FINGERPRINT SPOOF DETECTION USING DEEPNETS

SUGGALA PRUDHVI SAI
School of Computing and Electrical Engineering
University of Missouri-Kansas City
ps5m6@mail.umkc.edu

Declaration: I Mr. suggala Prudhvi sai, 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

Fingerprint spoof (fake) detection algorithms are used to distinguish between spoof and live fingerprints. This approach can be used as a pre-check before fingerprint recognition system to help in increase overall performance of fingerprint recognition system .The paper also deals with problems of performance degradation, when net is trained and tested with data from different fingerprint scanner vendors. The aim of this work is to design a deep nets based on sparsity auto encoders using different features extracted from LivDet 2011 and proposed a feature level fusion scheme helps in improving performance by up to 3% compared with net trained and tested with same sensor data .

INTRODUCTION

Biometrics are used as authentication check to provide access, but main deterrent of Biometric systems are spoofing attacks. Many spoof attack schemes are available to spoof the biometric systems [3], fake fingerprints can be easily made by materials such as gelatin silicone ,etc. Fingerprint sensors from different vendors are unable to produce identical data from same fingerprint scan [2].So this led to take data from different sensors at time of enrolment, if the system need to perform flawless in real world scenarios.

This pre-classification is supporting the recognition technique by rejecting the spoof sample not to move towards the recognition layer of the system. We started with train and test our system using same vendor sensor data which resulted great accuracy and performance gradually decreased when we try to test with other vendor sensor.

We have used the features (Binary Gabor patterns, Local binary patterns, Binary statistical image features) extracted from data to our deep nets to train our spoof detector. We had used the sparsity auto encoders in our system as pattern recognizers and softmax layers works as classifier in the end. Reason to choose Auto encoders, they find better patterns when are fed with good amount of data and trained in a regularized way [[4] page num - 12] .They may perform better than any native feature extraction technique. We used different combinations of parameters to get the optimal configuration to the auto encoders

used, evaluated outputs based on Receiver operating curve (ROC) that obtained in each scenario for spoof acceptance rate and live reject rate. Finding patterns from raw data may way more complex than finding from features extracted from data.

IMPLEMENTATION

In this project we are provide with the features extracted from livedet 2011 , Which consists of 3 features (BGP,BSIF,LBP) from 2 different sensor vendors digi , sage and a test data to check performance of system.

We found configurations of deepnets for Local binary patterns, Binary Statistical image features and Binary Gabor patterns of Digi and Sage when fed with Livedet 2011 data . Found a combination of features (BGP,LBP) that results in improvement of total performance of system than when it is trained with data similar features from both sensors i.e (BGP digi ,BGP sage). We also found performance degradation of system, when trained and tested with data from different sensors.

We used different combinations of hidden layers, number of auto encoders and parameters in cost function [formula 1]. We scaled the both features (BSIF, BGP) with a factor of 10^4 to increase the magnitude. Performance is plotted between spoof accept rate to Live reject rate.

We concatenated same features from two different sensors, We end up training three different nets trained for three features. We had divided train data into train and validation (90:10)% as a pre-check to test.

We tuned L2 Weight regularization (λ) which is l2 norm of Weights of net, which is penalty in cost function [formula 1], which helps neural net not to remember data. That squeezes insignificant weights to 0, deactivates the neurons also called as RIDGE regularization technique. We had chosen greater number of hidden layers than the inputs ,which may results in interesting structures when applied with good parametric values [[4] page13].Sparsity proportion(ρ) determines the number of neurons that are need to be activated of all the neurons in the net , We chosen (2-5%) of number of hidden layers to our problem . Sparsity regularization (β) is pressure value that is inserted on sparsity proportion. Sparsity proportion is intuitively chosen considering the number of hidden layers.Softmax layer is used as a classifier, trained by patters extracted from auto encoders.

When all trained auto encoders as stacked together which is trained again to fine tune the network as the final stage , which helps in boosting the performance of network [5].All the networks are trained using Back propagation algorithm and initialized with random weights.

Then deepnet is trained with data from one sensor and test with other of same feature i.e. (trained with DIGI BGP and tested with SAGE BGP), this is done for all 6 combination of features and sensors.

Results and Discussion

We tried different hidden layers sizes, number of auto encoders by choosing different combinations of parameters of sparsity Regularization, Sparsity proportion and L2weight regularization. We got best configuration by which gives less area under cure in ROC.

Although we even search best configuration from confusion matrix, but it is biased to the threshold limit that is chosen to classify from one to other. so, ROC be the best metric to evaluate the performance of the curve.

We have a test data included in dataset, but we also used a validation set which is taken from train data to validate the net before testing. We had taken 10% training data of two classes as to validate net, we followed this for all features.

We initially iterated with many combinations of hidden layers and auto encoders, then used a configuration (400,150) which resulted in AUC (0.0965) where it is trained for LBP feature from both sensors. Same pattern is followed with BSIF and BGP and best configurations are BSIF (750,200) which results AUC (0.0077) and BGP (500,150) .We had used 2 auto encoders stacked together gave the best performance. We had almost encountered 0% AUC) for all train and validation in 3 cases [refer figure 1, 2, 3].

Where same configuration are used to train data with single sensor and tested with other that had reported huge performance degradation .We found that the network is unable to capture the pattern which is common with both sensor data and all its output are almost of 0.5 AUC .configuration from the above cases adopted failed in this scenario [refer figure 4-9].

L2WeightRegularizarion responsible for weight decay, which needed to small as range of 0.004 ([1] page 209) but contradicting to it, we used high regularization like 0.5 for both LBP and BSIF [refer figure]. That gives intuition that there may be a combination less than this, because high regularization mean making many weights to zero. So, no necessary of that many neurons to solve it but we cannot surely claim that reduce in (λ) results in best performance.

Fusion of data i.e. (BSIF, BGP, LBP) is performed to improve the performance of data .The combination of digi (BGP, LBP) shows improvement of 3% in AUC when compared with the initial combinations [refer image]. It has 2 auto encoders of hidden layers (350, 75) .This kind fusion at data level helps in improve the performance of system.

Conclusion

The above method can provide accuracy as described in results for features Local binary patterns, Binary Statistical image features and Binary Gabor patterns extracted from livedet 2011. Auto encoders with high L2 Weight regularization factor like 0.5 can even performed better in this case. There is large performance degradation when train, tested on same and different sensors, This make a conclusion that different sensors provide different data .To make our system more accurate, we need to check both sensors to be matched at the time of enrolment and testing or we need get data from different sensors at the time of enrolment .

If our main aim to increase the accuracy of system using a single sensor, the feature fusion help to achieve it, as in the above problem 3% improvement is seen.

Spoof detections systems like above successfully isolated 90% spoof for the test data provide likely to help finger recognition system from spoofing attacks.

References

[1] J. C. Principe, et al., Neural and adaptive systems; Fundamentals through simulations. New York: Wiley, 1999

[2] An introduction evaluating biometric systems
P.J. Phillips Nat. Inst. of Stand. & Technol., CO, USA A. Martin

[3] Fingerprint Recognition at the Supermarket as insecure as Biometrics in Passports http://www.ccc.de/en/updates/2007/umsonst-im-supermarkt [web article]

[4] Notes by Andrew Ng on sparsity Auto encoders.

[5] EXTRACTING DEEP BOTTLENECK FEATURES USING STACKED AUTO-ENCODERS -Jonas Gehring1 Yajie Miao2 Florian Metze2 Alex Waibel1,2

Appendices

All the figures contains train validation and test roc respectively figure [1-9].

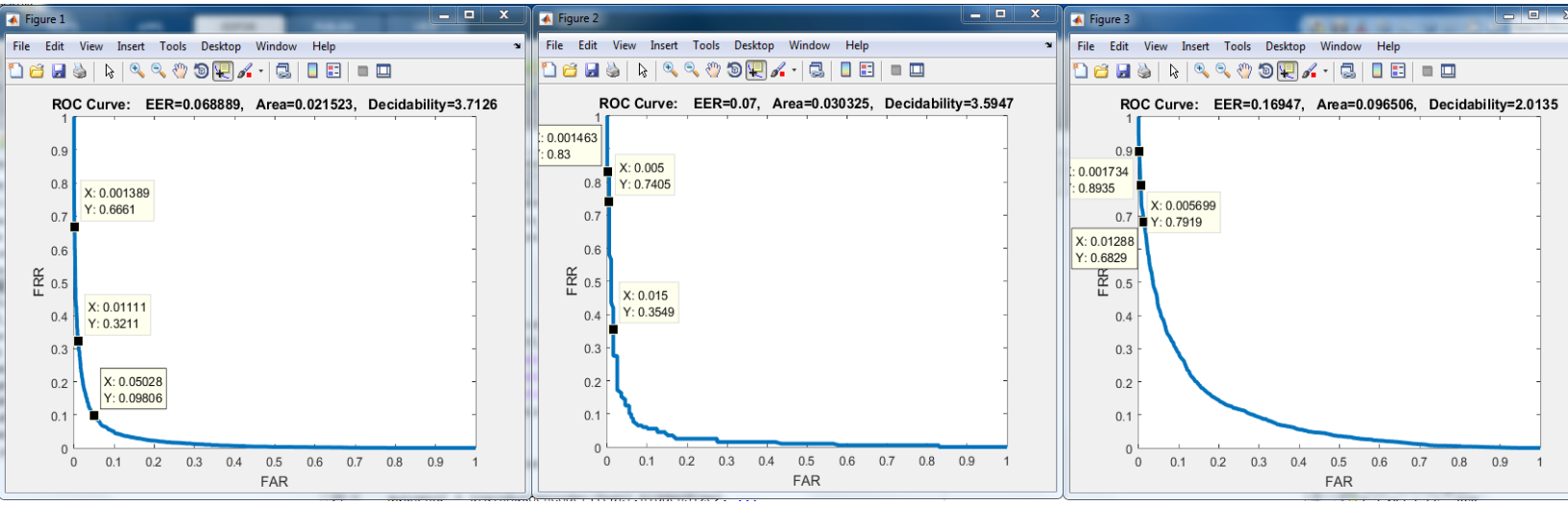

Figure 1 –ROC of LBP train data – [digi and sage]

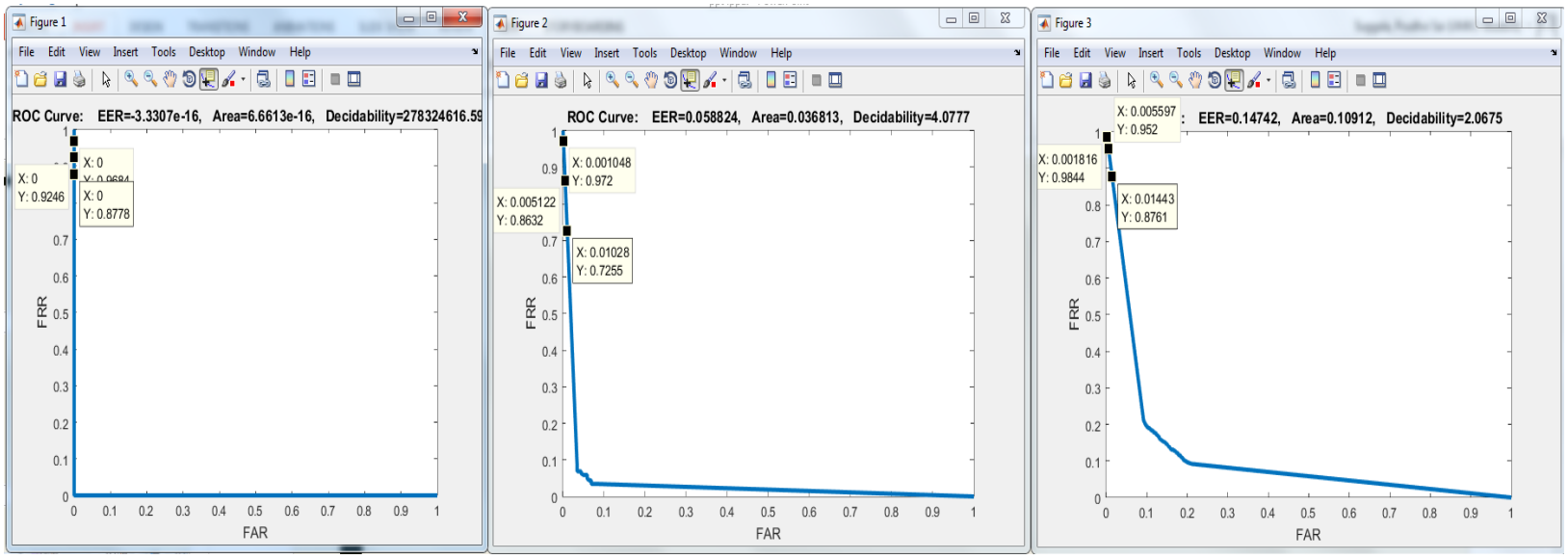

Figure 2 –ROC of BGP train data – [digi and sage]

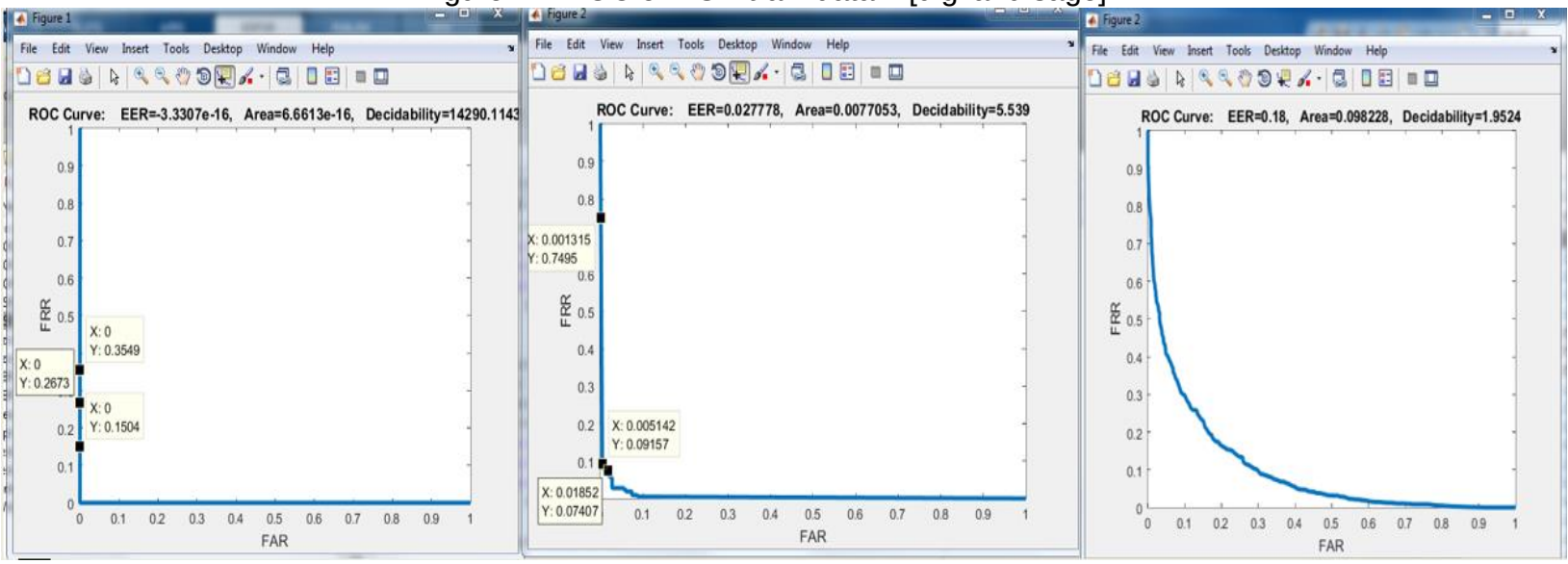

Figure 3 – Roc's of BSIF train data – [digi sage]

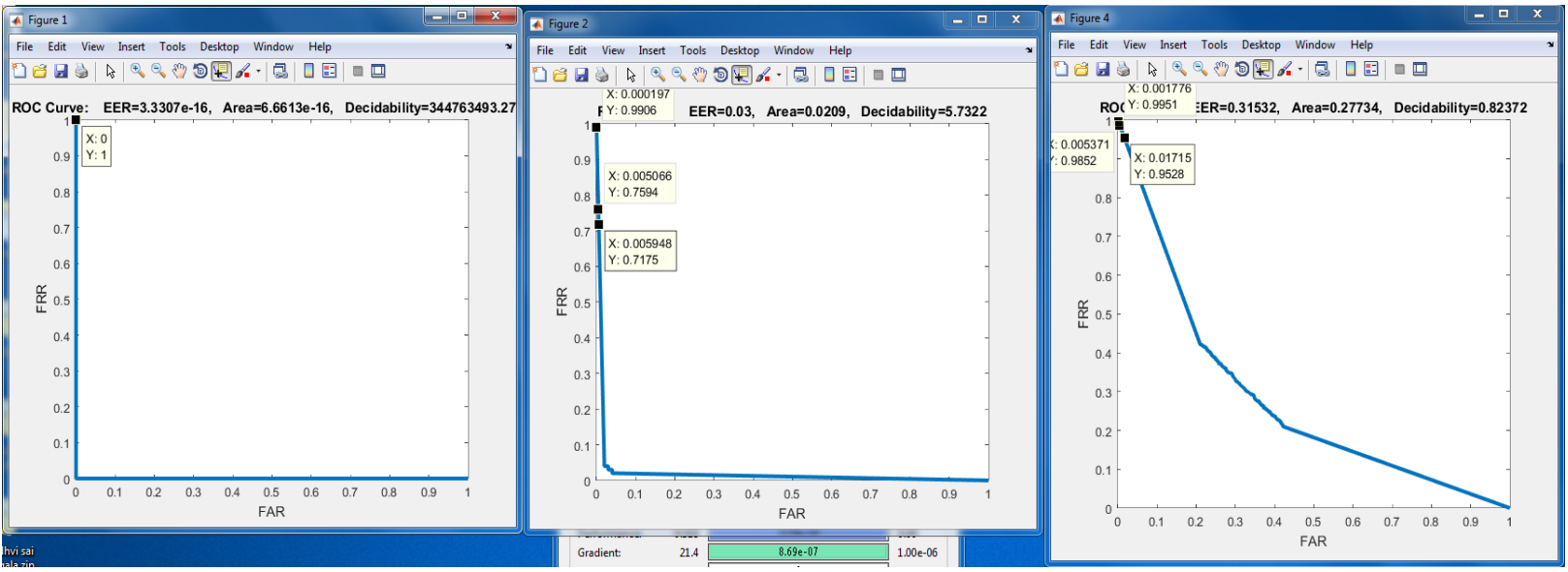

Figure 4 – Roc's of train data : DIGI BGP test data :SageBGP

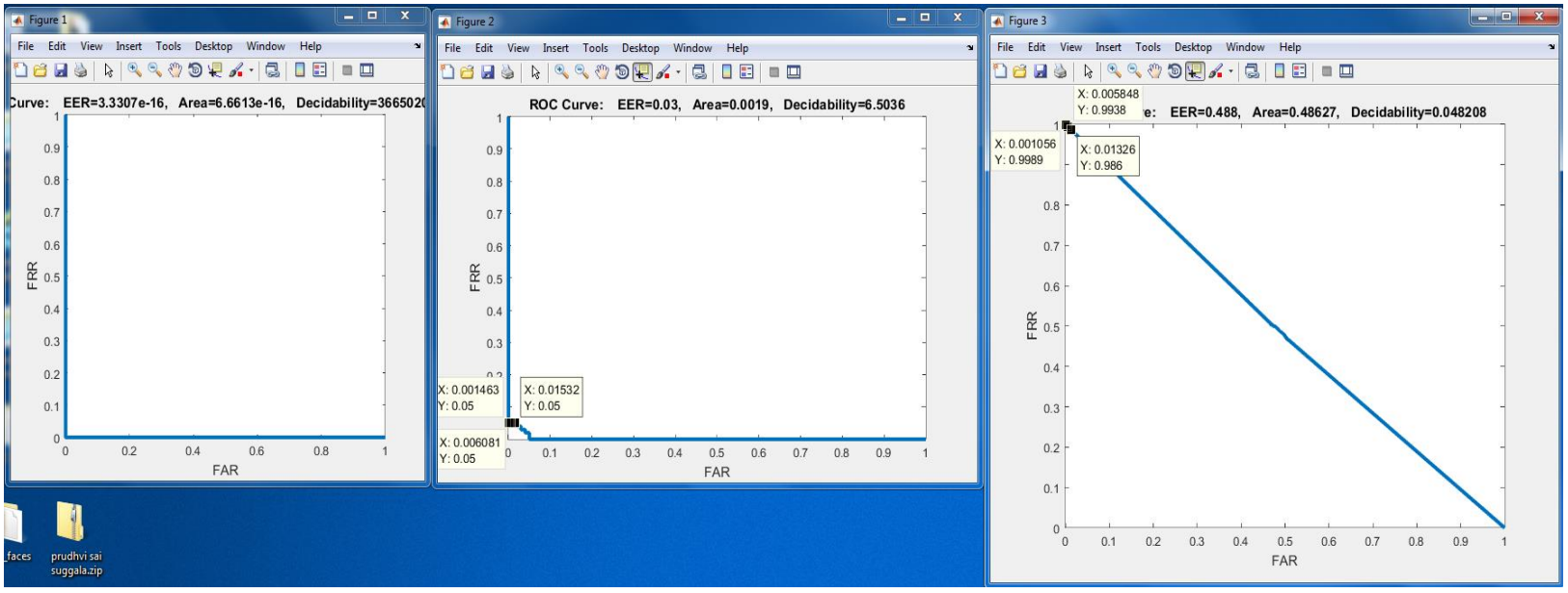

Figure 5– Roc's of train data : Sage BGP test data :Digi BGP

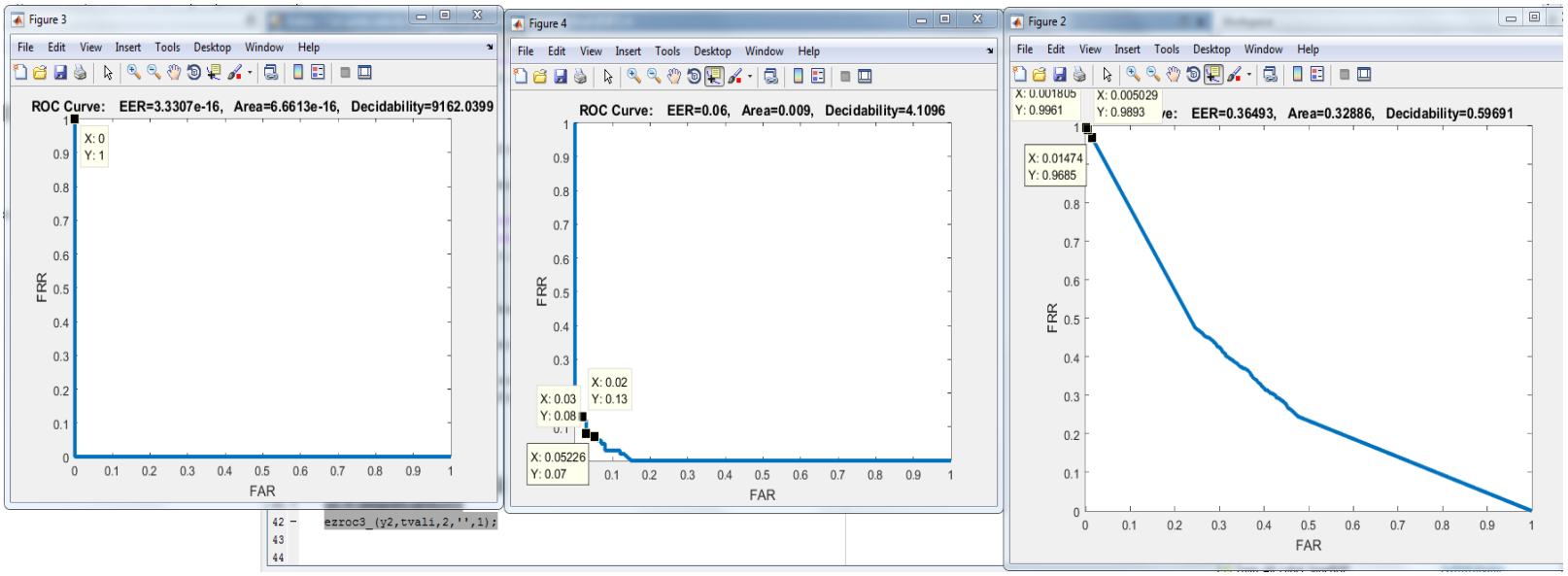

Figure 6– Roc's of BSIF train data : digi sensor test data :Sage sensor

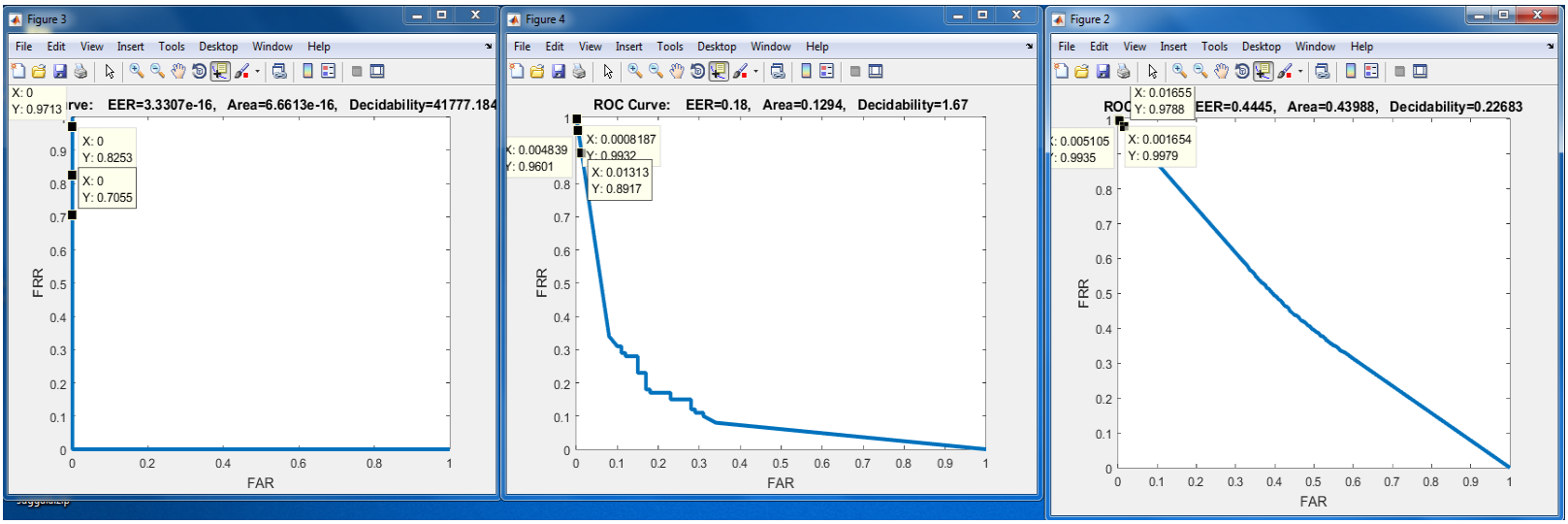

Figure 7– Roc's of BSIF train data : Sage sensor test data :Digi sensor

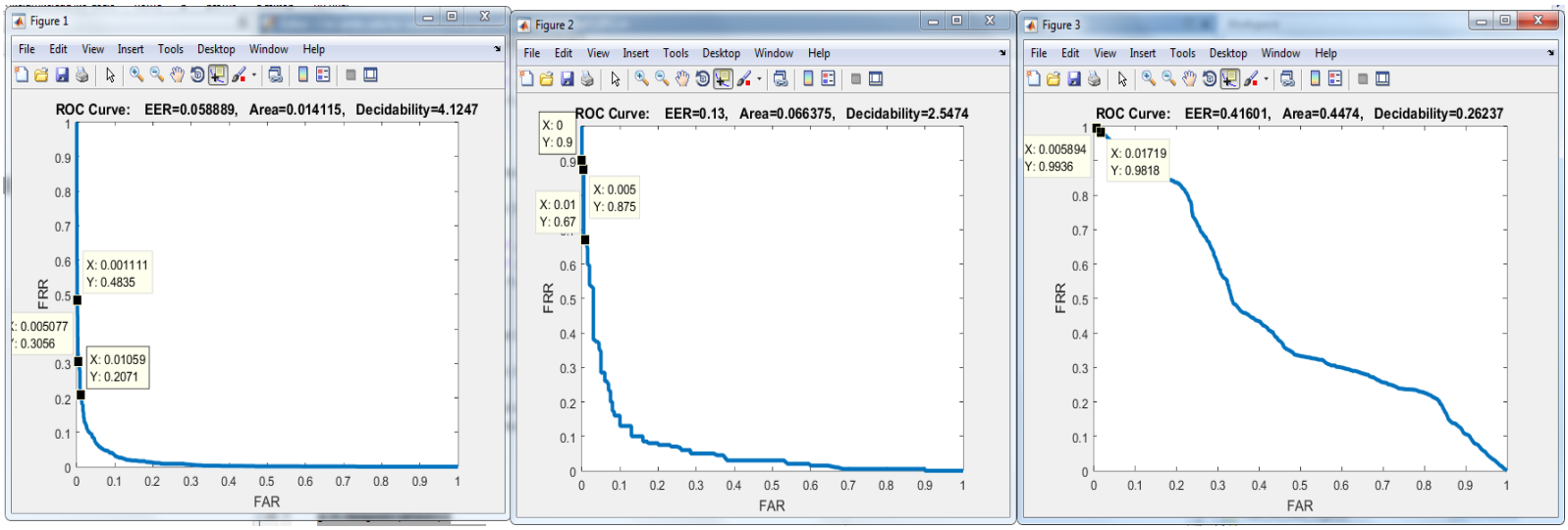

Figure 8– Roc's of LBP train data : Digi sensor test data :Sagesensor

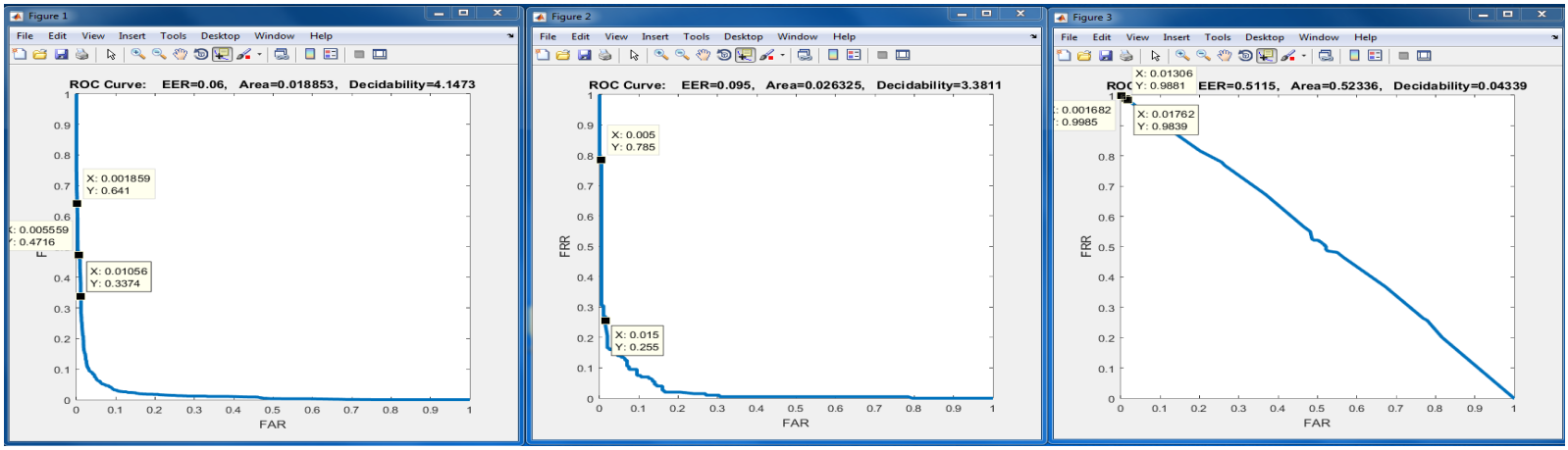

Figure 9– Roc's of LBP train data : Sage sensor test data :Digi sensor

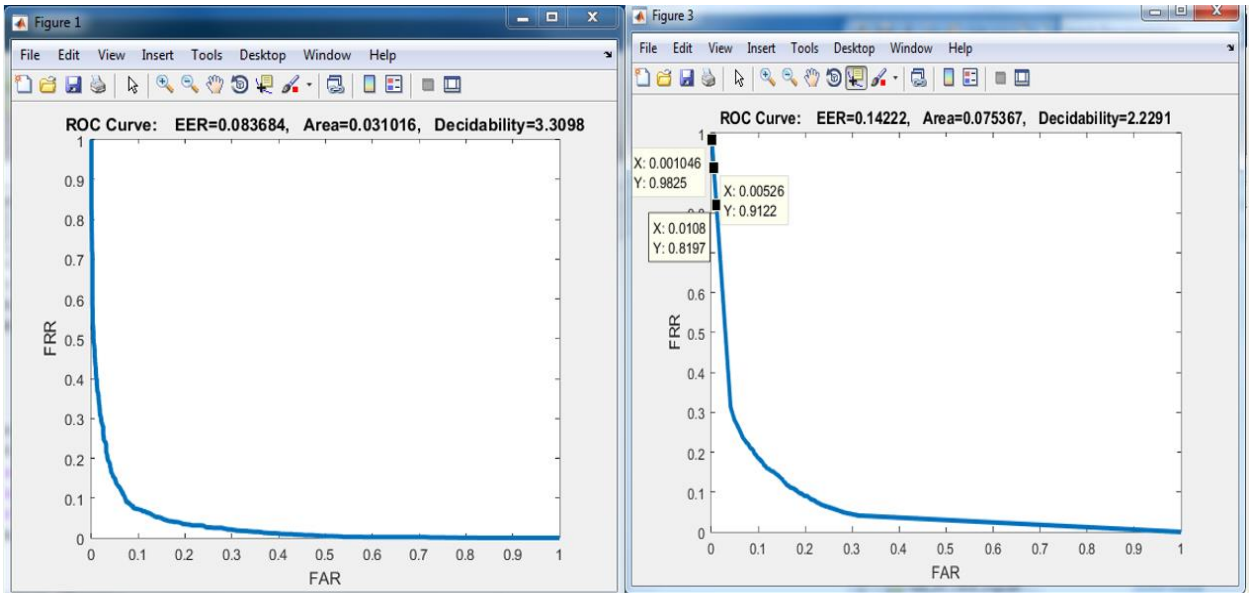

Figure 10– Roc's of train data : [digi LBP and digi BGP]

$$E = \underbrace{\frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K}\left(x_{kn} - \hat{x}_{kn}\right)^2}_{\text{mean squared error}} + \lambda * \underbrace{\Omega_{weights}}_{\substack{L_2 \\ \text{regularization}}} + \beta * \underbrace{\Omega_{sparsity}}_{\substack{\text{sparsity} \\ \text{regularization}}},$$

Formula -1 (F1)