Drone Detection by Neural Network using SURF Feature and GLCM Method

SUBMITTED BY:

Tanzia Ahmed (14101136)

Bir Ballav Roy (14301005)

Tanvir Rahman (14301072)

Supervisor:

Dr. Jia Uddin

Assistant Professor

Department of Computer Science and Engineering

SUBMISSION DATE: 3.25.18

Declaration

We, hereby declare that the thesis “Drone Detection by Neural Network using SURF and GLCM Method” is based on the results found by ourselves under the supervision of Assistant Professor Dr. Jia Uddin. It is carried out for the degree of Bachelor of Science in Computer Science. Materials of work used here found by other researchers are acknowledged by reference. This Thesis, neither in whole or in part, has been previously submitted for any degree. We, hereby declare that the thesis “Gender Recognition from Facial Images Detected using DSP Algorithm” is based on the results found by ourselves under the supervision of Assistant Professor Dr. Jia Uddin. It is carried out for the degree of Bachelor of Science in Computer Science. Materials of work used here found by other researchers are acknowledged by reference. This Thesis, neither in whole or in part, has been previously submitted for any degree.

Signature of Supervisor

Signature of Authors

Dr. Jia Uddin
Assistant Professor

Department of Computer Science and Engineering

BRAC University

Tanzia Ahmed (14101136)

Bir Ballav Roy(14301005)

Tanvir Rahman(14301072)

ABSTRACT

This presents a drone detection method using image processing. In the proposed model, a hybrid feature extraction method using SURF and GLCM is utilized. As a machine learning tool, we use a neural network for pattern recognition to train and test. Finally, we measure the performance of the proposed model using cross entropy. For our tested drone dataset, experimental results demonstrate improved performance over state-of-art models by exhibiting less cross entropy and percentage error. It also presents experimented results of drone detection using different combination of methods and the results why it is recommended to use our approach. The combinations we used are SURF, GLCM, SURF with GLCM, and MSER. Here we focused on an optimal combination of performance and error percentage results. The recommended approach is to detect drones with very minimum error percentage and very high performance.

Acknowledgment

We would first like to thank our thesis advisor Professor Dr. Jia Uddin of the Department of Computer Science & Engineering at BRAC University. The door to his office was always open whenever we ran into a trouble spot or had a question about our research or writing. He consistently allowed this paper to be our own work but steered us in the right the direction whenever he thought we needed it.

Finally, we are grateful to the influences and support of faculties, friends from BRAC University and our family members, which has been a giant motivation in this thesis journey. This accomplishment would not have been possible without them. Thank you.

Table of Content

Chapter 4: Extremal Setup and Result Analysis

Chapter 5: Conclusions and Future Works

List of Figures

List of Tables

Acronyms

GLCM: The Gray Level Co-occurrence Matrix.

MSER: Maximally Stable Extremal Region.

RGB: Red Green Blue (3 dimensional).

SURF: Speed Up Robust Features.

CHAPTER 01

INTRODUCTION

1.1 Motivations

In recent years, the application of neural networking in the field of object detection is rapidly growing momentum, and that is for good reasons. Deep learning is another prospect of it. Here, we tried to detect drones while it is still in the air of sensitive areas. For that, we used four combinations of different methods and tried to compare them in our preferred context. Drones are being used for different purposes nowadays. For example, it is used for providing unwanted materials to prison [Telegraph, February 16, 2016]. This is just one example, and there are quite a few instances of it. In this paper, we will show an better way to detect drones, and give reasons supported by evidence why we think so. The methods that have been used here are GLCM, MSER, SURF, and their different combinations. We took basic neural network tool of MATLAB, and we trained it to detect our desired objects.

Object detection is one of the major divisions in the field of computer vision and there are many types of research that have been conducted in this area. Object detection has been done using deep learning, it has been done with the help of neural network; in both cases, and several methods were used. Before expected features were calculated by the help of Support Vector Machine (SVM) [1-3] as a tool. In different cases, entropy theory [3], Blob [4] detection has been used for feature extraction. Here we used SURF feature extraction method which is essentially a Blob detection method. In every case we needed to do two things, detect an object and recognize it later. However, it is crucial that we do it with lowest computational cost. That is why we need to focus on performance and maintain the error percentage at very low at the same time. Neural networks are

very sensitive even to the lowest change of any properties of an object. Otherwise, it may lead to inefficient generalization of the results [5].

1.2 Contribution Summary

The summary of the main contributions is as follows:

- Feature extraction using MSER method
- Feature extraction using SURF feature method.
- Feature extraction using GLCM method.
- Training neural network.
- A better way to detect and recognize drone using GLCM and SURF combine
- Comparative analysis among them

1.3 Thesis Orientation

The rest of the thesis is organized as follows:

- Chapter 2 includes the necessary background information regarding the used algorithm.
- Chapter 3 presents our proposed model and its implementation.
- Chapter 4 demonstrates the experimental results and comparison.
- Chapter 5 concludes the thesis and states the future research directions.

CHAPTER 02

BACKGROUND INFORMATION

2.1 The Gray Level Co-occurrence Matrix (GLCM)

The gray level Co-event Matrix (GLCM) and related surface component estimations are photo examination strategies. Given a photograph made out of pixels each with a force (a particular dim degree), the GLCM is a classification of ways frequently remarkable blends of dark extents co-emerge in a photograph or picture portion. Surface capacity counts utilize the substance of the GLCM to give a measure of the variant in force (a.k.a. picture surface) at the pixel of intrigue.

stats = graycoprops (glcm, residences) figures the realities determined in houses from the gray stage co-frequency framework GLCM. GLCM is an m-with the guide of-n-by means of p cluster of substantial dim stage co-commonness networks. In the event that GLCM is a variety of GLCMs, details are a variety of certainties for each GLCM.

graycoprops standardizes the dim degree co-predominance grid (GLCM) all together that the entirety of its components is equivalent to one. Everything about, (c) inside the standardized GLCM is the joint plausibility occurrence of pixel sets with a depicted spatial dating having dim stage esteems r and c inside the photo. graycoprops utilizes the standardized GLCM to ascertain homes. Table-1 below shows all the properties that GLCM algorithm has along with their respective formula and description.

Table 1: Shows the properties of GLCM

Property	Description	Formula
'Contrast'	Returns a measure of the intensity contrast between a pixel and its neighbor over the whole image. Range = [0 (size(GLCM,1)-1)^2]	$\sum_{i,j} \lvert i-j \rvert^2 p(i,j)$
'Correlation'	Returns a measure of how correlated a pixel is to its neighbor over the whole image. Range = [-1 1]	$\sum_{i,j} \frac{(i-\mu i)(j-\mu j)p(i,j)}{\sigma_i \sigma_j}$
'Energy'	Returns the sum of squared elements in the GLCM. Range = [0 1]	$\sum_{i,j} p(i,j)^2$
'Homogeneity'	Returns a value that measures the closeness of the distribution of elements in the GLCM to the GLCM diagonal. Range = [0 1]	$\sum_{i,j} \frac{p(i,j)}{1+\lvert i-j \rvert}$

2.2 Speed-up Robust Features (SURF)

SURF utilizes square-molded channels as an estimate of Gaussian smoothing. (The SIFT approach utilizes fell channels to distinguish scale-invariant trademark focuses, where the distinction of Gaussians (DoG) is ascertained on rescaled pictures dynamically.) Filtering the picture with a square is significantly quicker if the fundamental picture is utilized:

$$S(x.y) = \sum_{i=0}^{x}\sum_{j=0}^{y} I(i,j) \qquad 1$$

The sum of the original image within a rectangle can be evaluated quickly using the integral image, requiring evaluations at the rectangle's four corners.

SURF uses a blob detector based on the Hessian matrix to find points of interest. The determinant of the Hessian matrix is used as a measure of local change around the point and points are chosen where this determinant is maximal. In contrast to the Hessian-Laplacian detector by Mikolajczyk and Schmid, SURF also uses the determinant of the Hessian for selecting the scale, as is also done by Lindeberg. Given a point p=(x, y) in an image I, the Hessian matrix H (p, σ) at point p and scale σ, is:

$$\mathrm{H(p,\sigma)} = \begin{pmatrix} L_{xx}(p,\sigma) & L_{xy}(p,\sigma) \\ L_{yx}(p,\sigma) & L_{yy}(p,\sigma) \end{pmatrix} \qquad 2$$

The box filter of size 9×9 is an approximation of a Gaussian with σ=1.2 and represents the lowest level (highest spatial resolution) for blob-response maps.

Interest points can be found at different scales, partly because the search for correspondences often requires comparison images where they are seen at different scales. In other feature detection algorithms, the scale space is usually realized as an image pyramid. Images are repeatedly smoothed with a Gaussian filter, then they are subsampled to get the next higher level of the pyramid.

Therefore, several floors or stairs with various measures of the masks are calculated:

$$\sigma_{approx} = \text{current filter size} \times \left(\frac{base\ filter\ scale}{base\ filter\ size}\right) \quad 3$$

The scale space is divided into a number of octaves, where an octave refers to a series of response maps covering a doubling of scale. In SURF, the lowest level of the scale space is obtained from the output of the 9×9 filters.

Hence, unlike previous methods, scale spaces in SURF are implemented by applying box filters of different sizes. Accordingly, the scale space is analyzed by up-scaling the filter size rather than iteratively reducing the image size. The output of the above 9×9 filter is considered as the initial scale layer at scale *s* =1.2 (corresponding to Gaussian derivatives with σ = 1.2). The following layers are obtained by filtering the image with gradually bigger masks, taking into account the discrete nature of integral images and the specific filter structure. This results in filters of size 9×9, 15×15, 21×21, 27×27... Non-maximum suppression in a 3×3×3 neighborhood is applied to localize interest points in the image and over scales. The maxima of the determinant of the Hessian matrix are then interpolated in scale and image space with the method proposed by Brown, et al. Scale-space interpolation is especially important in this case, as the difference in scale between the first layers of every octave is relatively large.

2.3 Maximally Stable Extremal Regions (MSER)

In PC vision, maximally stable extremal areas (MSER) are utilized as a strategy for blob location in pictures. This system was proposed by Matas et all to discover correspondences between picture components from two pictures with various perspectives. This strategy for separating a thorough number of comparing picture components adds to the wide-standard coordinating, and it has prompted better stereo coordinating and protest acknowledgment calculations

The original algorithm of Matas et al. is *O(n log(log(n)))* in the number *O(n)* of pixels. It proceeds by first sorting the pixels by intensity. This would take *O(n log(log(n)))* time, using BINSORT. After sorting, pixels are marked in the image, and the list of growing and merging connected components and their areas is maintained using the union-find algorithm. This would take *O(n)* time. In practice, these steps are very fast. During this process, the area of each connected component as a function of intensity is stored producing a data structure. A merge of two components is viewed as termination of the existence of the smaller component and an insertion of all pixels of the smaller component into the larger one. In the extremal regions, the 'maximally stable' ones are those corresponding to thresholds where the relative area change as a function of the relative change of threshold is at a local minimum, i.e. the MSER are the parts of the image where local binarization is stable over a large range of thresholds.

The segment tree is the arrangement of every single associated part of the edges of the picture, requested by consideration. Effective (semi direct whatever the scope of the weights) calculations for figuring it exists. Accordingly this structure offers a simple path for executing MSER.

More recently, Nister and Stewenius have proposed a truly (if the weight is small integers) worst-case method in, which is also much faster in practice. This algorithm is similar to the one of Ph. Salembier et al.

CHAPTER 03

PROPOSED MODEL

The best model in our research that we will recommend is the one where we have used both SURF and GLCM feature extraction algorithms. The approach is described step by step below with a flowchart-

3.1 RGB to Gray-scale Image and image rotation

At first, we have taken input images randomly and converted the 3D image to gray-scale 2D image. This will return a 2D array of double values.

The 2D image is rotated to 90° in order to keep the column size fixed to 64 for both train and test dataset [9].

3.2 Feature Extraction by SURF

Then each image's features have been extracted using blob detection algorithm, SURF [15]. The interest points are deduced by Hessian method using $equation\ 2$ [14]. It uses Gaussian smoothing method for local neighborhood deduction by $equation\ 1$ [14]. We did it for all input and target dataset.

Figure 1 represents a drone from our tarining dataset after applying SURF feature extraction method. We used all the features extracted for our model.

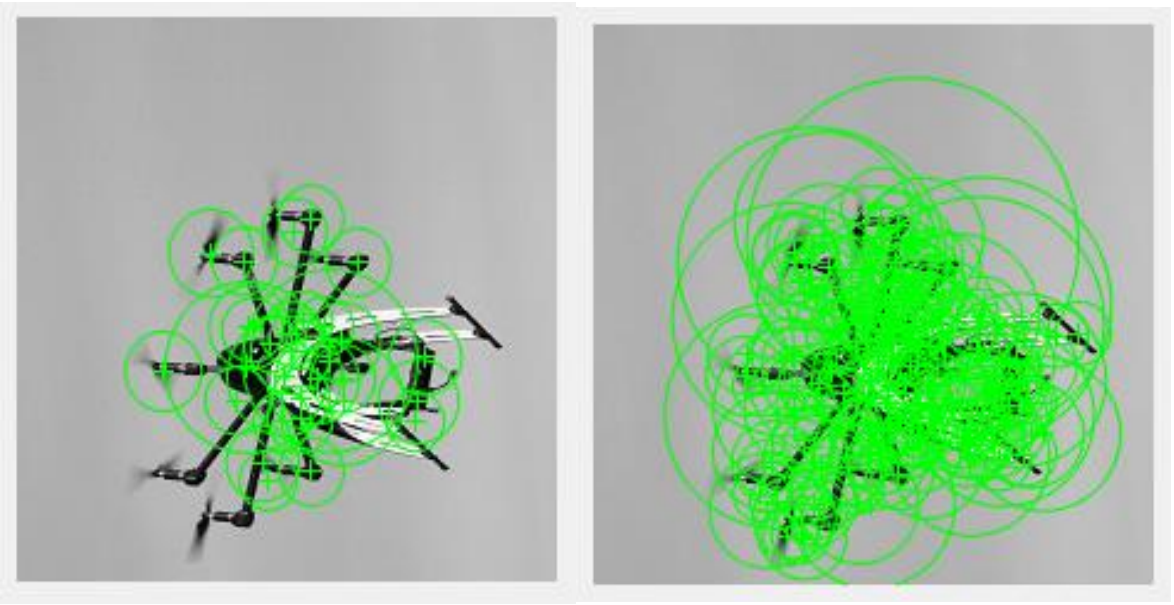


Figure 3.1 Rotated 2D drone feature detected by SURF- Strongest 30 points (at left) and all possible points (at right)

3.3 Feature Extraction by GLCM

The extracted features of the images are sent to graycomatrix function parameters and the offset is set to [2 0] [9]. Then GLCM returns an 8 by 8 matrix [10].

3.4 Preparing Dataset

We converted the 8 by 8 matrix of input images to the 1D matrix of the size of 64 by 1 and inserted as a row in the input dataset array. We do it for all input images. So, for n number of input images, we will have n × 64 sizes of the dataset. This goes for the target dataset as well. Thus, we have input dataset and target dataset.

Figure 3.2 represents the algorithm of our proposed model.

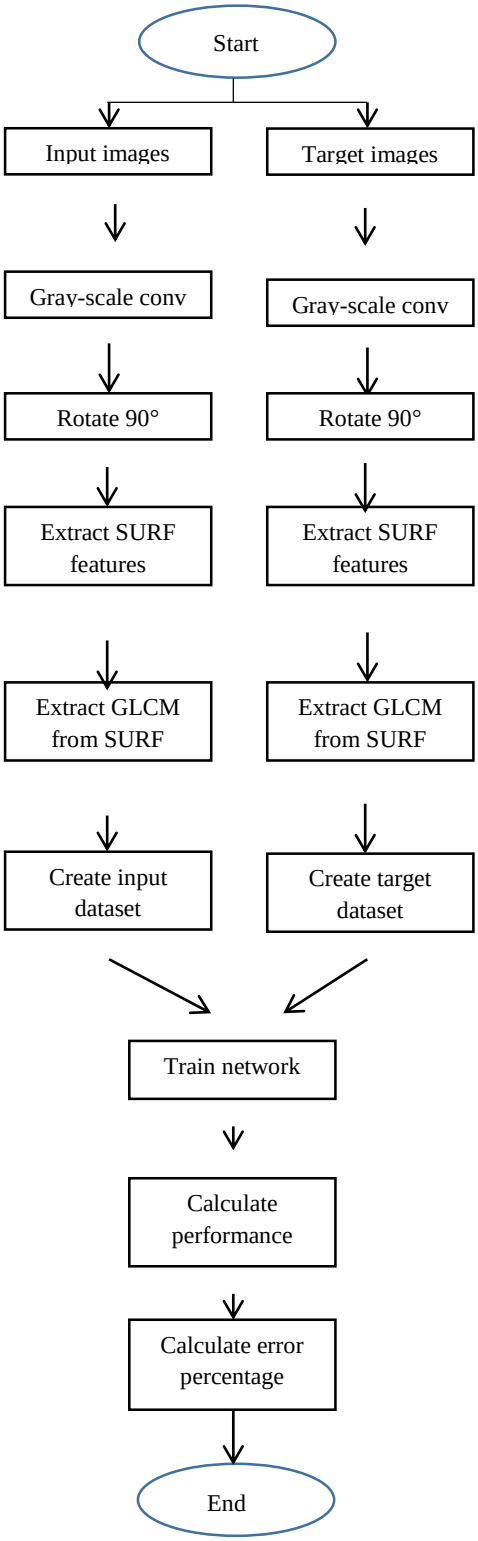


Figure 3.2 Flow Chart of Better Drone Detection Algorithm

3.5 Training Neural Network

Finally, the datasets are passed to our network which has 10 neurons or hidden layers and trained by ‘training’ function which is suitable for our model. We have used 100 epochs for faster results and it gives fine results [16].

CHAPTER 04

Experimental Setup and Result Analysis

4.1 Preparing Images for Dataset

Training Images: All the images of our image dataset have a clear sky in the background. As a result, the features that we will be getting will be of the drones alone. All the images are 3-dimensional (RGB) and the pixel of the images are 227 by 227.

Test Images: For test dataset, the images are also 3-dimensional (RGB) and of 227 by 227-pixel size.

The pixel sizes of the images have to be uniform to prepare the dataset for Neural Network that has not been trained to accept dataset of various columns. Since the images are transformed into columns of double values later, where column numbers depend on pixel sizes, so, we ensured that the images have uniform sizes to fit into our net.

4.2 Transforming Image Dimension

Images were taken randomly as input or train images, and also for test or target images. Every image was converted to a gray-scale image. All feature extraction algorithms require a 2D image, as the dataset is stored in a 2D array of double values. Moreover, it becomes easy to differentiate the image from the background and extract the required double values.

4.3 Extracting Features

For detection of any object using Neural Network, it is very important to create a dataset by extracting features of that object's image with the appropriate algorithm(s). Feature extraction algorithms can vary from object to object. Since we are detecting drones and analyzing the system performance and accuracy,

therefore, for feature extraction we have used three object detection algorithms. Those three algorithms are – SURF (Speeded-Up Robust Features), MSER (Maximally Stable External Regions) and GLCM (Gray-Level Co-occurrence Matrix). We did this in order to get a better comparison and for better understanding. We have done 4 experiments using these feature extraction algorithms and divided our experiments into 4 versions: version 3, version 4, version 5 and version 6, we will address these versions as v3, v4, v5 and v6 respectively in this paper.

4.3.1 v3-Using MSER for Feature Extraction

Maximally Stable External Regions is a feature extraction algorithm that uses blob detection method to find corresponding points of an image with two other images having a different angle of view [8]. If $Q_1..,Q_{i-1},.., Q_i$ is a sequence of a nested extremal regions ($Q_1 \subset Q_{i+1}$), then Q_{i*} is maximally stable if and only if, $q(i) = |Q_{i+\Delta} \backslash Q_{i-\Delta}|/|Q_i|$ has a local minimum at $i_{*.}$ Δ is a certain number of threshold on which the equation checks for regions that are stable. The detectMSERFeatures method returns a set of 'regions' that are described by the pixel lists [6-7]. This set of regions is used to extract feature from images which are displayed as ellipses or centroids [8].

We used the extracted matrix to collect the interest points and then formed the dataset of both train and test data.

4.3.2 v4-Using GLCM for Feature Extraction

Gray-Level Co-occurrence Matrix is generated by calculating the number of times a pixel with the gray-level intensity value at i occurs in a specific spatial relationship with the pixel j [m], where, $Q(x, y) = i$ and $Q(x + 1, y + 1) = j$ [12]. The size of GLCM matrix is determined by number oa f gray-level intensities which is by default 8. So, GLCM returns an 8 by 8 matrix of the

extracted features of the image, I, having $n \times m$ size. Here, x and y are offsets [9-11].

$$C_{\Delta x,\Delta y}(i,j) = \sum_{x=1}^{n} \sum_{y=1}^{m} \begin{cases} 1, if\ i\ and\ j\ true \\ 0, if\ i\ and\ j\ false \end{cases} \quad 4$$

We applied the algorithm to both train and test dataset and used the generated 8 by 8 matrix to convert it to the 1D matrix. Then we have created a dataset for n number of images with a matrix size of n×64 for both train and test cases.

4.3.3 v5-Using SURF for Feature Extraction

Speeded-up Robust Features is another blob detection algorithm that has three main parts of its algorithm – interest point detection, local neighborhood detection and matching [14]. It uses Gaussian smoothing method to filter integral images, which is a faster way of feature extraction than most other algorithms [14-15]. The original image is fragmented into several rectangular integral images and the summed up for faster calculations [14-15] as shown in *equation 1*.

It uses Hessian matrix to find points of interest. If p(x,y) is a point in an image and σ is the scale where the Hessian matrix H(p, σ) is to be determined then, as shown in *equation 2*.

We have applied this algorithm to both train and test cases and selected the point of interests to make our dataset.

4.3.4 v6-Using SURF and GLCM for Feature Extraction

For this version, the image was rotated to 90-degree angle after converting to the gray-scale image, as it was to be used in GLCM [9], by doing this the column number remains same i.e., 64 throughout the process. We first extracted the feature of the image with SURF and passed the resultant matrix to graycomatrix parameter. The SURF extracted image points are again extracted by GLCM for

better performance results. Then the generated 2D matrix is converted to 1D for each image. Then finally the dataset is being created for all images.

4.4 Passing Dataset for Training in Neural Net

Our net uses the train function 'Scale conjugate gradient backpropagation' because it is suitable for low memory situations and it has 10 hidden layers or neurons to learn [16]. For network train, the net, training dataset and target dataset are sent into the parameters.

4.5 Result Analysis

For results, we have considered the performance and error percentage of the network. The performance is calculated by cross-entropy per epochs; the minimum is the cross-entropy, the better is the performance. The percentage of error is calculated as,

$$percentError = \left(\sum (tind \cong yind)\right) / numel(tind) \quad 5$$

Here tind is 2D target vector indices and kind is 2D output vector indices.

4.5.1 MSER

Result as discussed before, for starters we used MSER method and did not get expected results in terms of performance, though error count was very low. Here cross entropy is much too high against its corresponding epoch, and thus disqualified as a preferred method.

Figure 3 shows the error histogram of the version where MSER feature extraction method is used and figure 4 shows the cross entropy of this model.

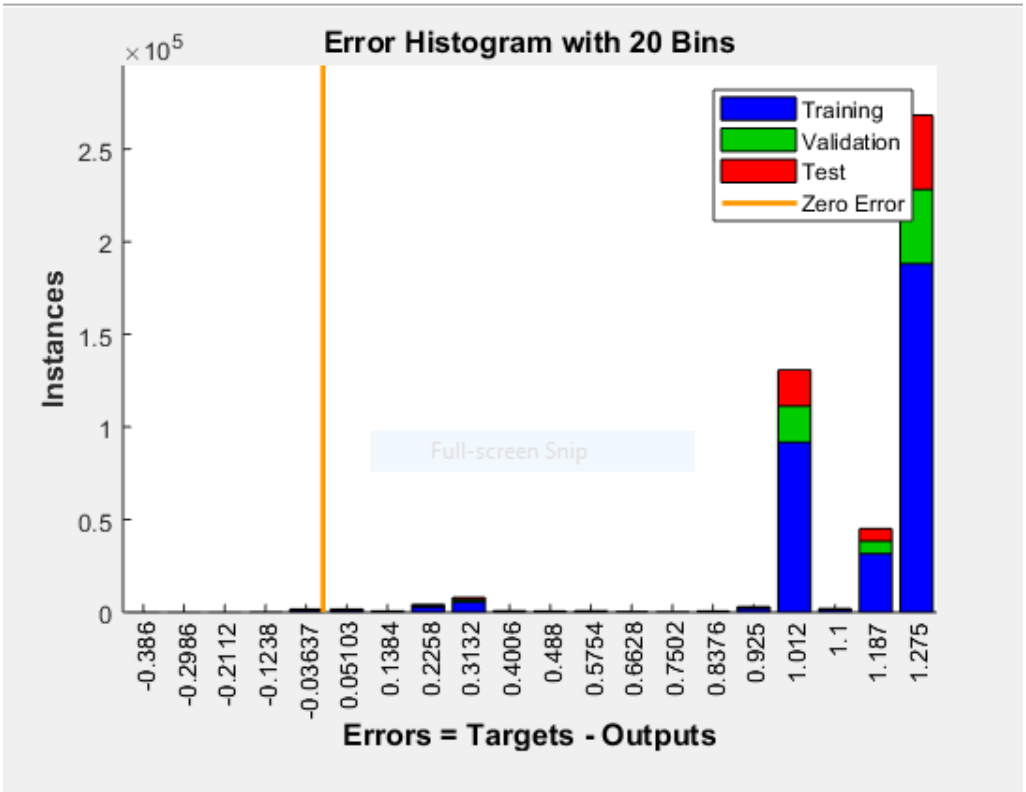
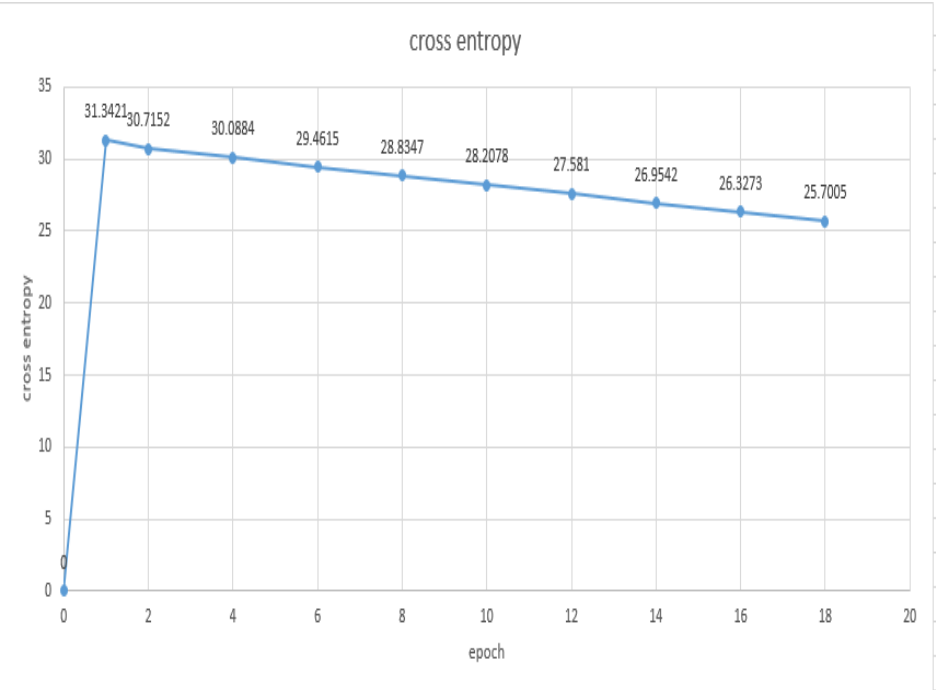
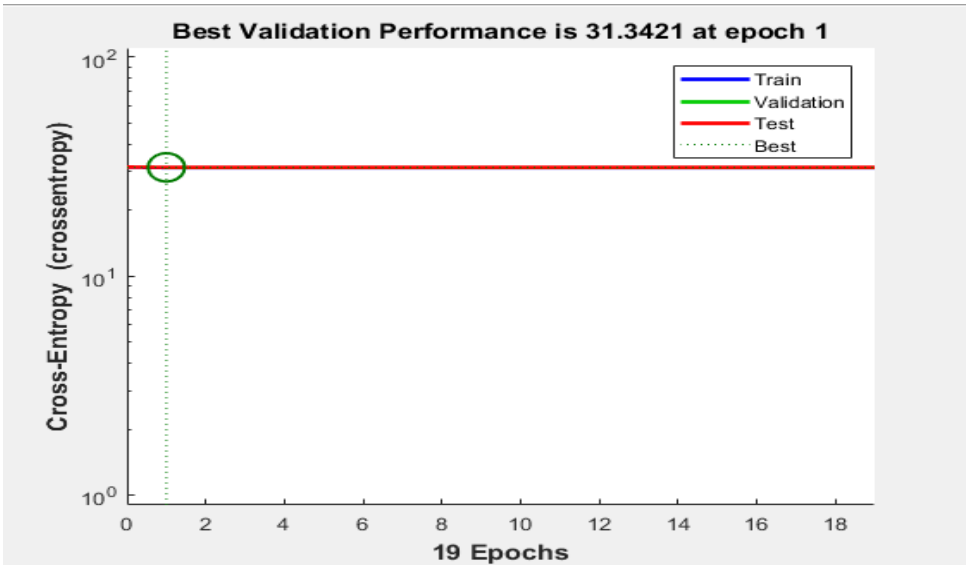


Figure 4.1 Training error Histogram



(a)



(b)

Figure 4.2: Performance graph of MSER method where cross entropy reaches as high as 31.4321 when epoch is 1. Our calculation represented in (a) and graph generated by MATLAB is shown in (b).

4.5.2 GLCM Method

Then we moved onto GLCM method. Here we got our expected performance result however here the error rate is almost 99% which makes it disqualified. This makes us understand one thing if we want a better performance we need to use GLCM method. However, only using GLCM method does not assure a good result.

Figure 4.3 represents the error histogram of this model and figure 6 represents the cross entropy vs. instances graph of this model.

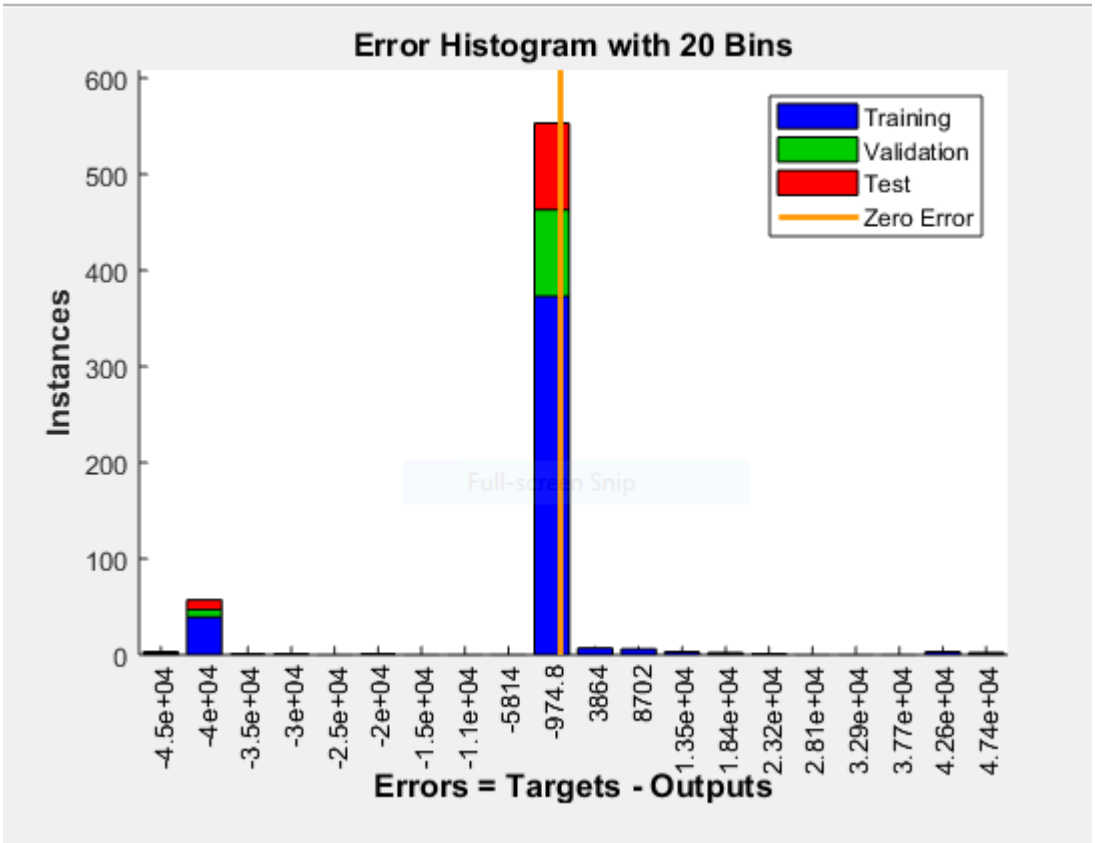
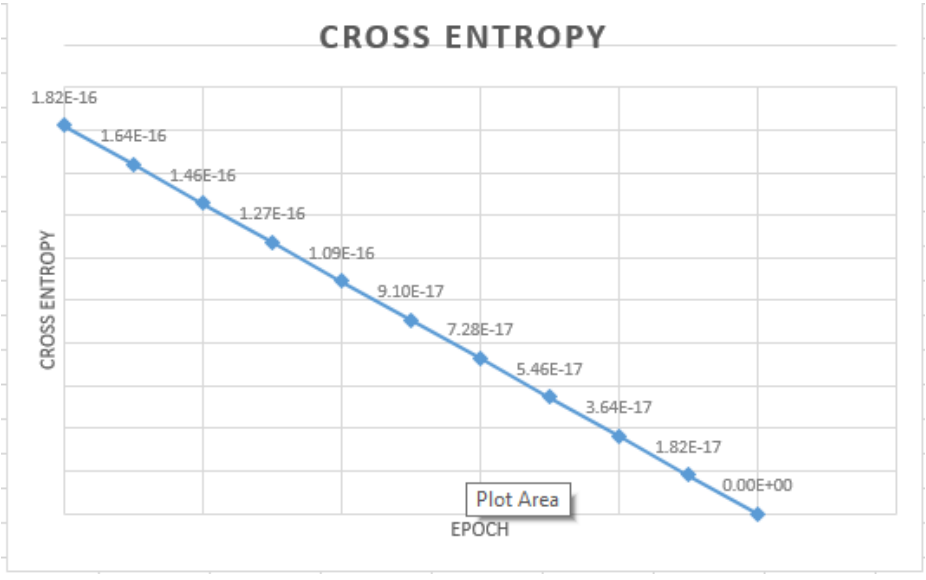
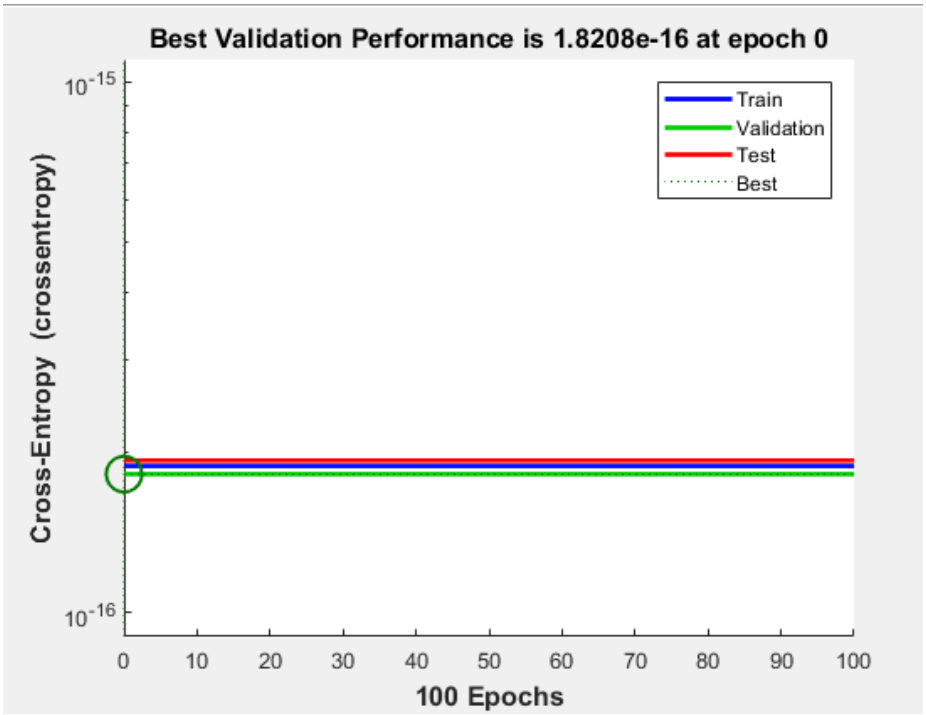


Figure 4.3 Training error Histogram



(c)



(d)

Figure 4.4 Performance graph of GLCM method where cross entropy is as low as 1.82e-16. Our calculation represented in (c) and graph generated by MATLAB is shown in (d).

4.5.3 SURF Feature

After getting disappointing error performance in GLCM method, we tried detecting drones by using SURF feature extraction method and we got extraordinary error performance. Here, error percentage is as low as 2.34 percentage. However, the cross-entropy reaches as high as 30.5. Even though we got good results in terms of error performance we cannot take it as an better method.

Figure 4.5 shows the error histogram of SURF feature extraction version and figure 8 represents the cross entropy of the model.

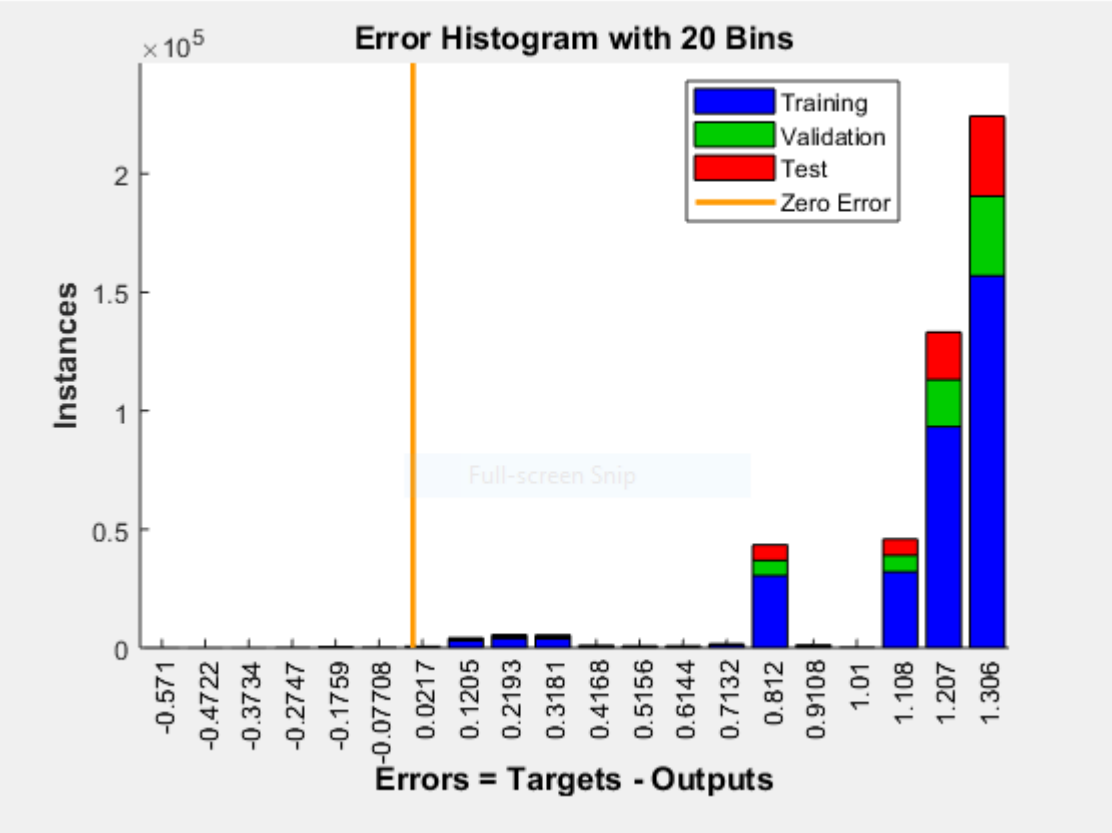
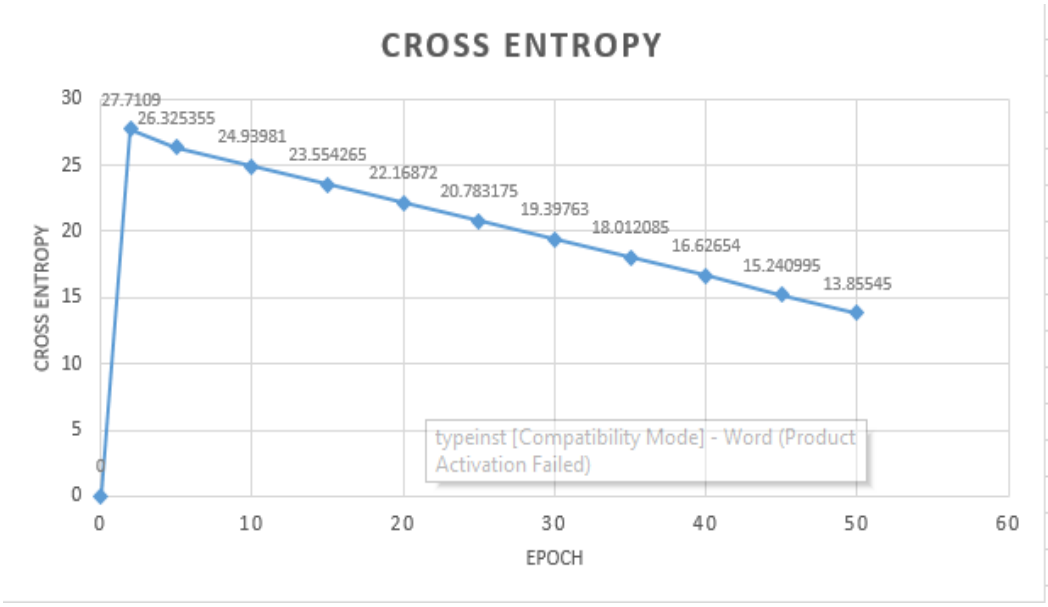
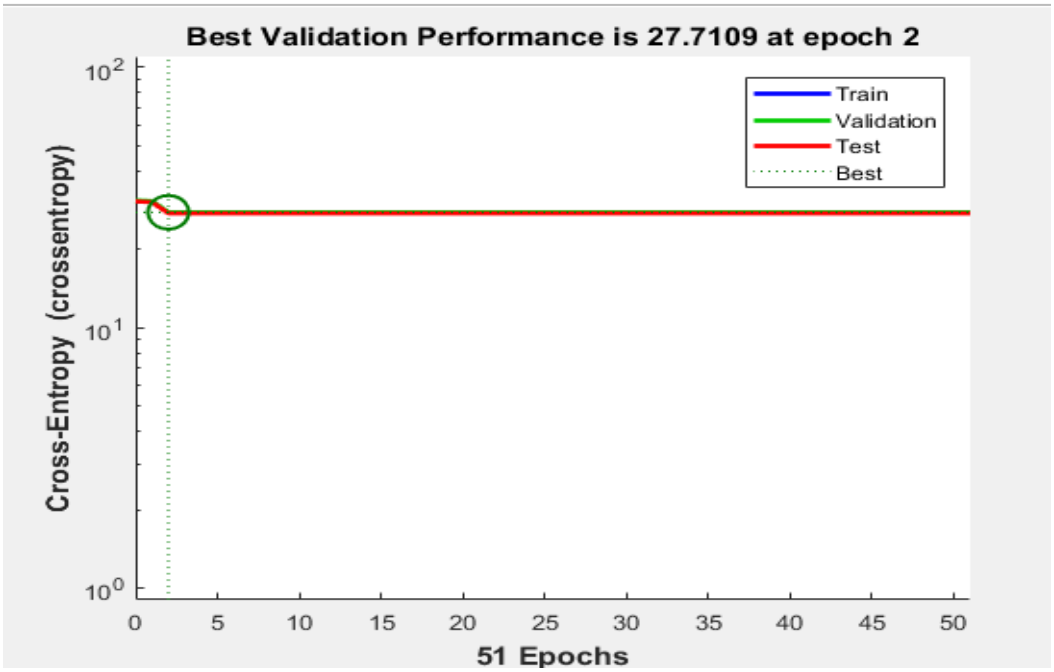


Figure 4.5 Training error Histogram



(e)



(f)

Figure 4.6 Performance graph of GLCM method where cross entropy is as high as 30. Our calculation represented in (e) and graph generated by MATLAB is shown in (f).

4.5.4 SURF Feature Extraction with GLCM Method

Finally, we got our better results where we used SURF feature extraction method for extracting the features and GLCM method to train the neural network. Here performance or cross entropy is as low as 7.34e-17 and the error performance is not higher than 35 percent. We need to detect drones with shortest possible time and do so with efficiency.

Figure 9 shows the error histogram of this model and figure 10 below shows the cross entropy relation with instances for SURF feature and GLCM feature extraction method.

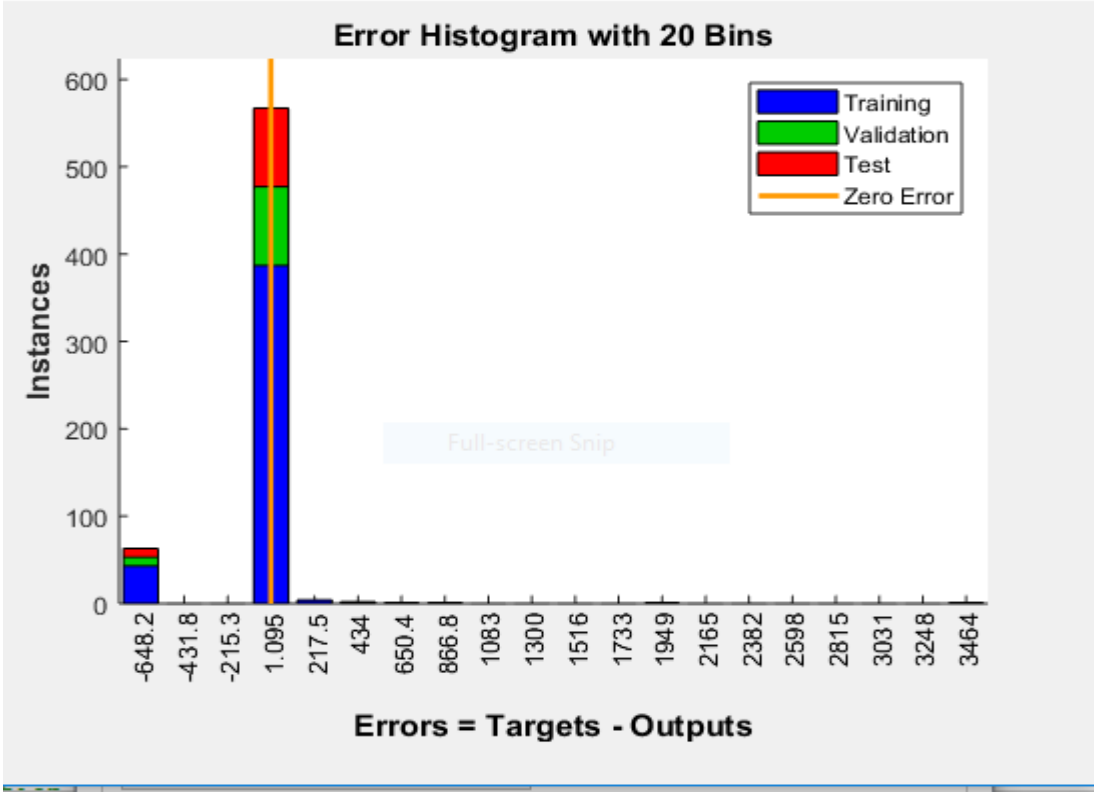
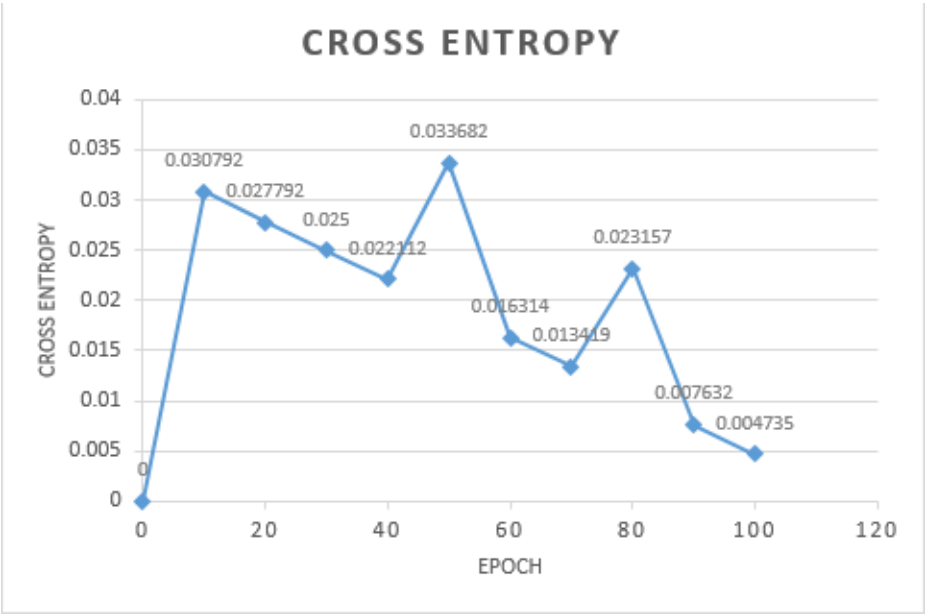
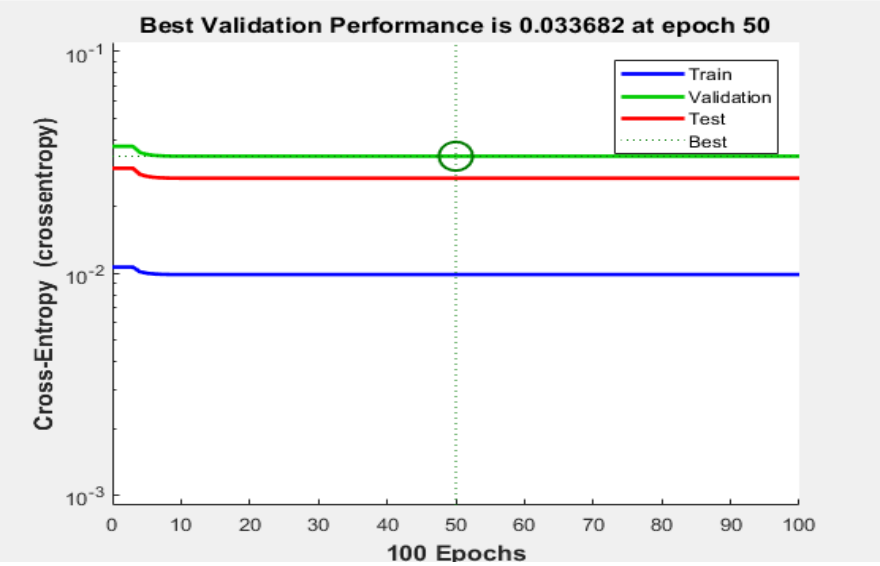


Figure 4.7 Training error Histogram



(g)



(h)

Figure 4.8 Performance graph of GLCM method where cross entropy is as low as 7.47e-17 (scale is measured). Our calculation represented in (g) and graph generated by MATLAB is shown in (h)**.**

4.5.5 Comparative Analysis

If we see the performance result and error performance of these four methods we can distinguish the better method to detect drones. With the trend analysis, we can safely assume that Surf feature with GLCM is the better way to detect drones while it is in the air. This way, we can detect the drones with a minimum amount of time and less complexity; that too with accepted error percentage rate.

Table 2 shows the comparative study of all the models with respect to their performance and percentage error. Figure 11 is the graphical display of the comparative study of the models and Figure 12 and 13 represents comparative study which are shown in 3D surface analysis and in area respectively.

Table 2: Comparative analysis at a glance

No	Version No.	Performance	Error percent
1	3	29.09	3.5
2	4	1.88e-16	89
3	5	30.5	2.34
4	6	7.34e-17	33

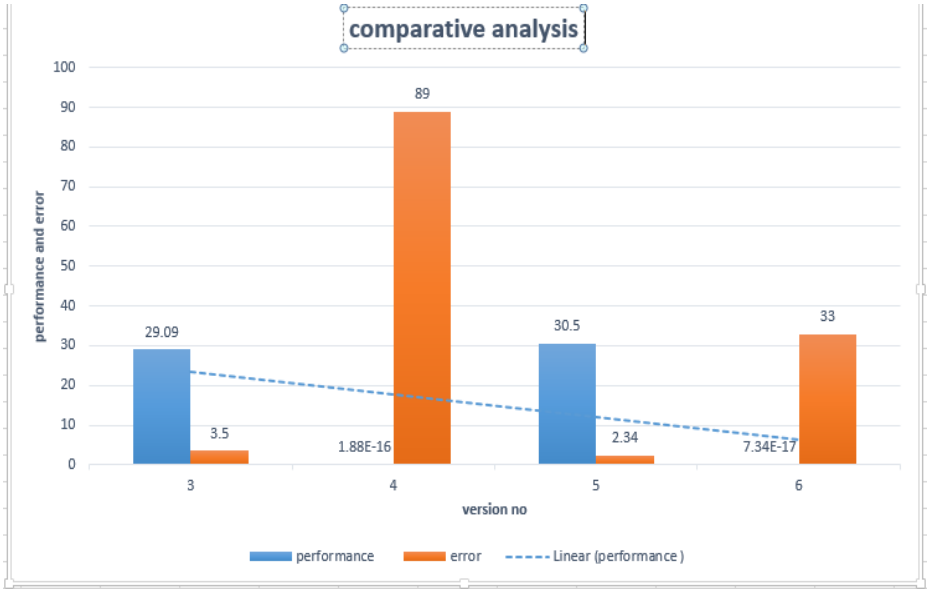


Figure 4.9 Comparative analysis with 4 methods side by side with trend analysis.

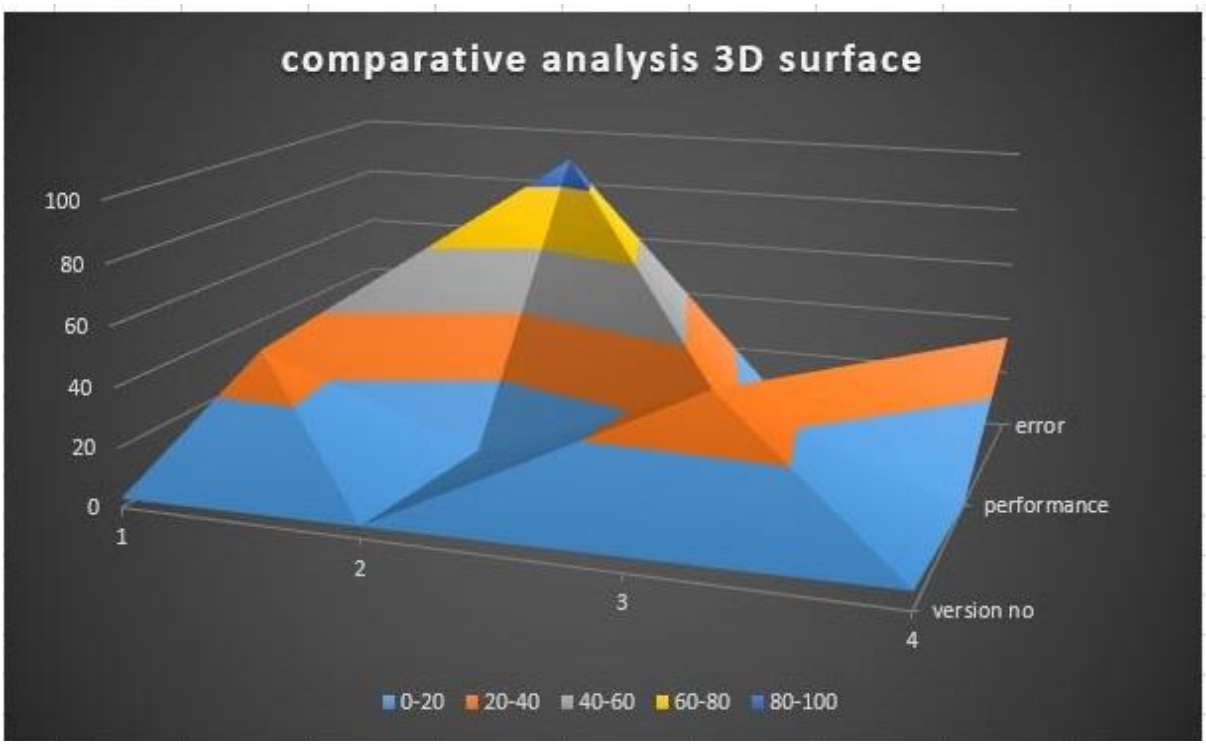


Figure 4.10 Comparative analysis trend (3D surface)**.**

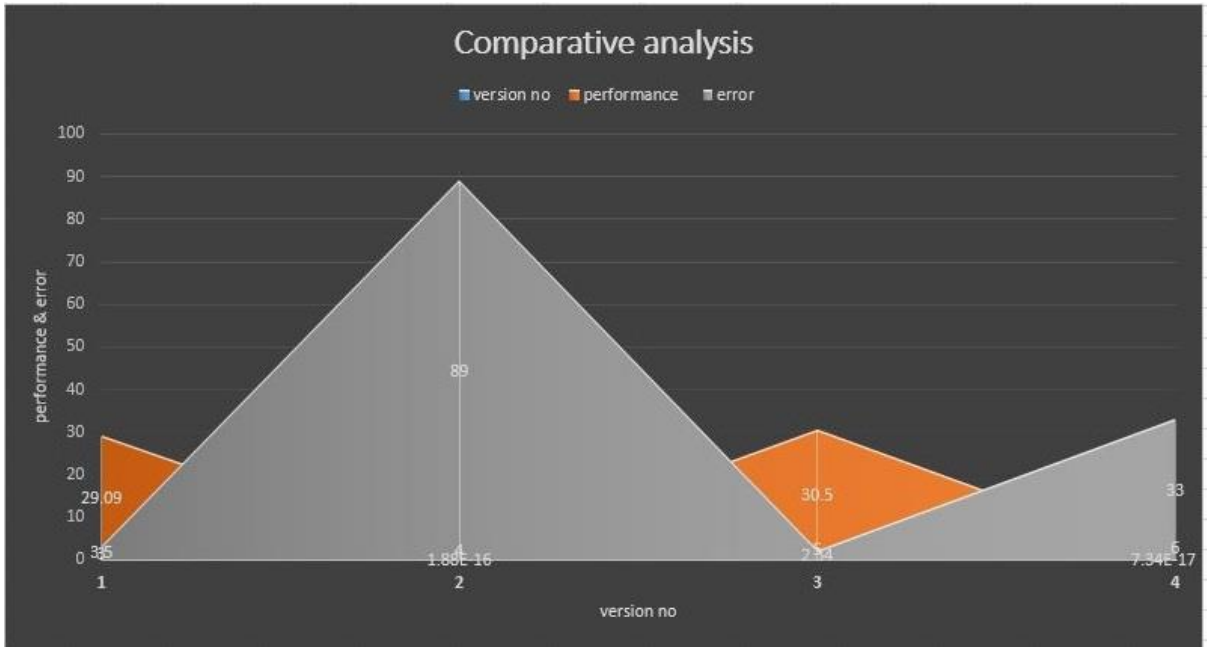


Figure 4.11 Comparative analysis trend (area)

Figure 4.11 represents our comparative analysis in a 3D wireframe structure.

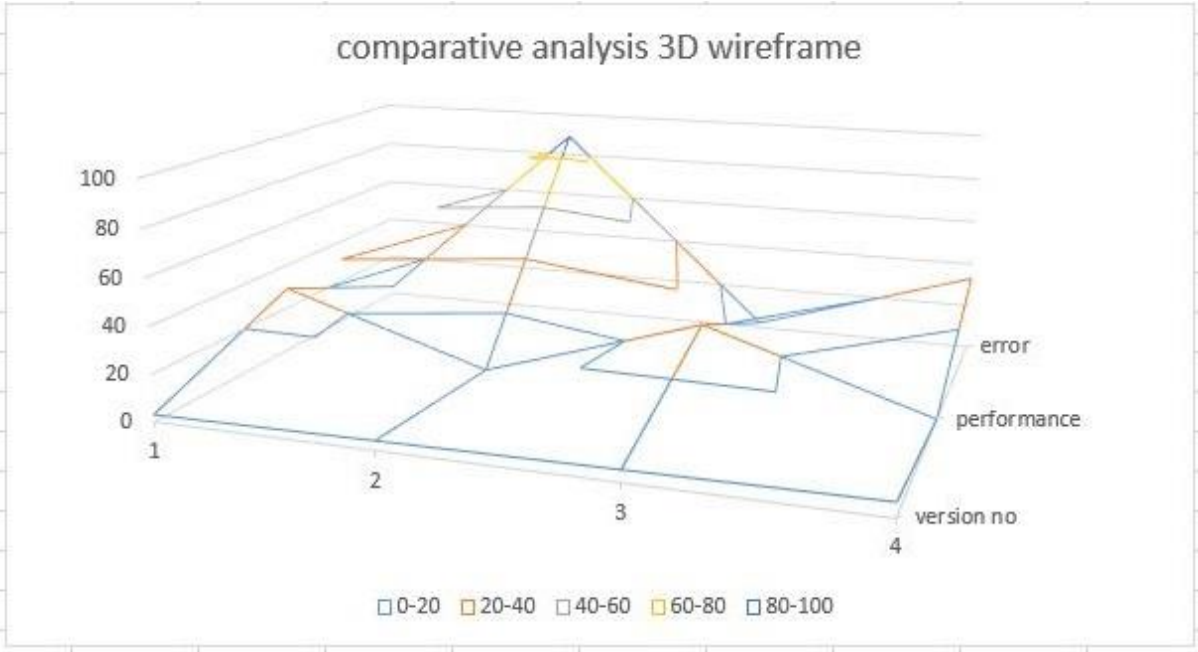


Figure 4.12 Comparative analysis trend (3D wireframe).

CHAPTER 05

CONCLUSIONS AND FUTURE WORKS

The misuse of drones is increasing day by day, especially, in protected areas giving rise to criminal activities. So, it is a prior need to take precautions by the dint of advancing technology. For that, today's world is growing into application of Neural Networking in various ways. Our aim was to apply this technology in order to make our system learn the appearances of drones, so that whenever the system captures one anomaly it can identify accurately as much as possible and as fast as it can. In the proposed model a high percent accuracy is achieved. SURF, GLCM, and MSER are used separately. Our proposed model gives us the better result both in error percentage and performance; we have shown some comparative analysis too. Thus, it can be said that this model has a great efficiency with higher accuracy.

Our future researches will be based on what better algorithms can be applied or even made to detect any kind of drones with more accuracy and least error, irrespective of the background.

REFERENCES

1. A. Samara, M. L. Menezes, and L. Galway, "Feature Extraction for Emotion Recognition and Modelling Using Neurophysiological Data," 2016 15th International Conference on Ubiquitous Computing and Communications and 2016 International Symposium on Cyberspace and Security (IUCC-CSS), Granada, Spain, 2016, pp. 138-144.
2. R. Dong, H. Meng, Z. Long and H. Zhao, "Dimensionality reduction by soft-margin support vector machine," 2017 IEEE International Conference on Agents (ICA), Beijing, China, 2017, pp. 154-156.
3. G. Yan, "Network Anomaly Traffic Detection Method Based on Support Vector Machine," 2016 International Conference on Smart City and Systems Engineering (ICSCSE), Zhangjiajie, Hunan, China, 2016, pp. 3-6.
4. M. d. Barbosa, C. d. Barbosa and A. F. Barbosa, "MuSSE: A Tool to Extract Meta-Data from Game Sprite Sheets Using Blob Detection Algorithm," 2015 14th Brazilian Symposium on Computer Games and Digital Entertainment (SBGAMES), Piauí, Brazil, 2015, pp. 61-69. doi:10.1109/SBGames.2015.22
5. F. E. H Tay, L. Lao, “Application of support vector machines in financial time series forecasting”, omega, vol. 29, no. 4, pp. 309-317, Aug. 2001.
6. E. Salahat, M. Qasaimeh, “Recent Advances in Features Extraction and Description Algorithms: A Comprehensive Survey,” 2017, arXiv:1703.06376v1 [cs.CV], https://arxiv.org/pdf/1703.06376.pdf
7. S. Cunzhao, W. Chunheng, X. Baihua, G. Song, "Scene Text Detection Using Graph Model Built Upon Maximally Stable Extremal Regions," 2013, *Pattern Recognition Letters*. 34 (2): 107–116.
8. https://www.mathworks.com/help/vision/ref/detectmserfeatures.html, MathWorks, v: R2018a, 2018.

9. https://www.mathworks.com/help/images/ref/graycomatrix.html, MathWorks, v: R2018a, 2018.
10. http://www.mathworks.com/help/images/create-a-gray-level-co-occurrence-matrix.html, MathWorks, v: R2018a, 2018.
11. https://www.mathworks.com/matlabcentral/fileexchange/22187-glcm-texture-features, MathWorks, v: R2018a, 2018.
12. P. Cosman, "Gray-Level Co-occurrence Matrices (GLCMs)," http://www.code.ucsd.edu/pcosman/glcm.pdf.
13. R. Haralick, K. Shanmugam, I. Dinstein, "Textural Features for Image Classification," 1973, *IEEE Transactions on Systems, Man, and Cybernetics*. SMC-3 (6): 610–621.
14. P. M. Panchal, S. R. Panchal, S. K. Shah, "A Comparison of SIFT and SURF, " 2013, International Journal of Innovative Research in Computer and Communication Engineering Vol. 1, Issue 2.
15. https://www.mathworks.com/help/vision/ref/detectsurffeatures.html, MathWorks, v: R2018a, 2018.
16. https://www.mathworks.com/help/nnet/gs/classify-patterns-with-a-neural-network.html, MathWorks, v: R2018a, 2018
17. https://www.mathworks.com/help/nnet/ref/crossentropy.html, MathWorks, v: R2018a, 2018.