Study of Deep Neural Networks for Finger Print Spoof Detection

Burra Venkata Krishna Karthik

Electrical Engineering

University of Missouri-Kansas City, MO-64110

vbcr4@mail.umkc.edu

Declaration- I Mr. Burra Venkata Krishna Karthik, hereby declare that the paper titled 'Study of Deep Neural Networks for Finger Print Spoof Detection' submitted by me is based on actual and original work carried out by me. Any reference to material obtained from other sources have been duly cited and referenced. I further certify that the paper has not been published or submitted for publication anywhere else nor it will be send for publication in the future.

Abstract

A fingerprint spoof detector using Deep neural network construction based on Autoencoders using soft max layer is a pattern classifier that is used to distinguish a live finger from a spoof one in the subject of a fingerprint recognition system. An autoencoder neural network is an unsupervised learning algorithm that applies backpropagation, setting the target values to be equal to the inputs ([1], page 12). Data set was provided by LivDet2011 consisting of live and spoof images fabricated using materials such as Silgum, Latex, Gelatin and Ecoflex separated into training and test sets (non-overlapping set). The original images from these databases and features such as Local Binary Pattern (LBP), Binary Statistical Image Features (BSIF) and Binary Gabor Pattern (BGP) are extracted from live and fake images using two different fingerprint sensors -Digital and Sagem. Two layers for deep learning trained using auto encoders are used and different parameters were tested and the parameters having the highest Area Under Curve (AUC) was selected and then it was trained with soft max layer. Those two Autoencoders and Softmax layer (binary classifier) ([6]) were stacked and a neural network was formed. Binary label is created for live and spoof images to perform testing. This network was tested and Receiver Operation Curve (ROC) ([1], page 388) for that output of neural network was plotted and finally provided conclusions about the accuracy of the system. The ROC curve plotted is Spoof Accept Rate or False Accept Rate(FAR) VS Live Reject Rate or Genuine Reject Rate (GRR).

Introduction

Finger print can be easily fabricated using commonly available materials, such as Silgum, Latex, Gelatin and Eco flex ([3], page1). These spoof finger prints can be used by a person to launch a spoof attack by placing them on a finger print sensor and claiming identity of another person. In this project, Finger print spoof detection using deep neural networks is implemented and the data base was provided by LivDet2011. The database consists of 1000 live and 1000 fake fingerprint samples in the training set and the same number of samples in the test set ([3], page 4). Validation is done on training set for about 20%. The original images from these databases and features such as Local Binary Pattern (LBP), Binary Statistical Image Features (BSIF) and Binary Gabor Pattern

(BGP) are extracted from live and fake images using two different fingerprint sensors -Digital and Sagem.

Feature BSIF encodes texture information as a binary code for each pixel by linearly projecting local image patches onto a subspace, whose basis vectors are from natural images ([3], page 4). Feature LBP forms labels for the image pixels by thresholding the neighborhood of each pixel with the center value and considering the result as a binary number. ([3], page 4). Feature BGP, encodes texture information by convolving the image with Gabor filters and binarizing the responses ([3], page 4).

In recent years, deep artificial neural networks (including recurrent ones) have won numerous contests in pattern recognition and machine learning. This historical survey compactly summarizes relevant work, much of it from the previous millennium ([4], page1). Shallow and deep learners are distinguished by the depth of their credit assignment paths, which are chains of possibly learnable, causal links between actions and effects ([4], page1). Which modifiable components of a learning system are responsible for its success or failure? What changes to them improve performance? This has been called the fundamental credit assignment problem (Minsky, 1963) ([4], page4). There are general credit assignment methods for universal problem solvers that are time-optimal in various theoretical senses ([4], page4). One such method commercially important is Deep Learning (DL) in Artificial Neural Networks (NNs) ([4], page4). Deep learning algorithms attempt to model high-level abstractions in data by using multiple processing layers and use Graphical Processing Unit (GPU).

In this paper, in section 2, the methods used for this algorithm and the parameters used for auto encoders using deep learning are used. In section 3, the results obtained are shown. Primarily discuss about the results of receiver operation characteristics (ROC) ([2], page 388). In section 4 the conclusions are drawn on the results obtained. In section 5 the references are mentioned. Section 6 contains appendix of all the tables, figures and formulae which were used in this project.

Method

In this project, the data base LivDet2011 consists of live and spoof images fabricated using materials such as Silgum, Latex, Gelatin and Ecoflex separated into training and test sets (non-overlapping set). The original images from these databases and features such as Local Binary Pattern (LBP), Binary Statistical Image Features (BSIF) and Binary Gabor Pattern (BGP) are extracted from live and fake images using two different fingerprint sensors -Digital and Sagem. An auto encoder training neural network is unsupervised learning algorithm and uses back propagation ([2], page 149) for setting the target values to be equal to the input. ([1], page 12). For training an auto encoder we have four different parameters which can be changed to obtain an optimal result/best configuration. The four different parameters are Hidden layer sizes, L2W regularizations also known as weight decay ([1], page12), Sparsity Regularization ([5]) and Sparsity Proportion also known a Sparsity parameter ([1], page 14). Auto encoder output will try to be almost same as the input. In this project, each train data is of 1000 live and 1000 spoof images and have different features accordingly. LBP has 54 features, BGP has 216 features and BSIF has

512 features. Consider taking LBP data input which has 54 features, first hidden layer size as 400, so the corresponding output of auto encoder is 54 same as input, the network will be forced to learn a compressed version of the input ([1], page 13). Using Second Hidden Layer size of 150, the input will have the first hidden layer size which is 400 and the output will be same as input that is 400 here. Due to forcing of network for a compressed version of output, some co-relation between inputs is seen ([1], page 12). So, different hidden layer sizes were used to obtain different configurations and different ROC were plotted for different hidden layer sizes. L2W Regularization also known as weight decay ([1], page12) is another parameter which can affect the training of auto encoder. It's like weight decay it affects more for smaller weights while weight update after each iteration ([2], page 246). Imposing a Sparsity constraint on hidden layers can find interesting co-relation between inputs ([1], page 14). So, two other parameters called sparsity regularization ([5]) and sparsity proportion come into the picture. Most of the neurons will be inactive in a network to change this use sparsity proportion. ([1], page 14). The sparsity proportion is average number of activations over the sample. So, add a term called sparsity regularization to cost function ([1], page 15). Use one or two auto encoders with different hidden layer sizes, weight regularizations and different sparsity proportions to get the optimal or best configuration. After training, the auto encoder is encoded to extract the features of the input. The features extracted from the auto encoder will be used to train the soft max layer. Then create a deep neural network stacking auto encoders and soft max layer. Then train the deep neural network and find the corresponding output for that trained network and plot ROC curve. Similarly, find the output for the test data and plot the ROC curve using the test label. The ROC curve plotted is Spoof Accept Rate or False Accept Rate(FAR) VS Live Reject Rate (Genuine Reject Rate).

Results and Discussions

To achieve the best configuration, parameters of auto encoders are varied accordingly to obtain the maximum AUC.

First, Consider BSIF feature with combined data from digital and sagem sensors, we take one autoencoder and change the hidden layer size accordingly to find the optimal Area Under Curve(AUC). As you can see in the table (Refer Table 1), the AUC ranges from 0.76223 to 0.91134 while changing the hidden layer sizes as shown in the table. Logically, higher the hidden layer sizes, the better AUC. But, this might not be true in all cases as you can see for Hidden Layer size 1=500, the AUC obtained is less than the Hidden layer 1 having size=400. Similarly, perform and find the best configuration for LBP (Refer Table 3) and BGP(Refer Table 4) features.

Second, Consider BSIF feature with combined data from digital and sagem sensors, use 2 different auto encoders and stack them to get better results. In this process auto encoder 1 hidden layer sizes were varied from 100 to 500 and auto encoder 2 hidden layer sizes were chosen from 100 to 500. As we see the AUC in table (Refer Table 2) has improved significantly, we can conclude that the stacking of the auto encoders gave better results. The AUC ranges from 0.7863 to 0.92334. Choose optimal values of L2W regularization, Sparsity Proportion and Sparsity regularization. The best AUC is achieved with Hidden layer 1 size =400 and Hidden Layer 2 size=150. Similarly, perform and find the best configuration for LBP Refer Table 3) and BGP (Refer Table 4) features.

Thirdly, adding an another autoencoder so making it three auto encoders now, the AUC results are not quite as better as using just two auto encoders. Therefore, only two autoencoders are used here.

Fourth, changing L2W regularization or weight Decay parameter, significant change in the AUC is obtained from the ROC curves. It's generally low in the range of 0 to 1, it affects more for smaller weights than larger weights ([2], page 209). Weights that are smaller than a certain value can be eliminated ([2], page 209). So, ROC's were plotted for different values of weights starting from 0.004 to 0.016. It is seen that as the weights are increased the AUC values were dropping. (Refer Table 2)

Fifth, changing Sparsity Regularization parameter. It is observed that as the value of the Sparsity Regularization parameter increases, the AUC decreases. For Sparsity Regularization of 2 it was better than the Sparsity Regularization of 4. (Refer Table 2)

Sixth, varying Sparsity Proportion parameter. When this parameter is increased, the number of active neurons increases. So, AUC increases. Higher the Sparsity Proportion, better AUC value (Refer Table 2).

Seventh, In Task1, use the features LBP, BSIF and BGP individually to train and test the deep network. Data is taken individually from Digital and Sagem sensors and trained and tested accordingly. Take Digital sensor case, Consider BGP features, train and test the data completely. It is seen that the train AUC is 0.94565 and test AUC is 0.78421 (Refer Fig1). Now, consider BSIF features, train and test the data completely. It is seen that the train AUC is 0.92595 and test AUC is 0.83605 (Refer Fig3). Now, consider LBP features, train and test the data completely. It is seen that the train AUC is 0.93913 and test AUC is 0.79821(Refer Fig5). So, we can see that the test AUC is better for BSIF among LBP, BSIF and BGP considering digital sensor.
Take Sagem sensor case, Consider BGP features, train and test the data completely. It is seen that the train AUC is 0.91238 and test AUC is 0.85794 (Refer Fig2). Now, consider BSIF features, train and test the data completely. It is seen that the train AUC is 0.93019 and test AUC is 0.81046(Refer Fig4). Now, consider LBP features, train and test the data completely. It is seen that the train AUC is 0.95194 and test AUC is 0.83794 (Refer Fig6). So, we can see that the test AUC is better for BSIF among LBP, BSIF and BGP considering sagem sensor. So, we can see that the test AUC is better for BGP among LBP, BSIF and BGP considering Sagem sensor.

Eighth, In Task2, Same features are taken, but trained and tested on different sensors accordingly. Consider BGP features, train the data on digital sensor, validate the train data and test on sagem sensor. It is seen that the train AUC is 0.94565, validate AUC is 0.89016 and test AUC is 0.78421 (Refer Fig7). Now, consider BSIF features, train the data on digital sensor, validate the train data and test on sagem sensor. It is seen that the train AUC is 0.98982, validate AUC is 0.81411 and test AUC is 0.7534 (Refer Fig9). Now, consider LBP features, train the data on digital sensor, validate the train data and test on sagem sensor. It is seen that the train AUC is 0.94787, validate AUC is 0.91079 and test AUC is 0.5461 (Refer Fig11). So, here considering training on digital sensor and testing on sagem sensor, we see that BGP feature has the best test AUC. So, BGP is considered best feature among BGP, BSIF and LBP when trained on digital sensor and tested on sagem sensor. (Refer Table 5).

Now, consider training the features using sagem sensor and testing on on digital sensor. Consider BGP features, train the data on sagem sensor, validate the train data and test on digital sensor. It is seen that the train AUC is 0.98417, validate AUC is 0.97731 and test AUC is 0.45771(Refer Fig8). Now, consider BSIF features, train the data on sagem sensor, validate the train data and test on digital sensor. It is seen that the train AUC is 0.929, validate AUC is 0.9266 and test AUC is 0.56195 (Refer Fig10). Now, consider LBP features, train the data on sagem sensor, validate the train data and test on digital sensor. It is seen that the train AUC is 0.96856, validate AUC is 0.98628 and test AUC is 0.52169 (Refer Fig12). So, here considering training on sagem sensor and testing on digital sensor, we see that BGP feature has the best test AUC. So, BSIF is considered best feature among BGP, BSIF and LBP when trained on sagem sensor and tested on digital sensor. So, we can see the performance degraded. (Refer Table 5).

Ninth, now consider the Bonus case, which is evaluating fusion of features (such as LBP and BSIF) considering only digital sensor for training, validating and testing the data. Now, consider fusion of BGP and LBP features. Here, the LBP and BGP features are concatenated and trained using digital sensor. Validate the train data and then perform the testing on digital sensor. It is seen that the train AUC is 1(100%), validate AUC is 0.94189 and test AUC is 0.57721 (Refer Fig13). Now, Consider the fusion of LBP and BSIF features. Here, the BSIF and LBP features are concatenated and trained using digital sensor. Validate the train data and then perform the testing on digital sensor. It is seen that the train AUC is 1(100%), validate AUC is 0.94118 and test AUC is 0.51(Refer Fig14). So, we can see the performance degraded drastically (Refer Table 6).

Conclusions

In this project, Data set was provided by LivDet2011 consisting of live and spoof images fabricated using materials such as Silgum, Latex, Gelatin and Ecoflex separated into training and test sets (non-overlapping set). The original images from these databases and features such as Local Binary Pattern (LBP), Binary Statistical Image Features (BSIF) and Binary Gabor Pattern (BGP) are extracted from live and fake images using two different fingerprint sensors -Digital and Sagem. Then, trained auto encoders taking the different training sets using different parameters of auto encoders like different hidden layer sizes and different L2W, Sparsity regularization and Sparsity proportion and found the optimal value of those parameters which happened to be 2 auto encoders of hidden layer sizes of 400 and 150 and L2W and Sparsity regularizations of 0.004 and 2 and Sparsity Proportion of 0.4. Then used soft max layer to train the best configuration, stacked the auto encoder 1, auto encoder 2 and soft max layer together creating a deep network. Consequently, obtained training result and testing result. To achieve the best configuration, parameters of auto encoders are varied accordingly to obtain the maximum AUC. Two autoencoders with two different hidden layer size are considered. L2W regularization or weight decay parameter is changed accordingly. It is seen that as the weights are increased the AUC values were dropping. Fifth, changing Sparsity Regularization parameter. It is observed that as the value of the Sparsity Regularization parameter increases, the AUC decreases. Varying Sparsity Proportion parameter. When this parameter is increased, the number of active neurons increases. So, AUC increases. When the epochs value is increased, the AUC is increased.

References

1. Sparse auto encoder by Andrew Ng
2. J. C. Príncipe, et al., Neural and adaptive systems: fundamentals through simulations. New York: Wiley, 1999.
3. Rattani, Ajita, and Arun Ross. "Automatic adaptation of fingerprint liveness detector to new spoof materials." *Biometrics (IJCB), 2014 IEEE International Joint Conference on*. IEEE, 2014

4. Schmidhuber, Jürgen. "Deep learning in neural networks: An overview." *Neural Networks* 61 (2015): 85-117.
5. Rosasco, Lorenzo. "Sparsity Based Regularization." (2009).
6. Liao, Bin, et al. "An Image Retrieval Method for Binary Images Based on DBN and Softmax Classifier." *IETE Technical Review* 32.4 (2015): 294-303.

Appendices

Table 1- AUC's of 1 auto encoder with different Parameters, BSIF feature

Hidden Layer Size	L2W Regularization	Sparsity Regularization	Sparsity Proportion	AUC
100	0.004	1	0.25	0.76233
100	0.004	1	0.25	0.79865
200	0.012	1	0.25	0.82698
200	0.004	1	0.25	0.88943
300	0.004	2	0.4	0.87765
400	0.004	2	0.4	0.91243
400	0.008	4	0.4	0.90692
500	0.016	4	0.4	0.86987

Table 2-AUC's using 2 auto encoders, BSIF feature

Hidden Layer1 size	Hidden Layer2 size	L2W Regularization	Sparsity Regularization	Sparsity Proportion	AUC
100	100	0.004	1	0.25	0.78346
100	100	0.008	1	0.25	0.76256
200	100	0.012	1	0.25	0.82456
200	100	0.004	1	0.25	0.82114
300	100	0.004	2	0.4	0.90115
400	150	0.004	2	0.4	0.92352
400	200	0.008	4	0.4	0.88342
500	100	0.016	4	0.4	0.87113

Table 3-AUC's using 2 auto encoders, LBP feature

Hidden Layer1 size	Hidden Layer2 size	L2W Regularization	Sparsity Regularization	Sparsity Proportion	AUC
100	100	0.004	1	0.25	0.81226
100	100	0.008	1	0.25	0.83356
200	100	0.012	1	0.25	0.85984
200	100	0.004	1	0.25	0.81134
300	100	0.004	2	0.4	0.85696
400	150	0.004	2	0.4	0.90225
400	200	0.008	4	0.4	0.8771
500	100	0.016	4	0.4	0.89123

Table 4- AUC's using 2 auto encoders, BGP feature

Hidden Layer1 size	Hidden Layer2 size	L2W Regularization	Sparsity Regularization	Sparsity Proportion	AUC
100	100	0.004	1	0.25	0.85654
100	100	0.008	1	0.25	0.84394
200	100	0.012	1	0.25	0.81225
200	100	0.004	1	0.25	0.86597
300	100	0.004	2	0.4	0.91069
400	150	0.004	2	0.4	0.90225
400	200	0.008	4	0.4	0.88219
500	100	0.016	4	0.4	0.90236

Table 5- Task2

SENSOR	FEATURE	HL 1	HL 2	No of Autoencoders	L2WR	SR	SP	AUC Train	AUC Validate	AUC Test
Train-Digi, Test-Sage	LBP	400	150	2	0.004, 0.004	2,1	0.4, 0.4	0.94787	0.91079	0.5461
Train-Digi, Test-Sage	BSIF	400	150	2	0.004, 0.004	2,1	0.4, 0.4	0.98982	0.81411	0.7534
Train-Digi, Test-Sage	BGP	400	150	2	0.004, 0.004	2,1	0.4, 0.4	0.93807	0.89016	0.57721
Train-Sage, Test-Digi	LBP	400	150	2	0.004, 0.004	2,1	0.4, 0.4	0.96856	0.98628	0.52169
Train-Sage, Test-Digi	BSIF	400	150	2	0.004, 0.004	2,1	0.4, 0.4	0.929	0.9266	0.56195
Train-Sage, Test-Digi	BGP	400	150	2	0.004, 0.004	2,1	0.4, 0.4	0.98417	0.97731	0.43771

Table 6- Bonus

SENSOR	FEATURE	HL1	HL2	No of Autoencoders	L2WR	SR	SP	AUC Train	AUC Validate	AUC Test
Train-Digi, Test-Sage	LBP+BGP	400	150	2	0.004,0.004	2,1	0.4,0.4	1	0.94189	0.5772
Train-Digi, Test-Sage	BSIF+LBP	400	150	2	0.004,0.004	2,1	0.4,0.4	1	0.94118	0.50013

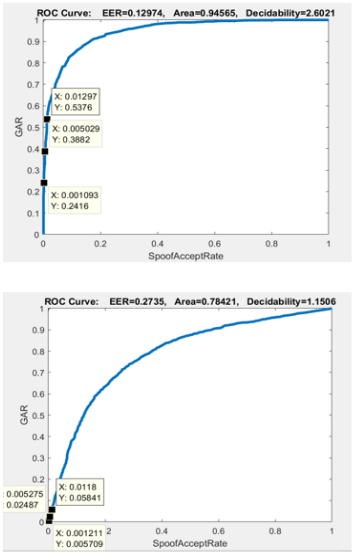

Fig1: Training Digital BGP and testing with Digital BGP

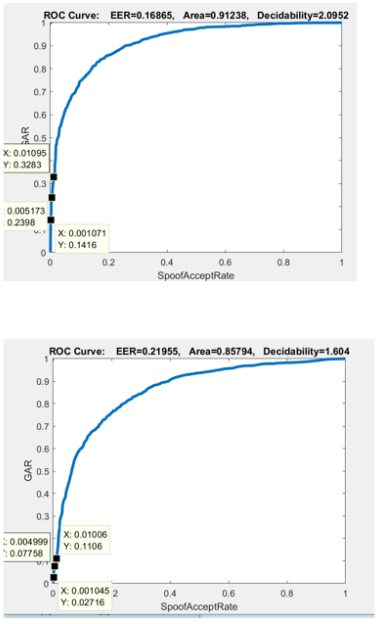

Fig2: Training Sage BGP and testing with Sage BGP

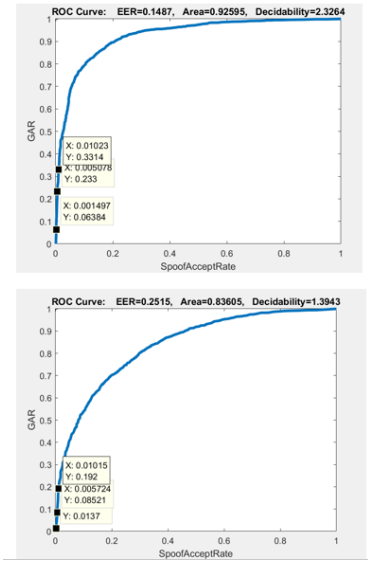

Fig3: Training Digital BSIF and testing with Digital BSIF

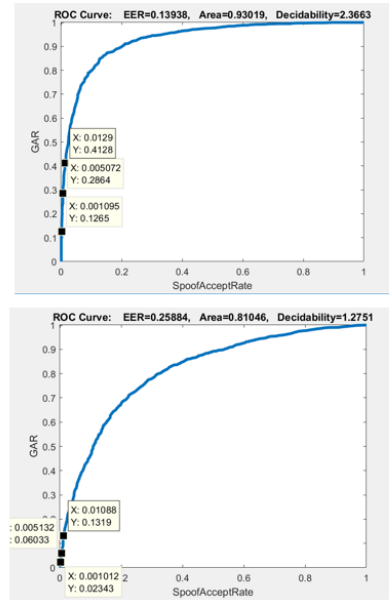

Fig4: Training Sage BSIF and testing with Sage BSIF

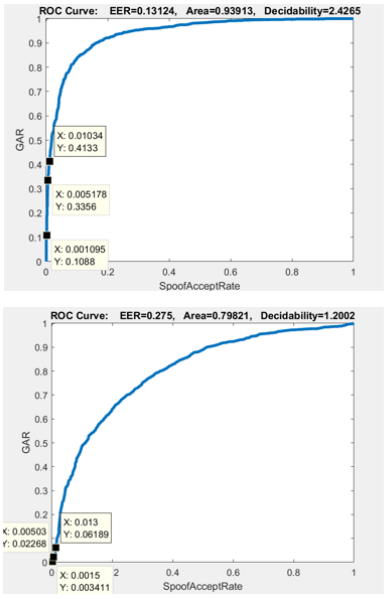

Fig5: Training Digital LBP and testing with Digital LBP

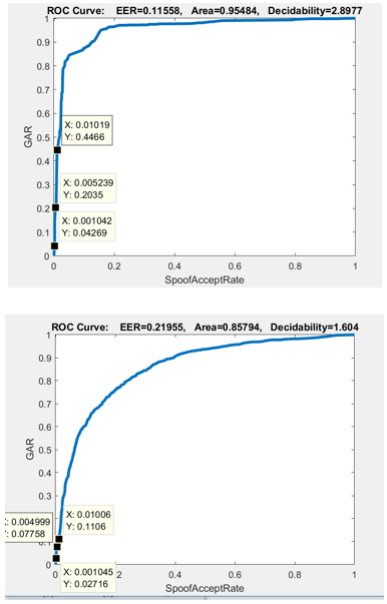

Fig6: Training Sage LBP and testing with Sage LBP

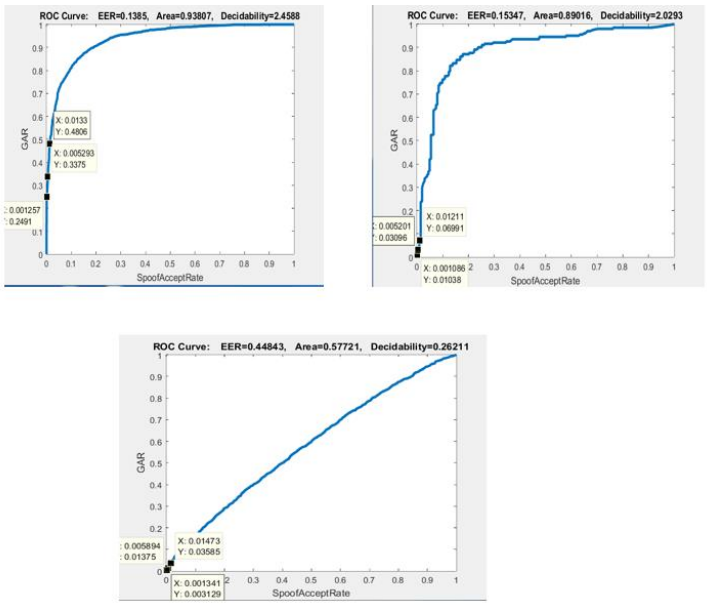

Fig7: Training Digi BGP and testing with Sage BGP

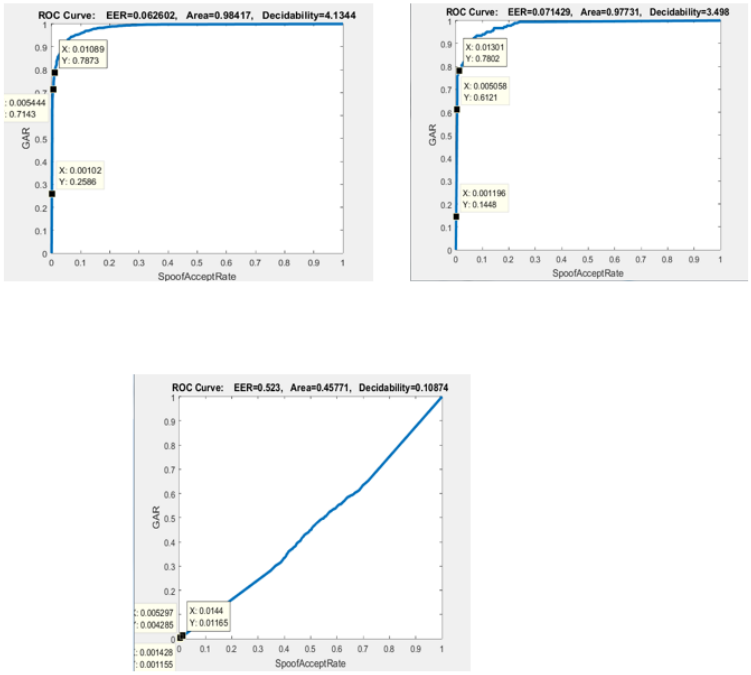

Fig8: Training Sage BGP and testing with Digi BGP

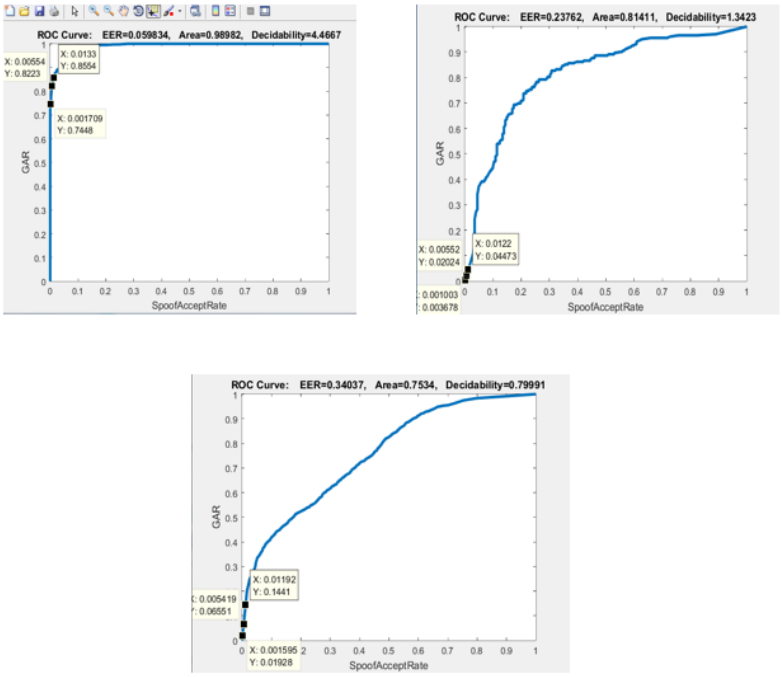

Fig9: Training Digi BSIF and testing with Sage BSIF

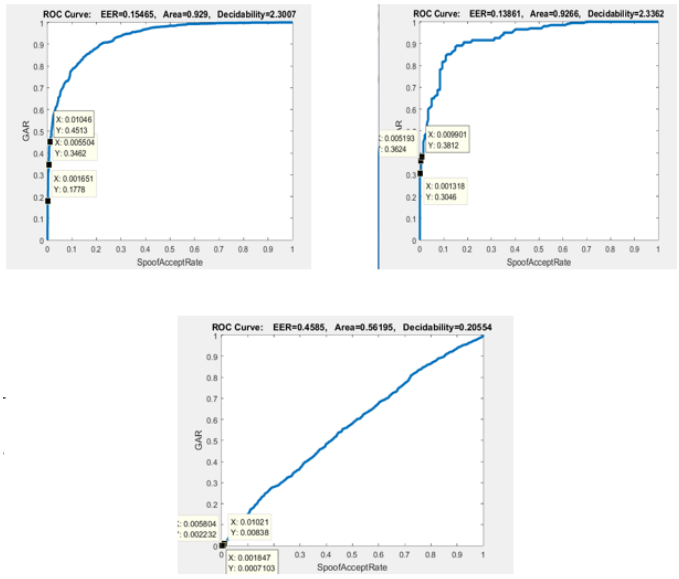

Fig10: Training Sage BSIF and testing with Digi BSIF

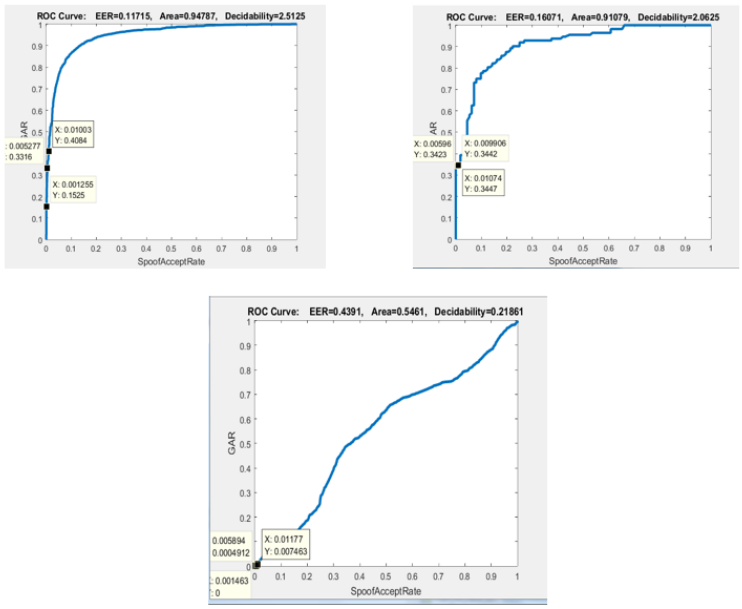

Fig11: Training Digi LBP and testing with Sage LBP

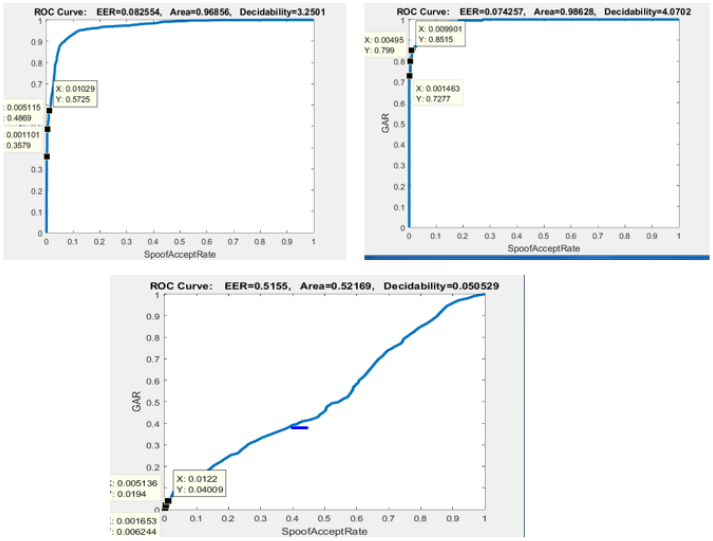

Fig12: Training Sage LBP and testing with Digi LBP

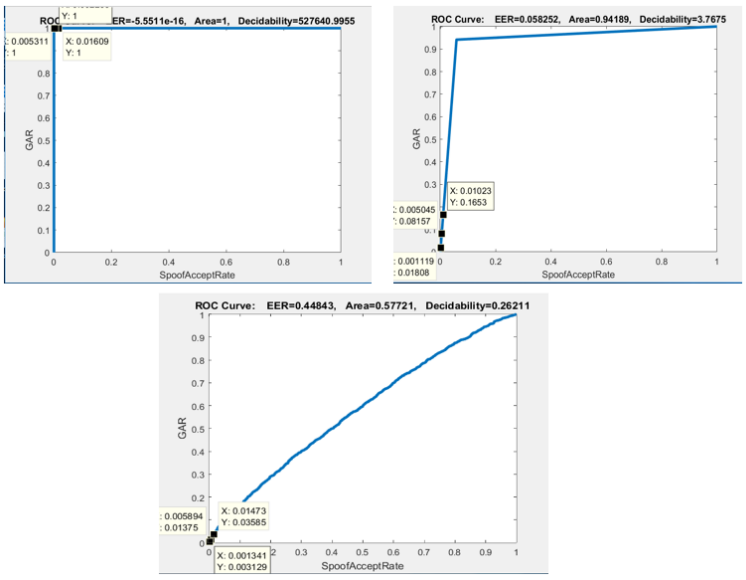

Fig13: BONUS Training and testing LBP+BGP on digital sensor

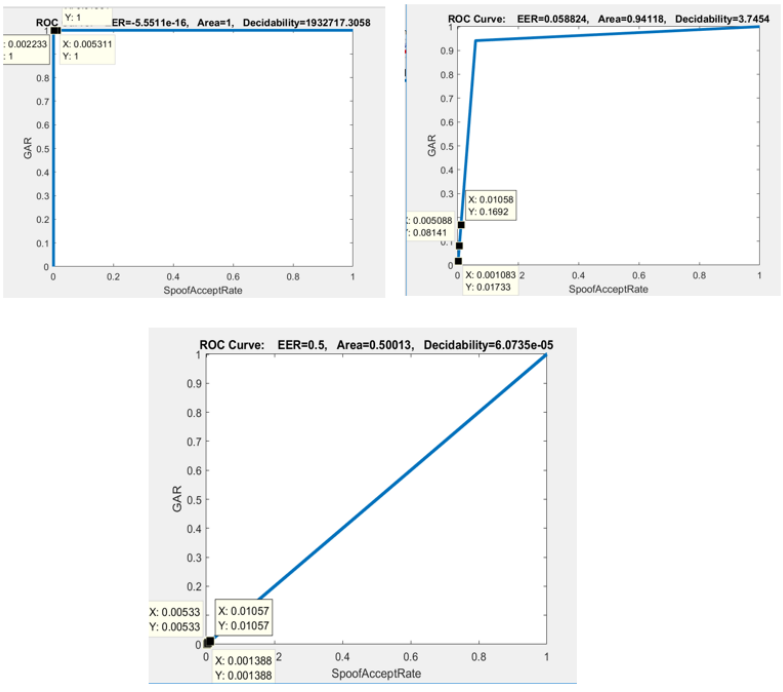

Fig14: BONUS Training and testing LBP+BSIF on digital sensor