International Journal of Electronics Communication and Computer Engineering
Volume 4, Issue 2, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

Genetic Algorithm: An Enhanced Feature Selection Tool for Face Detection

K. Mohan, Dr. K. V. Ramanaiah, Dr. S. A. K. Jilani

Abstract **– In real time, the face detection is a big task and various techniques have been proposed over the past decade. In general a large number of features are required to be selected for training purposes of face detection system. Often some of these features are irrelevant and does not contribute directly to the face detection algorithm. This creates unnecessary computation and usage of large memory space. In this paper we propose to enlarge the features search space by enriching it with more types of features with the help of Genetic Algorithm (GA) and can be used in real time to select the best feature of the image, with in the Adaboost framework, to provide better classifiers with a shorter training time. The GA carries out an evolutionary search over possible features search space which results in a higher number of feature types and sets selected in very less time. Experiments on a set of images from Bio identification database proved that by using GA to search on large number of feature types and sets, GA technique is able to obtain cascade of classifiers for a face detection system that can give higher detection rates, lower false positive rates and less training time.**

Keywords **– Genetic Algorithm, Cascade of Classifiers, Bio Identifications, Features, Memory.**

I. INTRODUCTION

To detect human faces in images in real-time is a real challenging problem. Viola and Jones were the first who developed a real-time frontal face detector by introducing boosted cascade of simple features that achieves comparable detection and false positive rates to actual state-of-the-art systems. Many researchers have proposed to enhance the idea of boosting simple weak classifiers. Lienhart et al. showed that by extending the basic feature types and sets, detectors with lower error rates are produced. However, extension of feature types automatically leads to much higher number of feature sets and expand the search space thus increases the training times. Recapitulating the research that is based on the publication of Viola and Jones we can see that there are mainly two problems to deal with: (1) Extending the feature sets and being able to search over these very large sets in reasonable time. (2) The best feature solution sets are not known in advance. To overcome these problems, we use GA in combination with Adaboost to search over a large number of possible features. Our goal is to find better cascade of classifiers by using a very large feature sets in lesser time, and that achieve comparable or even better classification results compared to the cascade of classifiers that are trained exhaustively over a small feature sets. The use of Evolutionary Algorithms in the field of image processing, especially automatic learning of features for object detection has received growing interest. Treptow and Zell showed an Evolutionary Algorithm can be used within Adaboost framework in single stage classifiers to find features which provide better classifiers for object detections such as faces and balls. In 2006, J. S. Jang and J. H. Kim introduced the employment of Evolutionary Pruning in cascaded structure of classifiers which has a purpose to reduce the number of weak classifiers found before by Adaboost for each stage of cascade training. However, this approach was aimed to focus more on increasing the performance of face detection speed by reducing the number of weak classifiers in already built cascade while ignoring the cascade training time.

Our approach is by focusing on reducing the computational training time to build 15 stages cascade of boosted classifiers while having similar or better cascade performance. In order to achieve that goal, we enrich the possible feature solutions by adding seven new types of features. These additional features will increase the size of the search space thus the cascade of classifiers training time. GA is then implemented into Adaboost framework to select good features sets which represent sets of strong classifiers for each stage of cascade training. The cascade built consists of the combination of eight basic features and seven new feature types.

The paper is organized as follows: In the next Section, Adaboost learning procedure is introduced. Section III describes cascade of boosted classifiers. In Section IV, application of GA into Adaboost framework to build cascade of classifiers using large number of feature types is introduced. This includes the characteristics of seven new feature types. Section V show the result of performance of the cascade trained using GA compare with cascade trained exhaustively in terms of hit rates, missed rates, false positive rate and training time. The experiments were done by using open source software, Intel Open CV, as done in [2]. Section VI summarizes and concludes our proposed work and points out perspectives for further future research.

II. ADABOOST LEARNING OF OBJECT DETECTORS

Viola and Jones developed a reliable method to detect objects such as faces in images in real-time. An object that has to be detected is described by a combination of a set of simple Haar-wavelet like features shown in Fig. 1. The sums of pixels in the white boxes are subtracted from the sum of pixels in the black areas. The advantage of using these simple features is that they can be calculated very quickly by using “integral image”.

An integral image II over an image I is defined as follows

$$II(x,y) = \sum_{x' \le x, y' \le y} I(x',y') \qquad (1)$$

International Journal of Electronics Communication and Computer Engineering
Volume 4, Issue 2, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

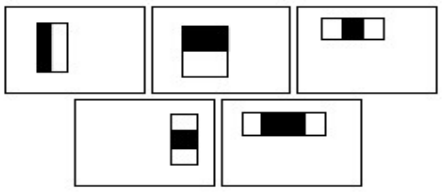

Fig.1. Five different basic types of rectangle features within their sub window of 24x24 pixels. These five types of features are the initial features used to train cascade of classifiers exhaustively.

In [1] also, it is shown that every rectangular sum within an image can be computed with the use of an integral image by four array references. A classifier has to be trained from a number of available discriminating features within a specific sub window in order to detect an object. The possible positions and scales of the five different feature types as shown in Fig. 1 produce about 90,000 possible alternative features within a sub window size of 24x24 pixels. This number exceeds largely the number of pixels itself. Therefore, a small set of features which best describe the object to be detected, has to be selected. Adaboost is a technique that initially selects good classification functions, such that a final "strong classifier" will be formed, which is in fact, a linear combination of all weak classifiers. In the general context of learning features, each weak classifier h j (x) consists of one single feature f j :

$$h_j(x) = \begin{cases} 1 : p_j f_j(x) < p_j \vartheta_j \\ 0 : otherwise \end{cases} \qquad (2)$$

Where V_j is a threshold and p_j a parity to indicate the direction of the inequality. The description of Adaboost algorithm to select a predefined number of good features given a training set of positive and negative example images is shown in [1].

The Adaboost algorithm iterates over a number of T rounds. In each iteration, the space of all possible features is searched exhaustively to train weak classifiers that consist of one single feature. During this training, the threshold V_j must be determined for the feature value to discriminate between positive and negative examples. Therefore, for each possible feature and given training set, the weak learner determines two optimal values (thresholds), such that no training sample is misclassified. For each weak classifier $h_j(x)$, the error value ε_j will be calculated using misclassification rate of all positive and negative training images. It gives each feature $h_j(x)$ trained with its respective error value ε_j which is between 0 and 1. The best feature $h_t(x)$ found with the lowest error rate ε_j will be selected as the weak classifier for this 1/T iteration. After the best weak classifier is selected, concerning all training examples are reweighted and normalized to concentrate in the next round particularly on those examples that were not correctly classified. At the end, the resulting strong classifier is a weighted linear combination of all T weak classifiers.

III. Cascade of Boosted Classifiers

This section describes an algorithm for constructing a cascade of classifiers which drastically reduces the computational time. The main idea is to build a set of cascade boosted classifiers which are smaller, but more efficient, that will reject most of the negative subwindows while detecting almost all positive instances. Input for the cascade is the collection of all subwindows also called scanning windows. They are first passed through the first stage in which all sub-windows will be classified as faces or non faces. The negative results will be discarded while the remaining positive sub-windows will trigger the evaluation of the next stage classifier. The sub-windows that reach and pass the last layer are classified as faces (See Fig. 2). Every layer actually consists of only a small number of features. In the early stages, with only small number of selected features it is possible to determine the existence of a non-face. On the other hand, determining the presence of a face usually needs more features. The trained cascade boosted classifiers usually have an increasing number of features in each consecutive layer until its last layer and therefore became increasingly more complex.

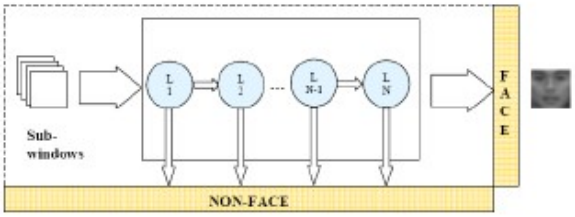

Fig.2. Cascade from simple to complex classifiers with N layers.

During the training of the cascade of boosted classifiers, the number of features per layer or stage was driven through a "trial and error" process. In this process, the number of features was increased until a significant reduction in the false positive rate could be achieved. In our case, each stage was trained to eliminate 50% of the non-face patterns while falsely eliminating only 0.005% of the frontal face patterns; 15 stages were trained and a false alarm rate about $0.5^{15} \cong 3x10^{-5}$ and a hit rate about $0.995^{15} \cong 0.93$ were expected. More features were added until the false positive rate on the each stage achieved the desired rate while still maintaining a high detection rate. At each training stage, the false positive images from previous stages are added to the sets of negative or non-faces images and this set of images is used as negative images in the next stages training.

IV. Genetic Algorithm for Feature Selections

Genetic Algorithms (GAs) are a family of computational models inspired by natural evolution. GAs comprises a subset of evolution-based optimization techniques focusing on the application of selection, mutation, and recombination or crossover to a population of competing problem solutions. In our paper, the chromosome of GA

represents the specific type and location of one single feature in the sub window of 24x24 pixels. Each chromosome has six genes. The first five genes are integer types which consist of:

- *type* : type of feature
- *x* : coordinate x in sub-window
- *y* : coordinate y in sub-window
- *dx* : width of feature
- *dy* : height of feature

The sixth gene contains fitness value between 0 and 1. As described previously in Adaboost, every trained feature or weak classifier produce an error value ε_j and the feature with the lowest ε_j will be selected. Therefore, fitness function chosen here is 1- ε_j where *i* is the number between 1 and population size N. With this fitness function, the higher fitness value indicates the lower error value ε_j. To increase the possibility of selecting good features, we increase the number of feature types. The existing three feature types in Open CV as shown in Fig. 4 are added into the total feature sets. In addition to that, we add our new seven feature types which should increase the search space and possibilities of getting better cascade of classifiers with higher performance.

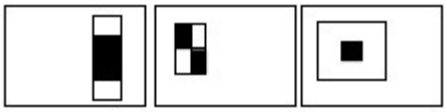

Fig. 3. The existing three types of features within their sub window of 24x24 pixels. These feature sets are added in training of cascade of classifiers with GA search

In Fig. 4, features *a, b, c* and *d* with the style of an L-shape and an inverse L-shape should have the ability to distinguish the image of face and non-face based on the pattern of the side of the face specifically on the left and right foreheads, temples and jaw lines. Feature *e* is the inverse of the middle feature shown in Fig. 2. And lastly feature *f* should distinguish the pattern of both eyes region with forehead and feature *g* should make the difference between face images and non-face images based on the pattern of eyes location. With all these, the total feature types equaled to 15. As a result, while more valuable and better types of features might be created as the potential sets of good feature solutions, the search space of all of these feature types increased dramatically. Therefore GA is used to select the features and to avoid the exhaustive search and high computational time.

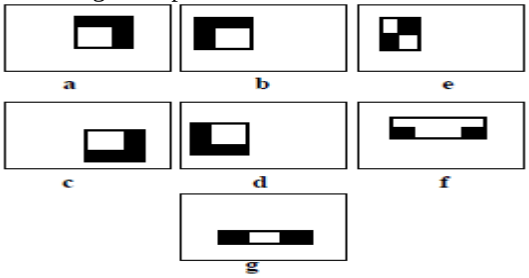

Fig.4. The newly proposed seven types of features within their sub window of 24x24 pixels. These feature sets are proposed and added in training of cascade of classifiers with GA search.

The parameters of Genetic Algorithm are shown in Table 1. In the first generation of GA, all chromosomes are randomly generated and evaluated to determine their fitness value. Ranking Scheme selection is used when all chromosomes are ranked based on their fitness values. The first chromosome will have the highest fitness value while the last one will have the lowest one. In each generation, half of the total populations will be selected and put into a mating pool and will become parent chromosomes.

The two-point standard uniform crossover is applied here. Based on the crossover probability rate, p_{co}, where $p_{co} \in \{0..1\}$, two different chromosomes are chosen from the mating pool. Two random gene positions, *m* and *n* such that $m, n \in \{1..5\}$ and $m \neq n$, are chosen randomly. Then, the genes at position *m* and *n* are crossover within these two parents to produce new children. In mutation process, mutation probability rate p_{mt} where $p_{mt} \in \{0..1\}$, is used. Based on this rate, mutation process will take place on the chromosome chosen from the mating pool. Once mutation occurs, a single gene position is selected randomly. If the first gene *type* is selected, the new and different type of feature will be selected randomly between 1 and 15. While in the other cases such as genes *x, y, dx* or *dy*, its value will be added with an integer value randomly chosen between -2 and +2. All parent chromosomes that have undergone crossover and mutation process will produce new children chromosomes and they are re-evaluated to determine their fitness values, 1-ε_i. The new chromosomes with high fitness values will have a better chance to be selected and inserted into GA population. For the new chromosomes that become invalid or no longer feasible, for example the sum of their x-coordinate and their width exceed the allowed width of the sub window, here is 24x24, then their fitness value will be assigned with zero, in other word, these features are not the good features and they shall be discarded. While the chromosomes with modest or average fitness values will not have a good opportunity to be selected for the next generation.

The 200-GA population size chosen is the result of a trial and error process in which a trade-off is made between speed and error rate. The size of a population should be sufficiently large to create sufficient diversity covering the possible solution space. Other parameters of GA are the probabilities for crossover and mutation operators. In their paper, Treptow and Zell had preferred 20% of crossover rate and 80% of mutation rate. Since they used EA to select about 200 features in only a single stage of classifiers, the ratio used seems to be reasonable.

Table 1: Parameters used in Genetic Algorithm

Genetic algorithm parameters	
Population size	150
Crossover type	Two points in standard uniform
Cross over rate	o.92
Mutation type	Single generation
Mutation rate	0.13

In our case, crossover rate was 90% and mutation rate was 10%. This is due to the fact that the training of 15

stages cascade of classifiers is slightly different from the training of a single stage classifiers as described in Section 3 even though both use Adaboost algorithm. At each different stage in the cascade training, GA will select features from slightly different search spaces. This is due to the new generated images samples incorrectly classified (false positive) in the previous stages that are added into the training sets. In order to avoid early local optima or to loose good features solutions, the exploration of feature search space should have higher priority. So, to train cascade of classifiers which involved 15 stages, as in our case, the ratio 9:1 crossover-mutation rate seems to be logical and reasonable.

V. Expected Experiment Results with Genetic Algorithm

In this Section, we compare the performance of the 15 stages cascade of classifiers that has been built in two different techniques: (1) Adaboost with exhaustive feature selections (Exhaustive) and (2) Adaboost with GA to select features (GA). Exhaustive search over only five basic feature types while GA search over a larger set of 15 feature types. The training sets used consist of 7,000 positive images and 3,000 negative images from various sources (See Fig. 5). The dimensions of these gray value images are 24x24 pixels and they are different one from another.

Fig.5. Examples of faces and non-faces images in the training set

The test set consists of face dataset images from BioID. This dataset contains 1,200 face images with different people, gender, face expression, size and location, level of illumination and also appearance of non-face objects. The examples are shown in Fig. 7. All images in the training and testing datasets are different from one to another.

Fig.6. Examples of images containing faces with various conditions used in the BioID test set.

The total trainings were stopped after a set of 15 cascaded stages has been built. The population size is 200, all chromosomes are initialized randomly, crossover rate is 0.9 and mutation rate is set to 0.1. GA is converged and will stop if no better single feature found within the next 50 consecutive generations. In case of no convergence, GA will stop as well if it reaches the maximum number of generations which is set to 200. Experiments were carried out on an Intel Pentium IV 3.0 GHz processor. GA was run 10 times and the average results are taken. In Table 2, we can see that GA performs 3.7 times faster than Exhaustive. The comparison of computational training time between Exhaustive and GA in the ten experiments is shown in Fig. 7.

Table 2: Training time taken to build 15 stages cascade of classifiers and their number of features selected.

Name of the Algorithm	Average training time in sec	Average time to select one feature	Average no. of features in cascade
Exhaustive feature selection 5F	133554	334	395
Genetic algorithm (GA) 15F	35762	57.4	625

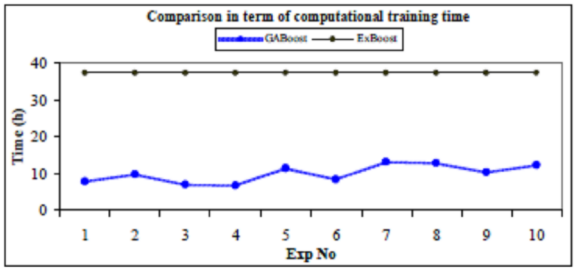

Fig.7. Comparison of computational training time between Exhaustive and GA in ten experiments.

Table 2 also show the total number of features found by both algorithms. We can see that GA selected more features compared to Exhaustive. However, with less training time, selection of a single feature just require 139.9 seconds in GA while in Exhaustive, 336.4 seconds need to find a single feature. This showed that selection of a single feature is 5.9 times faster in GA than in Exhaustive.

Table 3: Comparison of hit rates and false positive rate performed by the cascades of classifiers built by Exhaustive and GA.

Algorithm	Average hit rate (%)	False positive detection (%)
Exhaustive 5F	90.5	63.1
GA 15F	89.73	60.46
GA 15F (high-hit)	94.35	62.31
GA 15F (low-hit)	85.71	49.89

Fig.8. The examples of the test images show faces are detected. The top three images however show false positive detections in single image while in the bottom three images, detection of faces are done perfectly.

International Journal of Electronics Communication and Computer Engineering
Volume 4, Issue 2, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

Table 3 presents the average hit rate of GA (89.73%) is slightly superior to the hit rate of Exhaustive (90.5%). From the experiments also, the best hit rate achieved by GA is 94.35% while the worst hit rate is 85.71%. GA also performs better in false positive rates. False positive rate is calculated based on the sum all false positives rectangles detected divided by the total number of detection rectangles in the whole test sets. The examples are shown if Fig. 8. In our experiments, the average false positive rate achieved by GA is 60.46% that is lower than false positive rate of Exhaustive (63.1%). The hit rates and false positive rates of the ten experiments of GA are shown in Fig. 9.

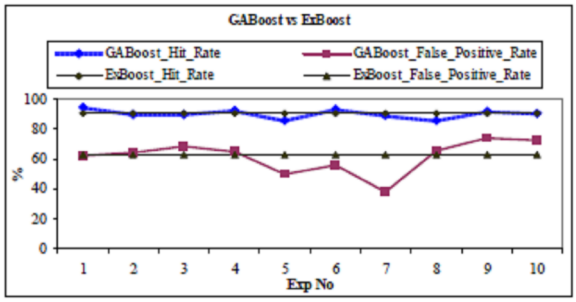

Fig.9. Hit rates and False Positive rates in GA and Exhaustive.

In Table 4, the results of the seven newly proposed feature types are examined. For each feature type *a*, *b*, *c*, *d*, *e*, *f* and *g* in Fig. 5, the average number of times that they are selected during cascade training using GABoost is shown.

These seven feature types contribute about 21% of the total features selected by GA in various stages. Among them, feature types *a*, *e* and *f* have important roles in the cascade training as they are selected many times. These three feature types contributes about 76.17% from the total seven new feature types proposed and about 15.67% from the total features selected in the cascades of boosted classifiers built by GA.

Feature types *b* and *c* have little impact on the cascades as they were selected only 15.4 and 11.5 times.

Table 4: Details of the average number of seven new feature types selected by GA

Feature type	a	b	c	d	e	f	g
Average no. of feature selection	28.9	15.4	11.5	2.3	37.9	31.8	4.6
New features	128.6						
Percentage from total features	20.52						

These numbers represent only 4.26% of the total features selected. Table 4 also shows that feature types *d* and *g* cannot be considered as good feature types since they are rarely selected in the cascade training. Fig.9. shows the distributions of number of feature types selected among these seven new feature types.

VI. Conclusion

In this paper, we have extended the work of Treptow and Zell by implementing GA inside the Adaboost framework to select features. Fifteen stages of cascaded classifiers are set up rather than building just a single stage classifier only. Seven new feature types were proposed in order to increase the quality of feature solutions. As a consequence, the feature solutions sets become bigger and we have shown how GA can be used to overcome the problem of feature selections in this huge search space. Training time was drastically reduced and the cascade of boosted classifiers trained using GA has achieved a better hit rate and lower false positive rate compared to the original exhaustive technique. This lower false positive rate might happen due to the higher number and more variety of features types selected by GA. We have shown also that the best cascade of boosted classifiers obtained by GA has outperformed the original cascade built exhaustively in computational training time, hit rate and false positive rate. On the other hand, the seven new feature types have shown different level of importance in this training. Three of these new feature types have shown very significant roles in the training of cascade of boosted classifiers. However, we believe that the performance of these seven feature types could be different if another set of training samples is used. Nevertheless, they are many directions for further research in this field. Other search algorithms such as Memetic Algorithm [12] may be used to replace GA to select features in cascade training.

In GA itself, the dynamic rates of crossover and mutation could also be implemented and analyzed. This dynamic rate had been used as a tuning tool in one part of the research. On the other hand, since the speed of detection rate was not addressed in this paper and by looking at the number of features selected, it seems that the performance of the face detector will be slightly slower. Thus, the improvement of the cascaded classifiers built by GA with very large feature types could be done by using the Evolutionary Pruning, an approach introduced recently in 2006 by J. S. Jang and J. H. Kim . Its purpose is to reduce the number of weak classifiers while maintaining the performance achieved by the cascade of classifiers.

Acknowledgment

I here acknowledge my sincere gratitude to Dr. K. V. Ramanaiah, Ph. D., for his constant motivation and valuable guidance to publish this paper throughout the research.

I wish to place on record my deep sense of gratitude to Dr. S .A. K. Jilani, Ph. D., without whose inspiration and able to guidance and whole hearted support, this paper would never have seen the light of the day.

REFERENCES

[1] P. Viola, M. Jones, Rapid object detection using a boosted cascade of simple features, *IEEE Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, December 11-13, Hawaii, USA, 2001.

[2] R. Lienhart, A. Kuranov and V. Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. *In DAGM'03, 25th Pattern Recognition Symposium*, pages 297-304, Germany, 2003.

[3] A. Treptow & A. Zell, Combining Adaboost Learning and Evolutionary Search to select Features for Real-Time Object Detection, *Proceedings Of the Congress on Evolutionary Computational CEC 2004, Vol. 2, 2107-2113*, San Diego, USA, 2004.

[4]. Lin, C. and Ling, W., "Automatic facial feature extraction by genetic algorithms," IEEE Transactions on Image Processing, Vol. 8, No. 6, pp. 834-845, 1999.

[5] K. Sung and T. Poggio. Example-based Learning For View-based Face Detection, *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 20, page 39-51, 1998.

[6] H. Schneiderman and T. Kanabe. A Statistical method for object detection applied to faces and cars, *International Conference on Computer Vision and Pattern Recognition*, page 1746-1759, 2000.

[7]. Chellappa, R., Wilson, C.L. and Sirohey, S., "Human and machine recognition of faces: a survey," Proceedings of the IEEE, Vol. 83, No. 5, pp. 705 -741, 1995.

[8] S.Z. Li, Z.Q. Zhang, H. Shum and H. J. Zhan. Floatboost Learning for Classification, *16th Annual Conference on Neural Information Processing Systems (NIPS)*, 2002.

[9] J. S. Jang and J. H. Kim. Evolutionary Prunning for Fast and Robust Face Detection, *IEEE Congress on Evolutionary Computation CEC 2006, pages 1293- 1299*, Vancouver, Canada, July 2006.

[10] Charles L. Karr and L. Michael Freeman, "Industrial Networks Fuzzy Logic and Genetic Algorithms" Prentice-Hall of India Private Limited, 2003.

[11] T. L. Seng, M. Khalid and R. Yusof. Tuning of A Neuro-Fuzzy Controller by Genetic Algorithm With An Application to A Coupled-Tank Liquid-Level Control System, *International Journal of Engineering Applications on Artificial Intelligence, Vol. 11, pages 517-529*, 1998.

[12] Areibi, S., Moussa, M., and Abdullah, H., A Comparison of Genetic/Memetic Algorithms and Other Heuristic Search Techniques, *International Conference on Artificial Intelligence*, pages 660-666, Las Vegas, Nevada, 2001.

AUTHOR'S PROFILE

K. Mohan
M.Tech. and having six years of teaching experience. Prasently working as a Assiatant professor in the department of Electronics and Comunnication Engineering in SEAT, Tirupati, Andhra Pradesh, India.
E-mail: kvmohank451@gmail.com

Dr. K. V. Ramanaia
Ph.D. and having 18 yars of teaching experience and published more than five international journals. He is prasently working as a Professor in the department of Electronics and communication Enginerering in Yogi vemana University, kadapa, kadapa dist, andhrapradesh, India.
E- mail: ramanaiahkota@yahoo.com

Dr. S. A. K. Jilani
Ph.D. and having a 15 yars of teaching and research experience and published more than eighteen international journals. He is prasently working as a Professor and director for research and development cell in the department of Electronics and communication Enginerering in MITS, Madanapalee, andhrapradesh,india.
e-mail: jilani_consult@yahoo.com