Chinmay Joshi

Chinmaj

ECE8720

#TakeHome 2

Balancing the Dataset

We have 83 AEP class vectors and 2482 non-AEP class vectors. Clearly the Dataset is imbalanced.

Final Dataset Used:

240 AEP-vectors, 240 non-AEP vectors, total **480** vectors

Methods Used to Balance the dataset (Training Dataset)

1. **SMOTE(Synthetic Minority Over-sampling Technique)**
    a. Taking 3 more synthetic samples per original sample in minority class(3*60)
    b. 3 Different neighbors are used to generate 3 different points
    c. Total class of 240 is created
    d. Taking more synthetic samples may lead to over-generalization if samples are far from original space, or overtraining if samples are close to one another.
2. **K-nearest neighbors Cluster** Classification:
    a. Down-sampling majority class based on clustering
    b. 20 different clusters are used and 12 vectors from each class taken
    c. Taking less samples may lead to loss of information
3. Combined data is used or Training purpose

Test Data-Set:

1. All remaining Vectors from above process taken into Test Data-set.
2. **23 AEP-Class vectors, 2242 non-AEP Class**
3. Ignoring some vectors from available dataset can be misleading. Hence, All dataset is used either for Training or for Testing purpose.

Case #1: Base Case (Without Bias)

EPOCH Iterations = 10000

LRATE= 0.00009/SIZE_TRAIN;

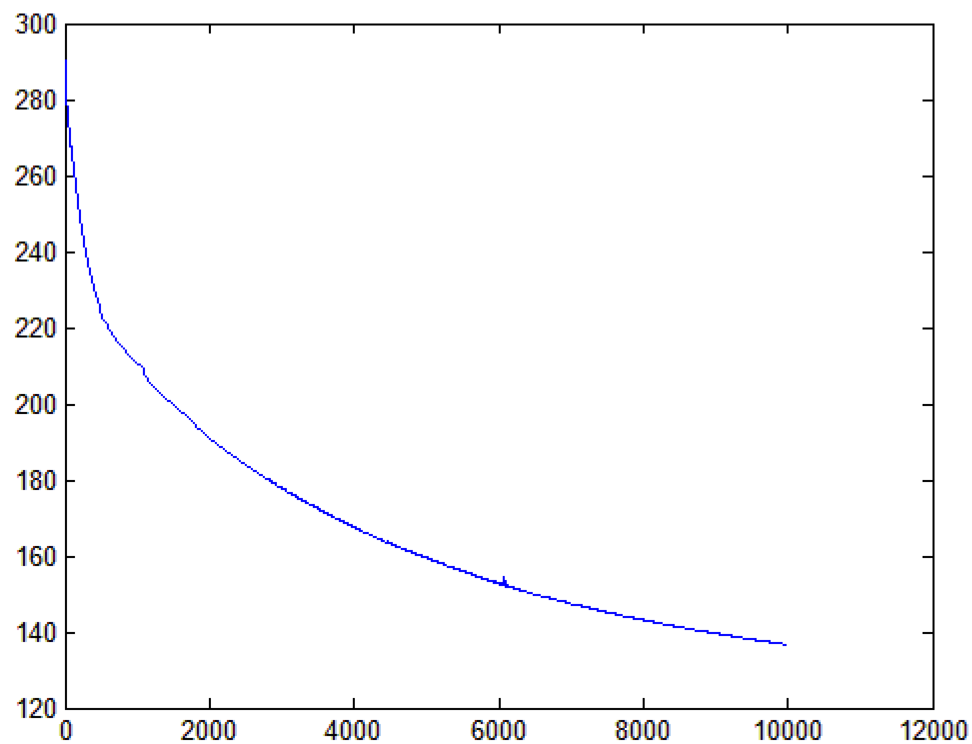
ETSS=737.0998

Specificity= TN/(TN+FP)= 0.6713

Sensitivity= TP/(TP+FN) = 0.9130

Learning Rate: Learning Rate is chosen in every case so that:

1. We will have minimal oscillations in E per Iterations graphs
2. Training converges fast enough for good Sensitivity-Specificity Values
3. Depends on Size of training dataset, as we add deltas together. Hence SIZE_TRAIN is used in LRATE equation to make algorithm dynamic for any training dataset
4. Learning Rate selected here is 0.0009/SIZE_TRAIN. This allows smooth variations in the weight space
5. Random Weight Initialization: Weights are selected such that no output should stuck at saturation, at least in initial iterations .



CASE # 2:   (With Bias)

ETSS= 836.9497

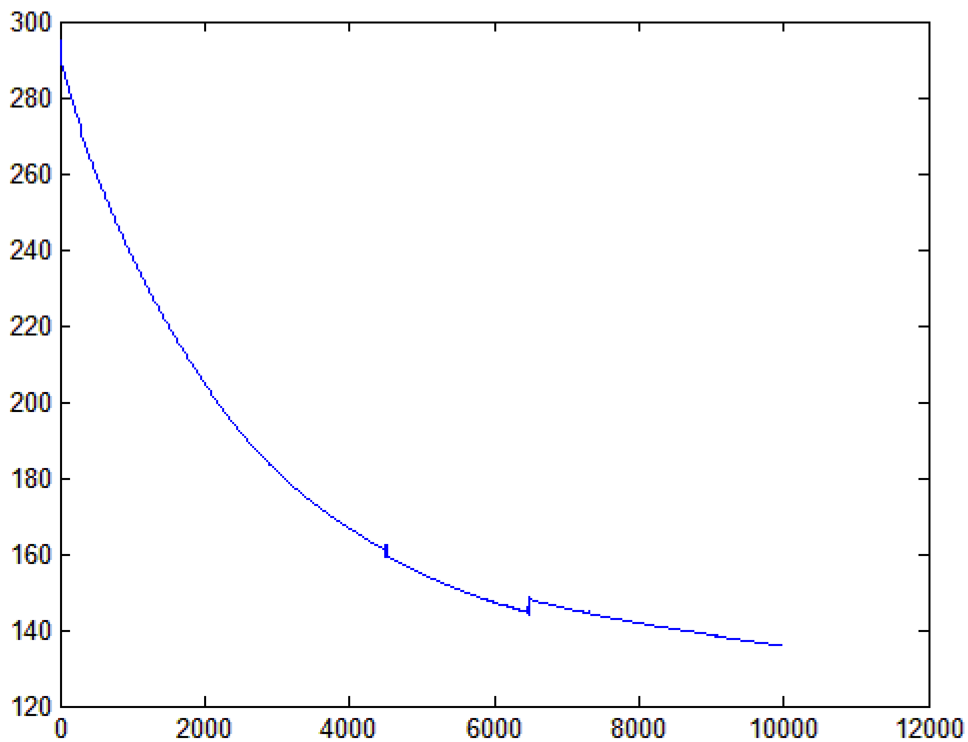
Sensitivity=0.9130

Specificity=0.6713

Iterations= 10000

Learning Rate= 0.00009/SIZE_TRAIN

Using Bias allows quicker convergence for good results.



CASE # 3: ROM Weight Initialization

ETSS= 1051.8

Sensitivity= 0.9130

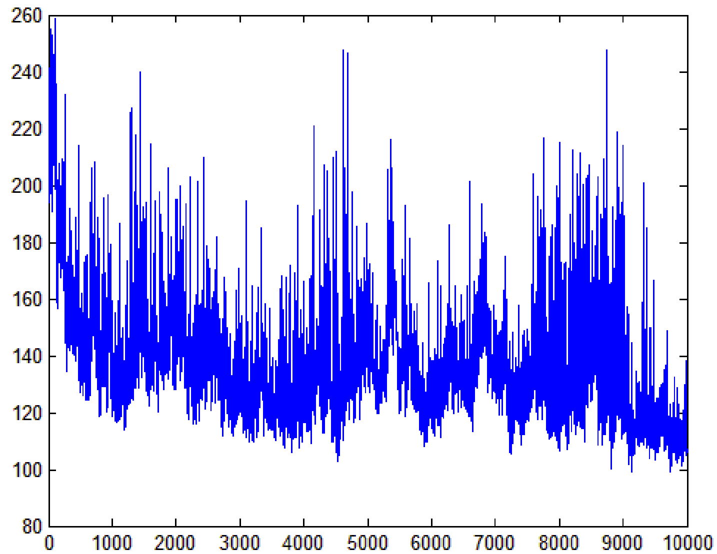
Specificity= 0.6481

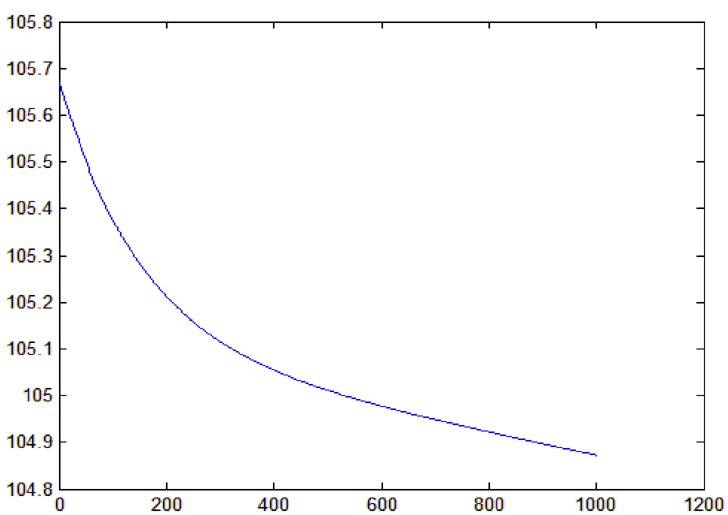
ROM Iterations = 10000

Gradient Descent iterations = 1000;

LRATE= 0.0002/SIZE_TRAIN

Figures: 1. ETSS vs ROM Iterations   2.ETSS vs GDR





Case#4: Momentum

Common Parameters:

Weights_initial: from ROM

Iterations=15000

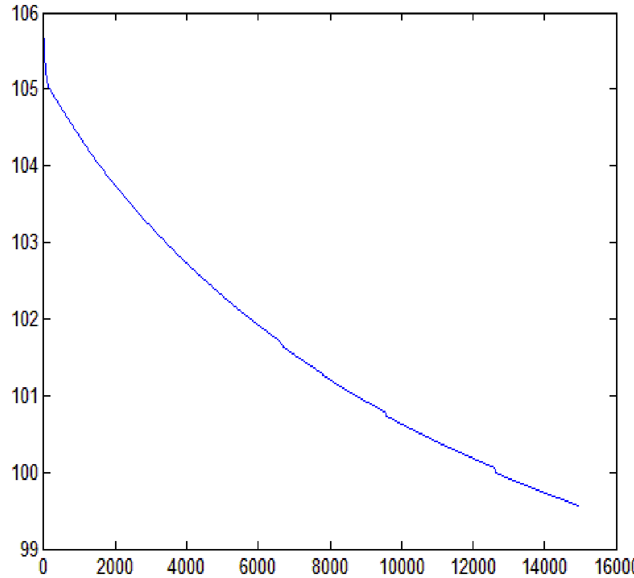
LRATE=0.0006/SIZE_TRAIN

Without momentum:

Sensitivity= 0.9130

Specificity= 0.5607

ETSS = 1240;



With momentum

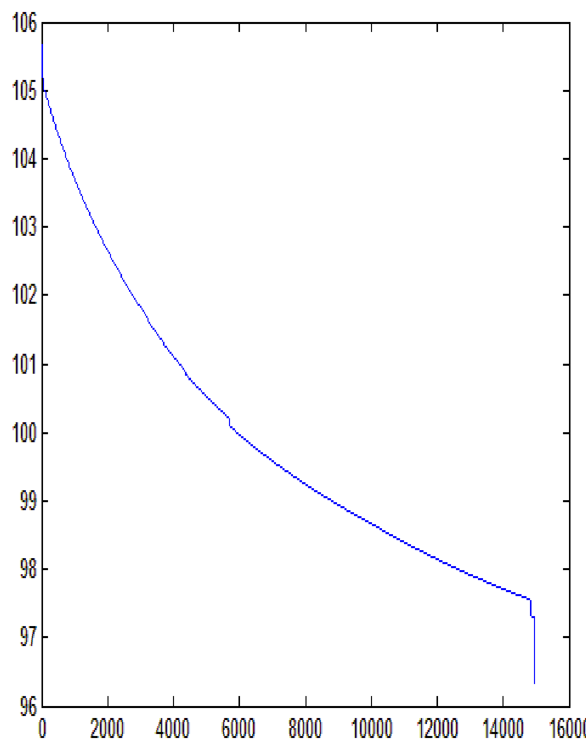
ETSS 1215.8

Sensitivity= 0.9130

Specificity= 0.5682

Alpha for Momentum = 0.52

As we can see, momentum allows weights to converge faster at local minima



Case # 5: Weight Decay

Common Parameters:

Weight Initialization : From ROM

LRATE = 0.00005/SIZE_TRAIN

Epoch Iterations = 10000

Without Weight decay

ETSS= 1270.05

Sensitivity= 0.9130

Specificity= 0.5522

WITH Weight Decay:

WEIGHTS_initial: ROM

With Weight Decay :

Beta=0.99999

Sensitivity= 0.9130

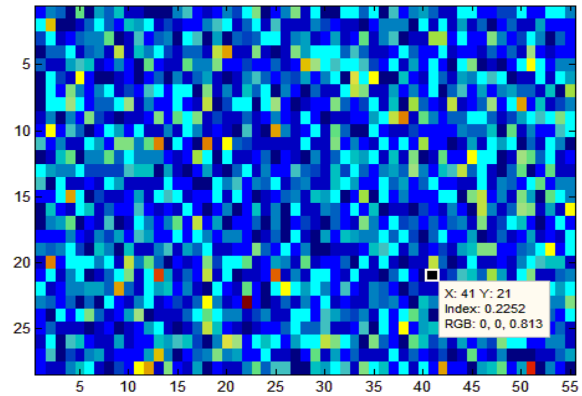
Specificity = 5526

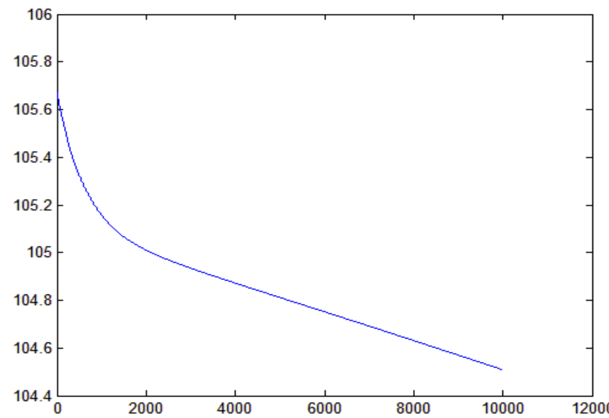
ETSS= 1247.6

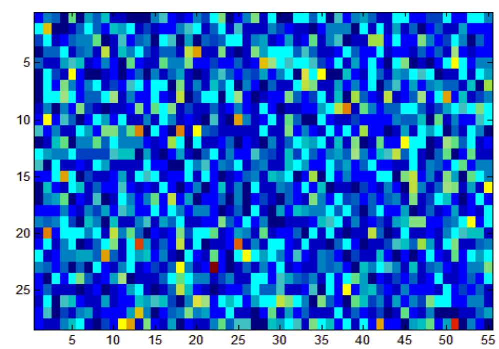
Figures:

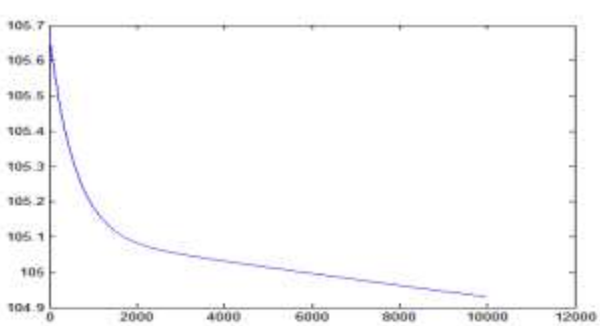
1,3: Imagesc op of absolute weights

2,4: ETSS vs iterations





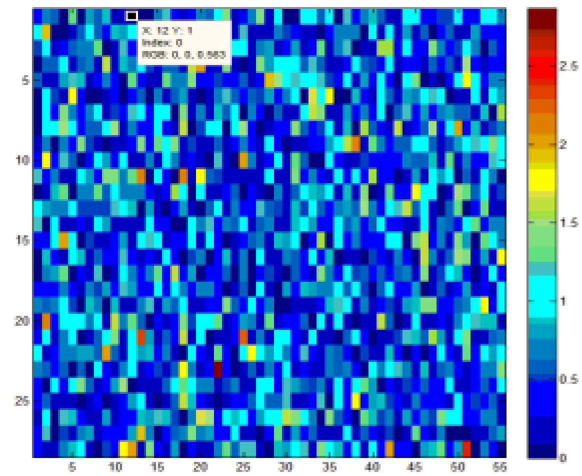




Case 6 Karnin:

Method of pruning: By 5 percent, based upon abs(S) values:

Abs(S) value gives sensitivity of the Weight wrt Error. Hence it is used to prune the network. 5 percent weights in 28*55 weights are pruned whose sensitivities were lowest.



Without pruning:

ROM weight initialization (Same as case 3)

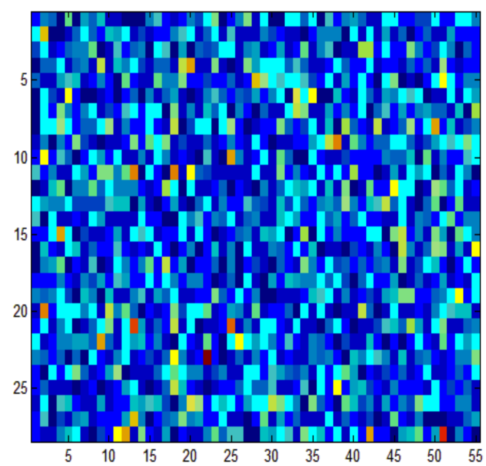
GDR Epoch iterations= 10000

Sensi 0.9130

Speci 5522

ETSS 1270.5

LRATE=0.00005/SIZE_TRAIN



Same except

Sensitivity 9565

Specificity 4349

ETSS 1588.4

Figures:

1. Abs(weights) after Pruning
2. Abs(weights) before Pruning
3. Weight Sensitivities
4. E per Iterations: retesting network