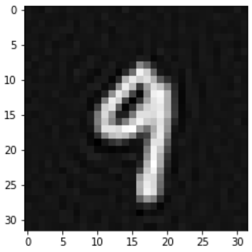
Setting 1



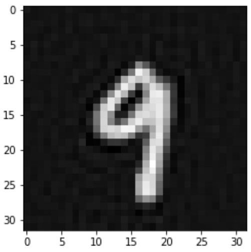
Setting 2

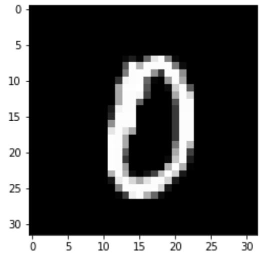


Setting 3



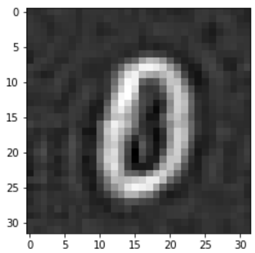
Setting 4

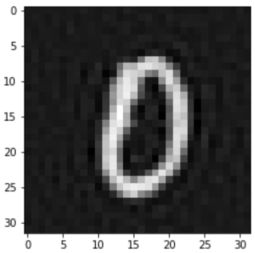


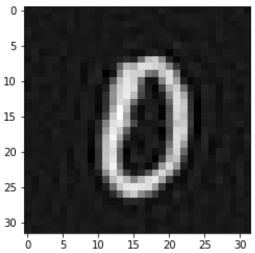


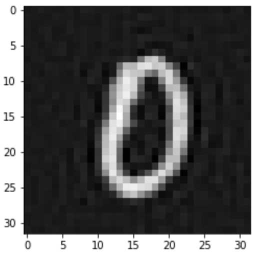
Original image

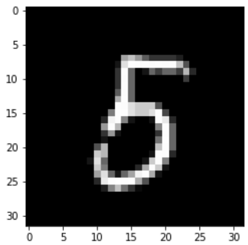
Setting 1 Setting 2 Setting 3 Setting 4





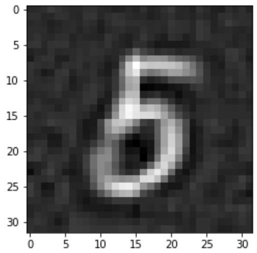




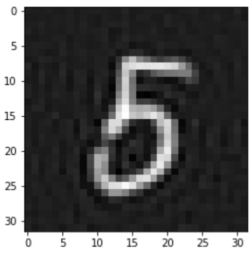


Original image

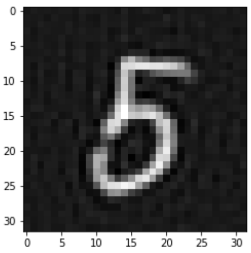
Setting 1



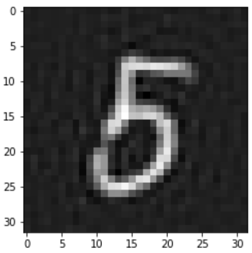
Setting 2

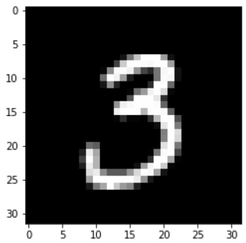


Setting 3



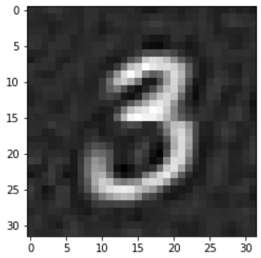
Setting 4



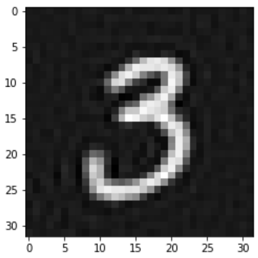


Original image

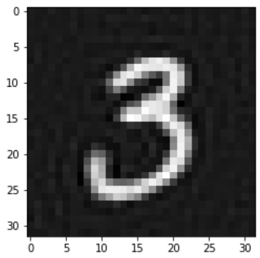
Setting 1



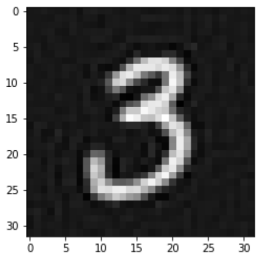
Setting 2



Setting 3



Setting 4



PSNR values:

	Zero image PSNR	Three image PSNR	Five image PSNR	Nine image PSNR
Setting 1	20.184959248232644	21.77279628582991	20.023856844794	22.15754418760983
Setting 2	26.073856131387345	27.925640537458122	26.277355862870202	28.4220506954776
Setting 3	26.10327710072024	28.098705633017612	26.418599257164242	28.55754762031724
Setting 4	25.69096042610623	27.332439434583353	25.699819454274348	28.405128892943065

III. Discussion

- As we can see that the inverse saab transform are not the same as original image. Thus there is loss involved in saab transform. This is because saab transform is a variant of PCA which is a lossy dimension reduction technique.
- As we can see as the number of kernels increases in 2nd conv layer keeping the number of 1st layer conv layer constant the reconstructed images become better. This can be evaluated numerically using PSNR values of reconstructed images.
- From results we can see that keeping number of filters constant in 1st conv layer and increasing number of filters in 2nd conv layer gives better reconstruction. Similarly, keeping number of filters in 2nd conv layer constant and increasing number of filters in 1st conv layer gives better reconstruction results.
- Convolution layers do spatial spectral filtering which causes the spatial resolutions to become coarser along the process.
- There are 2 types of losses introduced due to subspace approximation. Approximation loss and rectification loss. Rectification loss is removed by bias.
- If the number of anchor vectors is less than the dimension of input space there is approximation error. Therefore as the number of anchor vectors increases reconstruction result and PSNR values get better.
- Increase in the number of kernels in end layers of conv network causes more features to be extracted which in turn increases the number of saab coefficients generated. So, reconstructed images with more saab coefficients is more close to original images.

Problem 3: Handwritten digits recognition using ensembles of feedforward design

I. Abstract and Motivation:

To improve the performance of FF-CNN ensemble method is used. In this method the output of multiple FF-CNN is fused to improve accuracy of image classification problem. To ensemble the results of multiple FF-CNN it is important to increase the diversity of individual models. Here the strategies of using different parameter settings in the conv layers and flexible input image forms are used to increase the diversity of FF-CNN models.

II. Approach:

Architecture setting for individual one FF-CNN:
LeNet-5 architecture is used – Kernel size for each conv layer is 5x5 with stride 1. 6 and 16 filters are used in 1st and 2nd conv layers respectively. The number of nodes in 1st and 2nd fc layers are 120 and 80 respectively.

Procedure: (for filter sizes 5x5, 5x3, 3x5, 3x3)

1) For each diverse 4 FF-CNN model
   i) Train model with 60000 mnist train images.
2) Concatenate the output predicted vectors of all 10 FF-CNN.
3) Apply PCA to output from 2)
4) Train any multiclass classifier with input from 3) and labels as train labels.
5) Calculate training accuracy.
6) Calculate testing accuracy for 10000 test data.

Procedure: (for Laws filter settings)

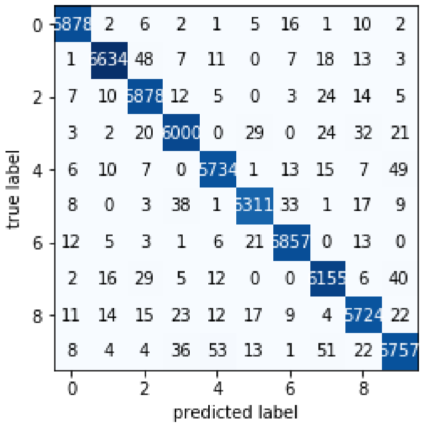
1) For each diverse 6 FF-CNN model
   i) Apply Laws filter to 10000 training images.
   ii) Train the conv part of network with result from 1) to learn kernels.
   iii) Train conv part of network with 60000 train images to generate features.
   iv) Train fc part of network with output from 3).
2) Concatenate the output predicted vectors of all 10 FF-CNN.
3) Apply PCA to output from 2)
4) Train any multiclass classifier with input from 3) and labels as train labels.
5) Calculate training accuracy.
6) Calculate testing accuracy for 10000 test data.

III. Experimental results

5x5 Laws filters used for 6 different settings.

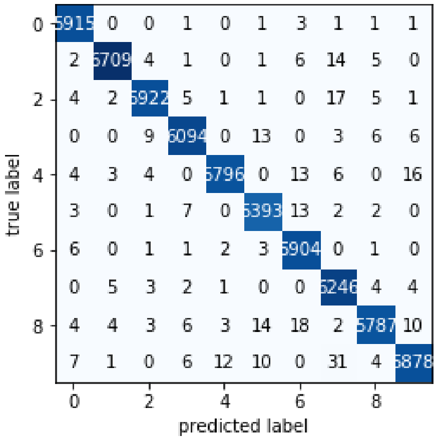
Settings	Filter Size (5, 5)	Filter Size (5, 3)	Filter Size (3, 5)	Filter Size (3, 3)	Laws filter E5E5	Laws filter E5S5	Laws filter E5W5	Laws filter S5E5	Laws filter S5S5	Laws filter S5W5	Ensemble
Train accuracy	97	97.11	97.08	97.10	97.05	97.20	97.01	97.07	97.06	96.91	98.21
Test accuracy	97.05	97.24	97.07	96.66	97	97.11	96.94	97.03	96.94	96.79	98.15

Confusion matrices for BP-CNN and FF-CNN



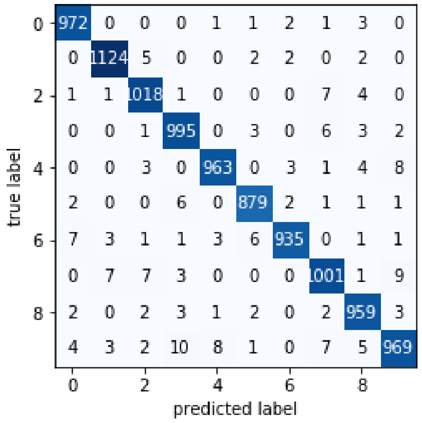
5878	2	6	2	1	5	16	1	10	2
1	6634	48	7	11	0	7	18	13	3
7	10	5878	12	5	0	3	24	14	5
3	2	20	6000	0	29	0	24	32	21
6	10	7	0	5734	1	13	15	7	49
8	0	3	38	1	5311	33	1	17	9
12	5	3	1	6	21	5857	0	13	0
2	16	29	5	12	0	0	6155	6	40
11	14	15	23	12	17	9	4	5724	22
8	4	4	36	53	13	1	51	22	5757

Fig1: Confusion matrix of FF-CNN for train data



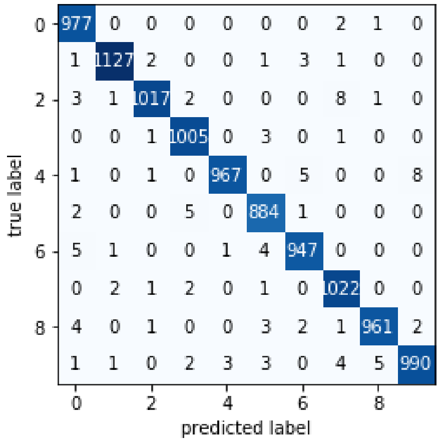
5915	0	0	1	0	1	3	1	1	1
2	6709	4	1	0	1	6	14	5	0
4	2	5922	5	1	1	0	17	5	1
0	0	9	6094	0	13	0	3	6	6
4	3	4	0	5796	0	13	6	0	16
3	0	1	7	0	5393	13	2	2	0
6	0	1	1	2	3	5904	0	1	0
0	5	3	2	1	0	0	6246	4	4
4	4	3	6	3	14	18	2	5787	10
7	1	0	6	12	10	0	31	4	5878

Fig2: Confusion matrix of BP-CNN for train data



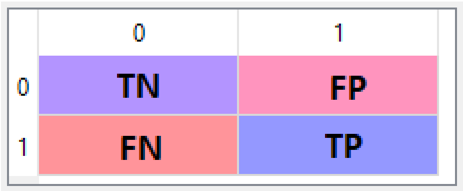
972	0	0	0	1	1	2	1	3	0
0	1124	5	0	0	2	2	0	2	0
1	1	1018	1	0	0	0	7	4	0
0	0	1	995	0	3	0	6	3	2
0	0	3	0	963	0	3	1	4	8
2	0	0	6	0	879	2	1	1	1
7	3	1	1	3	6	935	0	1	1
0	7	7	3	0	0	0	1001	1	9
2	0	2	3	1	2	0	2	959	3
4	3	2	10	8	1	0	7	5	969

Fig3: Confusion matrix of FF-CNN for test data



	0		2		4		6		8	
0	977	0	0	0	0	0	0	2	1	0
	1	1127	2	0	0	1	3	1	0	0
2	3	1	1017	2	0	0	0	8	1	0
	0	0	1	1005	0	3	0	1	0	0
4	1	0	1	0	967	0	5	0	0	8
	2	0	0	5	0	884	1	0	0	0
6	5	1	0	0	1	4	947	0	0	0
	0	2	1	2	0	1	0	1022	0	0
8	4	0	1	0	0	3	2	1	961	2
	1	1	0	2	3	3	0	4	5	990

Fig4: Confusion matrix of BP-CNN for test data



	0	1
0	TN	FP
1	FN	TP

Confusion matrix for binary classification (for reference)

IV. Discussion

- Strategy1 to generate 4 different settings of FF-CNN:
  I created 4 FF-CNNs with different filter sizes in each. i) (5x5, 5x5), ii) (5x3, 5x3), iii) (3x5, 3x5), iv) (3x3, 3x3). The number of filters in each conv layer is 6 and 16 respectively. These are diverse representative settings required for ensembling because different filter sizes result in different receptive field sizes in FF-CNN. This results in different statistics in input data. Therefore they generate different features at conv part output.
- Strategy 2 to generate 6 different settings of FF-CNNs: I adopted different image input forms to increase diversity in ensemble system.
  I applied 6 different 5x5 Laws filters to 10000 training images (less due to memory overflow issue). This creates 6 different sets of Laws filtered images containing frequency components in different subbands. These subbands provide important features in the input which helps the network select important features from the input more efficiently using saab transform.

Error Analysis:

- Error analysis is done using confusion matrix for both FF-CNN and BP-CNN for test and train images.
- From diagonal values of confusion matrix we can see there are high amount of correctly classified images by both FF-CNN and BP-CNN.

- Values not along the diagonal of confusion matrix are incorrectly classified images. They are almost the same for both FF-CNN and BP-CNN.
- From confusion matrix the percentage of error for both BP-CNN and FF-CNN is around 2% and therefore they are almost same. Thus BP-CNN and FF-CNN perform really good.
- If we consider label 1 the misclassification error percentage for BP-CNN is 0.7% and FF-CNN is 0.96% which is almost same deduced from above confusion matrices for test data. Similar result can be obtained for other classes as the mnist dataset is balanced.

Ideas for improving BP-CNNs and FF-CNNs

- BP-CNNs performance improvement:
  - i) We can improve performance of BP-CNNs by providing more labeled data through data augmentation techniques.
  - ii) By trying different training algorithms and select algorithm that performs best on the given dataset.
  - iii) Increasing the number of conv layers and filter size gradually might help to improve performance.
- Ideas to improve performance of FF-CNNs:
  - i) Applying channel wise PCA method explained in [2] we can improve the performance of FF-CNNs.
  - ii) Data set partitioning: In this procedures hard samples are separated from easy ones based on either the final decision vector of the ensemble classifier or the prediction results of all base classifiers.
  - iii) Semi supervised using FF-CNN-In this system backpropagation is not needed to derive network parameters. Outputs of multiple semi-supervised FF-CNNs can be combined to boost classification accuracy.
  - iv) We can use Backpropagation for fc layers to boost accuracy.

References:

1) Kuo, C. C. J., Zhang, M., Li, S., Duan, J., & Chen, Y. (2019). Interpretable convolutional neural networks via feedforward design. Journal of Visual Communication and Image Representation.
2) Chen, Y., Yang, Y., Wang, W., & Kuo, C. C. J. (2019). Ensembles of feedforward-designed convolutional neural networks. arXiv preprint arXiv:1901.02154.
3) [MNIST] http://yann.lecun.com/exdb/mnist/
4) https://github.com/davidsonic/Interpretable_CNNs_via_Feedforward_Design