Automatic Modulation Classification Using Combination of Genetic Programming and KNN

Muhammad Waqar Aslam, *Student Member, IEEE*, Zhechen Zhu, *Student Member, IEEE*, and Asoke Kumar Nandi, *Fellow, IEEE*

***Abstract*—Automatic Modulation Classification (AMC) is an intermediate step between signal detection and demodulation. It is a very important process for a receiver that has no, or limited, knowledge of received signals. It is important for many areas such as spectrum management, interference identification and for various other civilian and military applications. This paper explores the use of Genetic Programming (GP) in combination with K-nearest neighbor (KNN) for AMC. KNN has been used to evaluate fitness of GP individuals during the training phase. Additionally, in the testing phase, KNN has been used for deducing the classification performance of the best individual produced by GP. Four modulation types are considered here: BPSK, QPSK, QAM16 and QAM64. Cumulants have been used as input features for GP. The classification process has been divided into two-stages for improving the classification accuracy. Simulation results demonstrate that the proposed method provides better classification performance compared to other recent methods.**

***Index Terms*—Automatic modulation classification, Genetic programming, K-nearest neighbor, Classification using genetic programming, Higher order cumulants.**

I. Introduction

AUTOMATIC modulation classification (AMC) is a very important process before demodulation of a signal. It is a rapidly evolving area with various civilian and military applications. For civilian applications it is used for spectrum management, signal confirmation and interference identification. In military applications it is used for electronic warfare, surveillance and threat analysis [1]-[2]. Minimization or replacement of hardware in communication systems has resulted in the birth of software defined radio (SDR) [3]-[7]. SDR is a flexible communication device that can handle multiple modulation schemes. With the recent development in SDR, AMC has gained more attention than ever. AMC can be used at the front end of SDR to classify the modulation type of an incoming signal. In this way a single SDR can handle multiple modulations. Cognitive radio is also an emerging technology for dynamic spectrum access [8]-[10]. A cognitive radio is a SDR that intelligently senses its environment, tracks changes, plans the most appropriate response and reacts accordingly, so it can be called as refined SDR. One idea in cognitive radio is to allow a secondary user to reuse or share a radio spectrum originally allocated to primary user if it is under utilized by primary user. But in order to avoid interference with primary user, secondary user should have accurate knowledge of the signal used by primary user. For this reason the performance of cognitive radio can be greatly enhanced by using a reliable modulation classification scheme. A complete survey of different AMC techniques used so far can be found in [11].

The design of an automatic modulation classifier involves two steps: pre-processing of the incoming signal and proper selection of classification algorithm. Regarding the second step, two kinds of classification approaches have been used. The first approach is a likelihood based (LB) approach [12]-[16] and the second is called a feature based (FB) approach [17]-[31]. The former approach (LB) treats AMC as a hypothesis testing problem and compares the likelihood function of the incoming signal with a threshold value. The fundamental assumption made in this approach is that the probability density function (pdf) of an incoming signal is already known. If the actual pdf of unknown parameters is identical to the assumed pdf, the LB method gives optimal solution in the sense that it minimizes the probability of false classification. Wong and Nandi used Maximum Likelihood (ML) classifier for classification of phase-amplitude modulated schemes [12]. They introduced the idea of estimation of SNR to propose an Estimate ML (EsML) classifier. They proposed the use of minimum distance classifier for reducing the complexity of EsML classifier. They also used blind source separation (BSS) for rectifying the carrier phase offset problem. Wei and Mendel used Maximum Likelihood (ML) approach for classification of digital amplitude phase modulations [13]. They gave an upper bound for any classifier using ideal conditions and assuming that the number of available symbols is infinity.

In second approach, known as FB approach, several features are extracted from received signal and a decision is made according to the values of those features. This method may not be optimal but it is usually very simple, easy to implement and may give optimal performance when designed properly. Xi and Wu used higher order statistics in generic framework for blind channel estimation and pattern recognition [17]. The advantage of their method was that the complete channel information was not required. Wong and Nandi used genetic algorithm (GA) and artificial neural network (ANN) [20] for

Manuscript received March 14, 2011; revised September 16, 2011 and February 17, 2012; accepted April 23, 2012. The associate editor coordinating the review of this paper and approving it for publication was D. Reynolds.

A. K. Nandi is with the Department of Electrical Engineering & Electronics, The University of Liverpool, UK. He is also with the Department of Mathematical Information Technology, University of Jyvaskyla, Jyvaskyla, Finland (e-mail: a.nandi@liverpool.ac.uk).

M. W. Aslam and Z. Zhu are with the Department of Electrical Engineering & Electronics, The University of Liverpool, UK (e-mail: {m.w.aslam, z.zhu}@liverpool.ac.uk).

Digital Object Identifier 10.1109/TWC.2012.060412.110460

AMC. The best feature subset from combined statistical and spectral feature set was chosen using GA. They proposed resilient backpropagation (RPROP) algorithm for AMC. Swami and Sadler used fourth order cumulants for classification of ASK, PSK and QAM signals [21]. Their method was robust in the presence of frequency and phase offsets. Wong and Nandi used Naïve Bayes classifier in combination with higher order statistics for AMC [22]. Their mehod was robust to the presence of carrier and phase offsets. There are a lot of other techniques used for AMC and a detailed survey is presented in [11].

AMC has been tried by machine learning methods in the past: Azzouz and Nandi [18]-[19] used ANN for the classification of both, analogue and digital modulated signals. Wong and Nandi used ANN and GA for this purpose [20].

The work presented in this study involves combination of GP and KNN for automatic classification of BPSK, QPSK, QAM16 and QAM64 signals. The reason for choosing these modulations is, that these four modulations are included in one of the IEEE standards 802.11a (http://standards.ieee.org/getieee802/download/802.11a-1999.pdf). KNN has been used for fitness evaluation of individuals during training and also for testing the best individual produced by GP. GP has not been used in the past for modulation classification although it has been used for some other classification problems [32]-[40]. In this paper a new strategy has been used involving GP and KNN. The process has been divided into two stages to improve the classification accuracy. Two trees have been created for each stage for the classification of above mentioned modulations. A single tree is first created at the first stage to classify BPSK, QPSK and QAM16/QAM64. At the second stage another tree is created to classify QAM16 and QAM64. GPLab toolbox has been used for experiments in this study (http://gplab.sourceforge.net). All the experiments are done in MATLAB environment.

The paper is organized as follows: Section II explains signal model and the features used. Section III explains genetic programming and different terms of GP. Our proposed method is presented in Section IV. Section V is about experiments and results. Section VI gives a comparison of our method with other existing methods. Conclusion is drawn in Section VII.

II. Signal Model And Feature Extraction

A. Signal Model

A general expression for the baseband waveform received can be written as

$$y(n) = Ae^{j(2\pi f_o nT+\theta_n)} \sum_{l=-\infty}^{\infty} x(l)h(nT - lT + \epsilon_T T) + g(n) \tag{1}$$

where A is the unknown amplitude factor, f_o represents carrier frequency offset, T is the symbol spacing, $h(.)$ is the residual baseband channel effects, θ_n is the phase jitter which may vary from sample to sample, $x(l)$ is the symbol sequence, ϵ_T is timing error, and $g(n)$ is additive, white, Gaussian noise (AWGN). We assume that the channel has been equalized and the residual channel effect $h(.)$ is negligible. Constellation independent algorithms are available to equalize the effects of the pulse shaping filter and the linear channel (for example, see [21]). The parameters, such as T (the symbol timing), f_o (the carrier frequency offset), ϵ_T (timing error) and θ_n (phase jitter) are assumed to be known.

B. Feature Extraction

The modulation classification scheme proposed here is based on pattern recognition approach. The features extracted from the received signal are fourth order and sixth order cumulants. Cumulants are made up of moments, so moments have been implicitly used as features. For a complex valued stationary signal the cumulants can be defined as below [23]

$$\begin{aligned}
C_{40} &= cum(y(n), y(n), y(n), y(n)) = M_{40} - 3M_{20}^2 \\
C_{41} &= cum(y(n), y(n), y(n), y^*(n)) = M_{41} - 3M_{20}M_{21} \\
C_{42} &= cum(y(n), y(n), y^*(n), y^*(n)) \\
&= M_{42} - |M_{20}|^2 - 2M_{21}^2 \\
C_{60} &= cum(y(n), y(n), y(n), y(n), y(n), y(n)) \\
&= M_{60} - 15M_{20}M_{40} + 3M_{20}^3 \\
C_{61} &= cum(y(n), y(n), y(n), y(n), y(n), y^*(n)) \\
&= M_{61} - 5M_{21}M_{40} - 10M_{20}M_{41} + 30M_{20}^2M_{21} \\
C_{62} &= cum(y(n), y(n), y(n), y(n), y^*(n), y^*(n)) \\
&= M_{62} - 6M_{20}M_{42} - 8M_{21}M_{41} - M_{22}M_{40} \\
&\quad +6M_{20}^2M_{22} + 24M_{21}^2M_{20} \\
C_{63} &= cum(y(n), y(n), y(n), y^*(n), y^*(n), y^*(n)) \\
&= M_{63} - 9M_{21}M_{42} + 12M_{21}^3 - 3M_{20}M_{43} \\
&\quad -3M_{22}M_{41} + 18M_{20}M_{21}M_{22}
\end{aligned}$$

where M_{pq} represents the moment of a signal which is defined as

$$M_{pq} = E[y(k)^{p-q}(y^*(k))^q] \tag{2}$$

III. Genetic Programming

A. Introduction to Genetic Programming

Genetic programming (GP) belongs to the class of evolutionary algorithms which attempt to emulate Darwinian model of natural evolution. It is a machine learning methodology which is used to optimize a population of individuals (computer programs) with the help of fitness values. GP finds the solution of a problem in the form of a mathematical formula. Each solution is a computer program represented in the form of tree. Each tree has terminal nodes (data nodes) and internal nodes (function nodes). Each individual is given a fitness value which quantifies its ability to solve the given problem. The fitness value is computed using a user-defined fitness function. This fitness function used depends upon the nature of the problem. The advantages GP have on other machine learning methods are listed below.

(a) No prior knowledge about the statistical distribution of data is needed.

(b) Pre-processing of data is not required and data can be used directly by GP in its original form.

(c) GP returns mathematical function as output which can be used directly in application environment.

(d) GP has the inherent capability to select useful features and ignore others.

Typically GP implementation follows the following steps:

(a) GP starts with a randomly generated population of user

TABLE I
GP PROGRAM PARAMETERS

Parameter	Standard Value
Number of Generation	100
Population Size	25
Function Pool	plus, minus, times, reciprocal, negator, abs, sqrt, sin, cos, tan, asin, acos, tanh, *mylog
Terminal Pool	Moments and Cumulants
Genetic Operator	crossover, mutation
Operator Probability	0.9, 0.1
Tree Generation	ramped half-and-half
Initial Maximum Depth	28
Selection Operator	Lexictour
Elitism	replace

*mylog is a protected $log_e(x)$ function which ignores input if it is zero

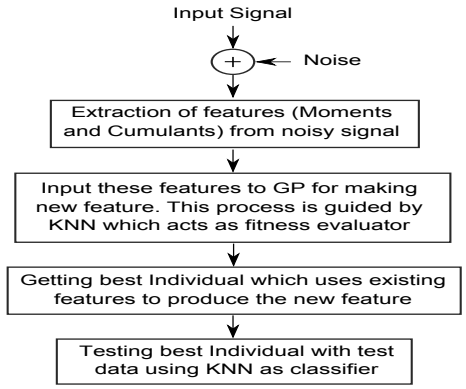

Fig. 1. System diagram explaining different stages of system.

defined size.
(b) Each individual is assigned a fitness value which represents the strength of the individual to solve the given problem.
(c) A genetic operator is applied on current generation to give birth to individuals of next generation. Genetic operators are explained in the next section.
(d) All the individuals are given fitness values and those individuals having better fitness values get transferred to the next generation.
(e) Step (c) and (d) are repeated till a desired solution is achieved. Otherwise GP is terminated after a certain number of generations set by the user.
The idea of GP can be described by the following equation.

$$g_{t+1} = g_o(f(g_t)) \tag{3}$$

where g_{t+1} is the new generation being produced, g_t is the current generation, function f chooses the fittest individuals in the current generation and the function g_o applies genetic operators on the current generation to produce the next generation. There are different ways to represent the individuals (computer programs) in GP. One of the common representation is tree representation and the same representation has been used here as well. A tree has terminal nodes, internal nodes and output node. Terminal nodes are the inputs, internal nodes represent the functions operating on inputs while the output node gives the output of the tree.

B. Some Basic terms of GP

1) Function Pool: Function pool contains all the operators that will be used by internal nodes of trees. The functions to be included in the function pool depend purely on the nature of problem. For example for logical problems, logical functions like AND, OR, XOR, etc. are used, and for non-linear problems non-linear functions are preferred. All these functions can have different number of inputs but always produce a single output. The function pool used is given in Table I.

2) Fitness Function: It is the most important parameter of GP that drives GP algorithm. It quantifies the ability of individuals to solve the problem. This is the value which is used to decide which individuals are going to be transferred to next generation. Whenever a new generation is created each individual is given a rank or fitness value calculated through fitness function. These individuals are then sorted according to fitness values. From this sorted list the individuals with better fitness values propagate to the next generation. Fitness function is a user-defined function and depends on the nature of the problem.

3) Genetic Operators: Genetic operators are the programs that operate on the current generation and produce the next generation. There are different types of genetic operators and some of them are given below.
(a) Crossover
This is the genetic operator used mostly in GP. In this genetic operator two individuals are crossed with each other. A node is randomly chosen on both trees. The sub trees from these nodes downwards are swapped with each other to make offspring.
(b) Mutation
Mutation takes only a single parent as input and returns a single child as output. It alters the parent tree in a random way to make a child tree. It randomly chooses a node on the tree and replaces the sub-tree down from that node with a randomly generated sub-tree.
(c) Reproduction
It is similar to asexual or cloning process. It just copies the individual from the current generation to the next generation without altering it.

IV. THE PROPOSED SYSTEM

A. GP for Classification

GP has been used in the past for classification purpose [33]-[40]. In [33] Espejo, Ventura and Herrera gave a survey on the application of genetic programming for classification purpose. Zhang, Jack and Nandi [34] used GP for feature generation and K-nearest neighbor (KNN) for classification purpose.We have adopted part of this approach. Zhang et al. [35] used the idea of multiple thresholds for multi-class classification. They used pre-defined threshold values for different classes. GP was used to generate outputs for different classes to fit in between those threshold values. The problem with this method was that these thresholds are problem dependent and their values vary for different problems. So the values of threshold need to be set manually. Also before using GP to optimize the outputs, one has to carry out a lot of tests to get the threshold values and the spacing between these threshold values. It is a time consuming task and still it is difficult to get optimum threshold

values. Eggermont, Eibben and Hemert [37] presented idea of weighting fitness function for data classification. The fitness function was modified in an on-line fashion giving higher weights to data which is hard for classification. Kishore et al. [38] and Muni et al. [39] presented the idea of dividing n-class classification into n 2-class classification problems to realize multi-class classification with GP. This method inherited the simplicity of 2-class problems. The resources required for this technique are n times the resources required for 2-class problem as it involves n 2-class problems. The complexity of this classifier increases with increase in number of classes making it suitable for only small number of classes.

In this paper GP has been used as a new feature generator and KNN has been used for fitness evaluation of new features created by the GP program. As the GP evolves, different combinations of input features (moments and cumulants) are tried by GP in the form of individuals. These individuals give a single output which is a new feature having combination of existing features. In the past literature, the existing features have been used for classification but a combination of those features should surely return better performance than individual features. GP automates this complex process of trying different combinations of existing features in an efficient way. Once GP returns the best individual (new feature), it is tested by KNN classifier. Any other classifier can also be used for classification at this stage but KNN has been chosen because of its simplicity. The full process is shown in Figure 1.

B. Two-Stage Genetic Programming

Swami and Sadler [21] gave a detailed analysis of AMC for different modulation schemes. They used higher order cumulants for classification of a large set of modulation schemes. They also did classification of BPSK, QPSK and QAM(>4). Their results showed that the classification of QAM16 and QAM64 was difficult compared to other modulations. Wong and Nandi [22] also used higher order cumulants for the classification of above mentioned modulations. Also they came to the conclusion that the classification of QAM(>4) is difficult as compared to the classification of other two modulation schemes. This shows that classification of BPSK, QPSK and QAM(>4) is easier as compared to the classification of QAM16 and QAM64. In order to cope with this problem we have used a two-stage genetic programming.

Initially GP was used to classify these modulations in a single stage but as the classification of BPSK and QPSK is easier compared to other two modulations, a drift was seen in GP to the classification of BPSK and QPSK and the improvement in classification of QAM(>4) was minimal. Therefore, in order to achieve a better performance for all classes, a two-stage genetic programming has been used here to counter this problem. At the first stage classification of BPSK, QPSK and QAM(>4) is performed and at the second stage GP is used again to do the classification of remaining two classes. So at the second stage GP creates a new tree for the classification of remaining classes. As this tree is independent from the first tree and it is solely devoted for the classification of QAM(>4) modulations, the accuracy would be better. The two stages are shown in Figure 2. The performance of GP for these two stages for classification of different classes is shown in Figure 3 and Figure 4.

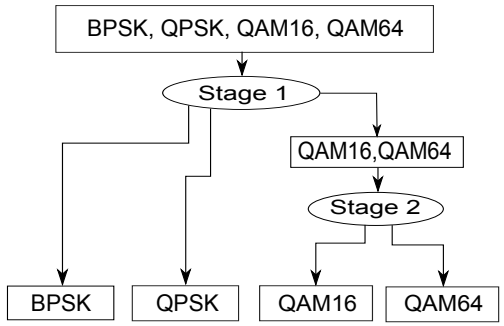

Fig. 2. Block diagram of the proposed method showing two stages.

C. Fitness Calculation Using KNN

Fitness is an important parameter of individuals in GP. It is the characteristic of an individual which tells how good an individual is in solving the given problem. Normally the output is set before the GP evolves and GP is asked to produce the required output from given training samples. The inverse of number of wrong classifications is termed as the fitness of an individual, so the higher the fitness value the better the individual is. We are also taking the inverse of the number of wrong classifications made by an individual to calculate its fitness. But instead of setting the outputs before the GP starts, KNN has been used here to evaluate the individual. Samples of each modulations are used in training the individuals. Out of these training examples (feature values) 75% are used for creating the reference samples for an individual and remaining 25% are used for evaluating the individual. Whenever an individual is created these 75% training samples are given as input to the individual. These samples make a reference space for each modulation. Ideally in this reference space, reference samples for each modulation should be apart from each other. Now the remaining 25% training samples are used as evaluation samples and are given as input to individual to test its performance. For each evaluation sample KNN is used to calculate the distance of the sample from neighboring reference samples. The formula for calculating the distance of a sample from its neighbors is given below

$$d_{er}{}^2 = (f_e - f_r).(f_e - f_r)^* \tag{4}$$

where f_e represents the sample to be evaluated, f_r is the reference sample value and * represents complex conjugate. This f_e is the feature generated by GP. After calculating the distance from neighbors, the modulation type with maximum number of neighbors is taken as modulation of this test sample. In a similar way the modulation type of these 25% evaluation samples is found. After all this testing has been done, individual is assigned a fitness value by calculating the number of wrong classification made by individual and then taking the inverse of this value. The formula for calculating the fitness function of individual is given below

$$f = \left(\sum_{i=0}^{n} (p_i e_i) \right)^{-1} \tag{5}$$

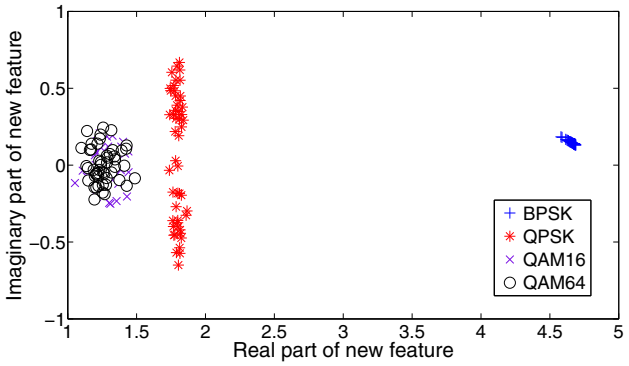

Fig. 3. Feature distribution for reference data for first stage 3-class classification. The data is for BPSK, QPSK, QAM16 and QAM64 each obtained from signals with 2048 number of samples and at 20 dB SNR.

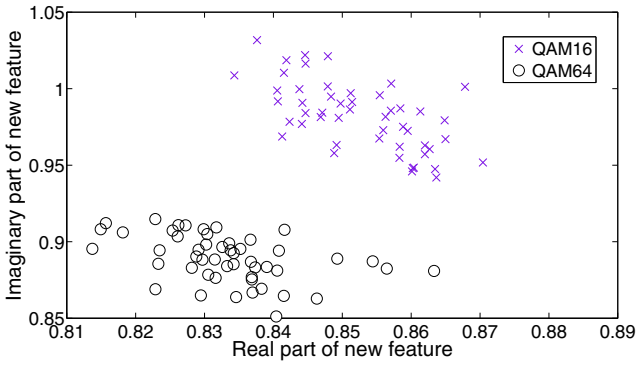

Fig. 4. Feature distribution for reference data for second stage 2-class classification. The data is for QAM16 and QAM64 each obtained from signals with 2048 number of samples and at 20 dB SNR.

where n is the number of classes to be classified, e_i is the number of classification errors made by individual for class i, p_i is the penalty term used for each class. By changing p_i one can bias the evolution of GP. The larger is the value of p_i for class i, the more biased would GP be to classify correctly class i. So this fitness value represents the inverse of errors made by an individual in classifying modulation classes. The individuals having larger fitness values which indicates less number of errors and better classification performances, will have more chances of participation in the next generation.

D. Complexity of The Proposed Classifier

Many researchers think that GP classifier will take a long time for classification as the time for evolution could be very long but the computational complexity of the final classifier is not to be confused with the training time of the classifier. Once we get the final solution from training a GP, that final solution is used for classification and the computational complexity of training a GP does not come into account while using the final solution. The final solution produced by GP has inputs as cumulants and some functions from the function pool. So the complexity of this particular solution really depends on the particular cumulants and functions used by final GP solution. We have used sixth order cumulants and the complexity of calculating these cumulants is not very high. The function pool used has been presented in Table I. Also the output produced used by this solution is tested through KNN classifier which has complexity of $O(nd)$ where n is the number of reference samples and d represents dimensions of reference data. Here we have used two dimensional data in the form of complex numbers but the function pool contains an abs function which returns the magnitude of complex number as output when the input is a complex number. If the final solution is using abs function in the last stage the final output could be a real value. In a nutshell the complexity of our final classifier is $O(nd)$ + complexity of final solution.

E. GP Program Parameters

The number of generations used in each run of GP is 100 while the number of individuals in each generation is 25. The program terminates itself if the performance reaches 100%. The genetic operator crossover has a probability of 90% and the probability of mutation is 10%. During the first stage the value of the penalty term p_i is set as 1 and QAM16 and QAM64 are treated as one class. In the second stage the value of p_i is set as 0 for the first two modulations so that they do not take part in the evolution and evolution is solely for the classification of QAM16 and QAM64. The other parameters used for GP are given in Table I.

V. Experiments and Results

A. Simple KNN Classifier

In this section only KNN has been used to test the effectiveness of KNN and input features without involving GP. This will help us to make a comparison of input features performance with KNN against the new feature's performance (generated by GP with KNN). There are seven features used here for performance evaluation of simple KNN classifier. These seven features are the cumulants defined earlier. The method is quite simple. Initially some of the feature values are used as reference data for KNN. As there are seven input features so this makes a seven dimensional space for KNN. After this some test values of these features are given to KNN and KNN calculates the distance of these test samples from K neighboring samples. The class having the maximum number of neighbors is considered as the class of test sample. The number of reference samples used for KNN are 200, each modulation type having 50 samples. The number of realizations for test are 40,000, each modulation having 10,000 realizations. The equation for calculating the distance of test sample from a reference sample is given below.

$$d^2 = \sum_{i=1}^{7} (f_i^t - f_i^r).(f_i^t - f_i^r)^* \tag{6}$$

where f_i^t is the i^{th} input feature value and f_i^r is the i^{th} reference feature value. Table II summarizes the result for simple KNN classifier.

B. GP with KNN Classifier

The parameters used for the experiments are given in detail in Table I. The number of generations used for all the

TABLE II
ACCURACY OF PERFORMANCE USING KNN AND GP-KNN WITH ONE STANDARD DEVIATION FOR DIFFERENT SNRS AND NUMBER OF SAMPLES

Method	SNRs	512	1024	2048	4096
KNN	5 dB	0.79	0.81	0.91	0.96
	10 dB	0.88	0.93	0.99	1.00
	15 dB	0.90	0.97	1.00	1.00
	20 dB	0.93	0.98	1.00	1.00
GP-KNN	5 dB	0.84±0.04	0.88±0.03	0.93±0.03	0.97±0.02
	10 dB	0.94±0.02	0.98±0.0	1.00±0.0	1.00±0.0
	15 dB	0.97±0.02	0.99±0.0	1.00±0.0	1.00±0.0
	20 dB	0.98±0.01	1.00±0.0	1.00±0.0	1.00±0.0

experiments were 100. The number of individuals in each of the experiments were 25. Total number of training experiments done is also 25. So the total number of individuals or solutions created were 625. The best tree out of these 625 trees was tested with test data and results are analyzed here. All the training experiments have been done in MATLAB/GPLab environment. Testing experiments are also carried in MATLAB/GPLab environment. The number of samples used are 512, 1024, 2048 and 4096 respectively, and the SNRs used are 5 dB, 10 dB, 15 dB and 20 dB. For each value of SNR and number of samples, 10,000 realization are produced. These 10,000 realizations are tested with the best tree and results are summarized. Table II shows the results obtained for particular combination of SNRs and number of samples. The results for simple KNN are also shown in the same Table. It is clear from these results that GP-KNN produces better results compared to simple KNN classifier.

The performance of GP for different SNRs and at 1024 number of samples is given in Table III in the form of confusion matrix. It is clear from this Table that classification of BPSK and QPSK is easier as compared to other two modulations. The classification performance for BPSK and QPSK is 100% in all the cases as shown in matrix. Figure 5 shows the performance against SNR for different values of number of samples. It is clear from the Figure that the classification performance reaches 100% at an SNR of 8 dB and 4096 number of samples. One can see from the Figure that greater the number of samples the better is the performance. In all the curves in Figure 5 there is a dip in performance at 3 dB. Table IV explains the reason behind this dip. Table IV gives the range of values of new feature created by GP at 2, 3, 4 and 5 dB for BPSK, QPSK, QAM16 and QAM64 respectively. At 2 dB the values of QAM16 and QAM64 are in between the values of BPSK and QPSK. As the SNR increases the feature values of BPSK and QPSK increase while the values of QAM16 and QAM64 decrease a little. At the 3 dB SNR, QPSK crosses both QAM16 and QAM64, and that is the reason why the performance is low at this particular SNR. As the SNR goes above 3 dB, QPSK feature value starts going above the feature values of both QAM16 and QAM64. This new feature value of QPSK continues to increase with increase in SNR. At 5 dB the feature value of QPSK is greater than the values of QAM16 and QAM64 so the performance always increases after this SNR. It is to be mentioned that these feature values are taken from the first stage where QAM16 and QAM64 are treated as one class, that is why their values are completely overlapping with each other in this Table.

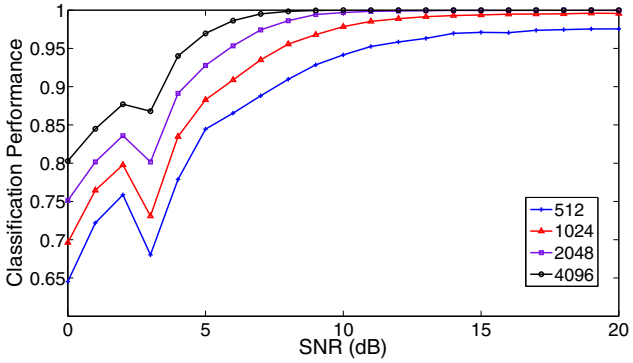

Fig. 5. Average classification accuracy for four class classification against SNR for different number of samples.

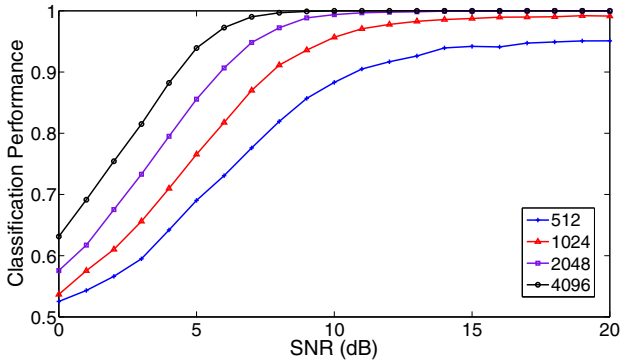

Fig. 6. Classification accuracy of QAM16 and QAM64 against SNR for different number of samples.

As our focus in this research was on the classification of QAM16 and QAM64 so the performance curves for these two modulations are presented separately as well. Figure 6 shows the performance of QAM16 and QAM64 for different SNRs. It is clear from this Figure that the performance reaches 100% at an SNR of 8 dB at 4096 number of samples. As the dip at 3 dB in Figure 5 was due to the overlap of QPSK with QAM(>4), that dip is not present in this Figure which considers only QAM16 and QAM64. Figure 7 shows the standard deviation of performance for different SNRs. It is clear from the Figure that standard deviation of performance is very low which proves the robustness of the classifier.

VI. COMPARISON WITH EXISTING METHODS

In [22] Wong, Ting and Nandi presented results for the same modulations that we have used in this paper. At an SNR of 10 dB they achieved performance of 90.2%, 94.4% and 97.9% at 512, 1024 and 2048 respectively using Naïve Bayes classifier. For the same SNR and number of samples the performance achieved through SVM was 91.2%, 94.8% and 97.9% respectively. They also used SVM and ML for classification. We have produced ML results ourselves. Figure 8 shows the comparison of our results with other methods. ML gives the upper bound performance but have

TABLE III
CONFUSION MATRIX FOR GP-TREE AT DIFFERENT SNRS AND AT 1024 NUMBER OF SAMPLES

SNR	Modulation Type	BPSK	QPSK	QAM16	QAM64
5 dB	BPSK	10000	0	0	0
	QPSK	0	10000	0	0
	QAM16	0	0	8091	3404
	QAM64	0	0	1909	6596
10 dB	BPSK	10000	0	0	0
	QPSK	0	10000	0	0
	QAM16	0	0	9557	423
	QAM64	0	0	443	9577
15 dB	BPSK	10000	0	0	0
	QPSK	0	10000	0	0
	QAM16	0	0	9870	145
	QAM64	0	0	130	9855
20 dB	BPSK	10000	0	0	0
	QPSK	0	10000	0	0
	QAM16	0	0	9924	104
	QAM64	0	0	76	9896

more computational complexity. It is clear from the Figure that our method gives better results compared to SVM and Naïve Bayes method. Up to 4 dB SNR, the performance of our method is same as the other two methods but above 4 dB, the performance of our method is better than those of the other two. In [21] Swami and Sadler presented a detailed analysis about the classification of different digital modulation schemes using cumulants. They also presented classification results for QAM16 and QAM64. They reported classification accuracy of 90% but their method had some limitations. There were two limitations: firstly the conditions were assumed to be noiseless and secondly the number of samples used was more than 10,000. Our method achieved 100% accuracy at 15 dB SNR and using 2048 number of samples. In order to compare GP-KNN results with cumulants results we implemented the method presented in [21] to classify the four modulations at a range of SNRs. The same tree structure and the same cumulants as presented in [21] have been used for results but with noisy conditions. Thresholds for classification tree were not specified in [21] but a Table was presented with theoretical values of cumulants and we have calculated thresholds from that Table. Classification accuracies were calculated at different SNRs using 1024 number of samples. These classification accuracies were averaged over 10,000 runs. We found classification accuracies (in percentage) for cumulants to be 98.2±0.22 at 20 dB (compared with 100±0 using GP-KNN), 87.1±0.45 at 15 dB (compared with 99±0) and 57.3±0.35 at 10 dB (compared with 98±0). The results clearly demonstrate the improvement in performance using GP-KNN (our research) compared to cumulants method. However, it will be possible to augment method presented in [21] by making thresholds SNR dependant and thereby to increase classification accuracies but not challenging the proposed method.

Xi and Wu [17] reported classification accuracy of 94% for an SNR of 10 dB using 2000 number of samples and the

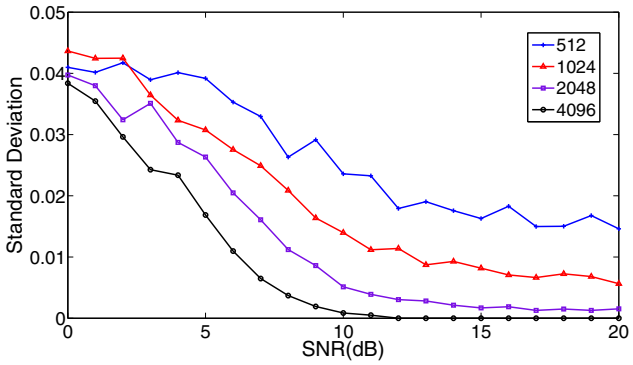

Fig. 7. Average standard deviations of classification accuracies of QAM16 and QAM64 against SNR for different number of samples.

modulations used were QAM4, QAM16 and QAM64. They used fourth order cumulants for this purpose. We achieved performance of 99.6% at an SNR of 10 dB and using 2048 number of samples. Mirarab and Sobhani [23] achieved performance of 94% at an SNR of 15 dB using 2000 number of samples. They used eighth order cumulants and a threshold dependent tree structure for this purpose. At the same SNR and at 2048 number of samples we achieved performance of 100%. Dobre, Ness and Su [24] reported classification accuracy of 70% at an SNR of 10 dB using 2000 samples. They used eighth order cyclic cumulants for classification. At 10 dB and 2048 number of samples we achieved 100% classification accuracy. Shi, Gong and Guan [25] used characteristic function method to compensate for inefficiency of cumulants in classifying higher order QAMs. They reported performance of 83% for 16QAM and 64QAM using 2000 samples at an SNR of 10 dB. They introduced timing offset in their study and it was assumed to be fixed within one frame of symbols. We have achieved 100% accuracy for the same settings but without time offset. Wu, Saquib and Yun [26] presented results for QAM4, QAM16 and QAM64 using higher order statistics. They reported classification accuracy of 94% at an SNR of 10 dB and using 2000 number of samples. We have achieved 99.6% classification accuracy at the same SNR and using 2048 number of samples for four class classification. Chaithanya and Reddy [27] reported classification accuracy of 93.5% using 1000 symbols and at 10 dB SNR for QAM16 and QAM64 using moments. For the same SNR and 1024 number of samples, we have achieved 97.8% for four class classification.

Lanjun and Canyan [28] reported classification accuracy of 78.8% using 8192 samples and at 10 dB SNR for four class classification (BPSK, QPSK, QAM16 and QAM64). They used spectral correlation function in combination with neural network for this purpose. At the same SNR and using 2048 number of samples, we have achieved 99.6% classification accuracy. In [14] F. Wang and X. Wang used Kolmogorov-Smirnov (KS) test for smaller signal samples. The method presented in [14] provides good classification accuracy using low number of samples but the improvement in performance is relevant only for high SNRs. It has been demonstrated in [14] that KS results and cumulants results are very similar

TABLE IV
RANGE OF VALUES OF NEW FEATURE AT DIFFERENT SNRS AND AT 1024 NUMBER OF SAMPLES FOR ALL MODULATIONS

SNR/Modulation	BPSK	QPSK	QAM16	QAM64
2 dB	3.2-5.5	0.6-1.3	1.2-1.5	1.3-1.5
3 dB	4.5-6.6	0.7-1.6	1.0-1.5	1.1-1.5
4 dB	5.8-7.7	1.3-1.9	0.7-1.4	0.9-1.5
5 dB	7.2-8.8	1.6-2.2	0.6-1.3	0.7-1.4

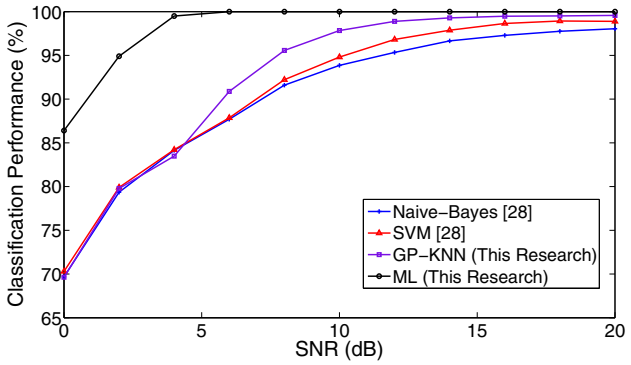

Fig. 8. Performance comparison of GP-KNN with other methods at different SNRs and using 1024 number of samples.

from 0 dB to 10 dB and the improvement is only from 10 dB to 20 dB. We have done experiments to compare our results with their results using the same three modulations and using 100 samples. We found that at lower SNR our method (GP-KNN) performs much better (e.g. 77% versus 67% at 5 dB and 82% versus 78% at 10 dB) while at higher SNR their method performs better (e.g. 86% versus 98% at 15 dB and 87% versus 100% at 20 dB). This shows that GP-KNN is quite robust for a wide range of SNRs and possesses graceful performance degradation. In [16] the complexity of KS classifier has been reduced by using off line CDF curves but the ECDF calculation and comparisons with offline CDF curves are still there. Although the complexity of GP-KNN (explained in Section IV.D) is a little higher than the KS method (after quantization) but the improvement in performance at lower SNR and robustness over a range of SNR makes it competitive.

VII. CONCLUSION

This paper presents a novel technique for automatic modulation classification of digitally modulated signals. The challenge for existing automatic modulation classification techniques is to find a feature or a combination of features good enough to distinguish between different modulations. This paper uses GP and KNN to find new features from existing features during the training phase. GP has the inherent capability to select useful features and ignore others. The modulations used are BPSK, QPSK, QAM16 and QAM64. The process has been divided into two stages and GP has been used to produce features for both stages. The two-stage classification helps in classification of QAM16 and QAM64. The results show that GP with KNN is highly successful in finding a new feature that can distinguish between these digitally modulated signals. Our results are compared with those from many of the existing techniques and it is found that the proposed technique gives better results compared to majority of these techniques and it appears to be more robust over a range of SNR.

ACKNOWLEDGMENT

Muhammad Waqar Aslam would like to acknowledge the financial support of the University of Azad Jammu and Kashmir, Pakistan. Zhechen Zhu would like to thank the University of Liverpool and the University of Liverpool Graduate Association (Hong Kong) for their financial support.

REFERENCES

[1] E. E. Azzouz and A. K. Nandi, *Automatic Modulation Recognition of Communication Signals*. Kluwer, 1996.
[2] A. K. Nandi and E. E. Azzouz, "Algorithms for automatic modulation recognition of communication signals," *IEEE Trans. Commun.*, vol. 46, pp. 431–436, 1998.
[3] N. Alyaoui, H. Ben Hnia, A. Kachouri, and M. Samet, "The modulation recognition approaches for software radio," in *Proc. 2008 International Conference on Signals, Circuits and Systems*, pp. 1-5.
[4] F. K. Jondral, "Software-defined radio-basic and evolution to cognitive radio," *EURASIP J. Wireless Commun. and Networking*, vol. 3, 2005.
[5] M. Islam, M. A. Hannan, S. A. Samad, and A. Hussain, "Modulation technique for software defined radio application," in *Proc. 2009 WSEAS International Conference on Circuits, Systems, Signal and Telecommunications*, pp. 179–182.
[6] K. E. Nolan, L. Doyle, P. Mackenzie, and D. O'Mahony, "Modulation scheme classification for 4G software radio wireless networks," in *Proc. 2002 IASTED International Conference on Signal Processing, Pattern Recognition, and Applications*, pp. 25–31.
[7] K. E. Nolan, L. Doyle, and D. O'Mahony, "Signal space based adaptive modulation for software radio," in *Proc. 2002 IEEE WCNC*, vol. 1, pp. 510–515.
[8] B. Ramkumar, T. Bose, and M. S. Radenkovic, "Combined blind equalization and automatic modulation classification for cognitive radios," in *Proc. 2009 IEEE Digital Signal Processing Workshop and IEEE Signal Processing Education Workshop*, pp. 172–177.
[9] B. Ramkumar, "Automatic modulation classification for cognitive radios using cyclic feature detection," *IEEE Circuits and Systems Mag.*, vol. 9, pp. 27–45, 2009.
[10] W. Hsiao-Chun, W. Yiyan, J. C. Principe, and W. Xianbin, "Robust switching blind equalizer for wireless cognitive receivers," *IEEE Trans. Wireless Commun.*, vol. 7, pp. 1461–1465, 2008.
[11] O. A. Dobre, A. Abdi, Y. Bar-Ness, and W. Su, "A survey of automatic modulation classification techniques: classical approaches and new trends," *IET Commun.*, vol. 1, pp. 137–156, 2007.
[12] M. L. D. Wong and A. K. Nandi, "Semi-blind algorithms for automatic classification of digital modulation types," *Digital Signal Process.*, vol. 18, pp. 209–227, 2008.
[13] W. Wei and J. M. Mendel, "Maximum-likelihood classification for digital amplitude-phase modulations," *IEEE Trans. Commun.*, vol. 48, pp. 189–193, 2000.
[14] F. Wang and X. Wang, "Fast and robust modulation classification via Kolmogorov-Smirnov test," *IEEE Trans. Commun.*, vol. 58, pp. 2324–2332, 2010.
[15] P. Urriza, E. Rebeiz, P. Pawelczak, and D. Cabric, "Computationally efficient modulation level classification based on probability distribution distance functions," *IEEE Commun. Lett.*, vol. 15, pp. 476–478, 2011.
[16] F. Wang, R. Xu, and Z. Zhong, "Low complexity Kolmogorov-Smirnov modulation classification," in *Proc. 2011 IEEE Wireless Comm. and Netw. Conf.*, pp. 1607–1611.
[17] S. Xi and H. C. Wu, "Robust automatic modulation classification using cumulant features in the presence of fading channels," in *Proc. 2006 IEEE Wireless Comm. and Networking Conf.*, vol. 4, pp. 2094–2099.
[18] E. E. Azzouz and A. K. Nandi, "Automatic identification of digital modulation types," *Signal Process.*, vol. 47, pp. 55–69, 1995.
[19] A. K. Nandi and E. E. Azzouz, "Modulation recognition using artificial neural networks," *Signal Process.*, vol. 56, pp. 165–175, 1997.

[20] M. L. D. Wong and A. K. Nandi, "Automatic digital modulation recognition using artificial neural network and genetic algorithm," *Signal Process.*, vol. 84, pp. 351–365, 2004.
[21] A. Swami and B. M. Sadler, "Hierarchical digital modulation classification using cumulants," *IEEE Trans. Commun.*, vol. 48, pp. 416–429, 2000.
[22] M. L. D. Wong, S. K. Ting, and A. K. Nandi, "Naïve Bayes classification of adaptive broadband wireless modulation types with higher order cumulants," in *Proc. 2008 International Conference on Signal Processing and Communication Systems*, pp. 1–5.
[23] M. R. Mirarab and M. A. Sobhani, "Robust modulation classification for PSK/QAM/ASK using higher order cumulants," in *Proc. 2007 International Conference on Information, Communications and Signal Processing*, pp. 1–4.
[24] O. A. Dobre, Y. Bar-Ness, and W. Su, "Higher-order cyclic cumulants for high order modulation classification," in *Proc. 2003 IEEE MILCOM*, pp. 112-117.
[25] Q. Shi, Y. Gong, and Y. L. Guan, "Modulation classification for asynchronous high-order QAM signals," *Wireless Communications and Mobile Computing*, 2011, pp. 1415–1422.
[26] H. Wu, M. Saquib, and Z. Yun, "Novel automatic modulation classification using cumulant features for communications via multipath channels," *IEEE Trans. Wireless Commun.*, vol. 7, pp. 3098–3105, 2008.
[27] V. Chaithanya and V. U. Reddy, "Blind modulation classification in the presence of carrier frequency offset," in *Proc. 2010 International Conference on Signal Processing and Communications*, pp. 1–5.
[28] Q. Lanjun and Z. Canyan, "Modulation classification based on cyclic spectral features and neural network," in *Proc. 2010 International Congress on Image and Signal Processing*, vol. 8, pp. 3601–3605.
[29] M. W. Aslam, Z. Zhu, and A. K. Nandi, "Automatic digital modulation classification using genetic programming with K-nearest neighbour," in *Proc. 2010 Military Communications Conference*, pp. 512–517.
[30] Z. Zhu, M. W. Aslam, and A. K. Nandi, "Augmented genetic programming for automatic digital modulation classification," in *Proc. 2010 IEEE International Workshop on Machine Learning for Signal Processing*, pp. 391–396.
[31] W. C. Headley and C. R. C. M. da Silva, "Asynchronous classification of digital amplitude-phase modulated signals in flat-fading channels," *IEEE Trans. Commun.*, vol. 59, pp. 7–12, 2011.
[32] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.
[33] P. G. Espejo, S. Ventura, and F. Herrera, "A survey on the application of genetic programming to classification," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 40, pp. 121–144, 2010.
[34] L. Zhang, L. B. Jack, and A. K. Nandi, "Extending genetic programming for multi-class classification by combining k-nearest neighbor," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 349–352.
[35] M. Zhang, V. B. Ciesielski, and P. Andreae, "A domain independent window approach to multiclass object detection using genetic programming," *EURASIP J. Applied Signal Process.*, vol. 8, pp. 841–859, 2003.
[36] L. Zhang and A. K. Nandi, "Fault classification using genetic programming," *Mechanical Systems and Signal Process.*, vol. 21, pp. 1273–1284, 2007.
[37] J. Eggermont, A. E. Eiben, and J. I. van Hemert, "A comparison of genetic programming variants for data classification," in *Proc. 1999 Symposium on Intelligent Data Analysis*.
[38] J. K. Kishore, L. M. Patnaik, V. Mani, and V. K. Agrawal, "Application of genetic programming for multicategory pattern classification," *IEEE Trans. Evolutionary Computation*, vol. 4, pp. 242–258, 2000.
[39] D. P. Muni, N. R. Pal, and J. Das, "A novel approach to design classifiers using genetic programming," *IEEE Trans. Evolutionary Computation*, vol. 8, pp. 183–196, 2004.
[40] H. Guo, L. B. Jack, and A. K. Nandi, "Feature generation using genetic programming with application to fault classification," *IEEE Trans. Systems, Man, and Cybernetics, Part B*, vol. 35, pp. 89–99, 2005.

Muhammad Waqar Aslam (S'12) received the B.Sc. Degree in electronics and telecommunication engineering from the University of Engineering and Technology Lahore, Pakistan, in 2007. He is currently working towards his Ph.D degree in electrical engineering at the University of Liverpool, Liverpool, UK. He worked as a Research Assistant at the University of Azad Jammu and Kashmir, Pakistan, until September 2008. His research interests include machine learning, pattern recognition, wireless communication and signal processing.

Zhechen Zhu (S'12) received his B.Eng. degree in the Department of Electrical Engineering and Electronics from the University of Liverpool, Liverpool, UK. in 2010. He attended Xi'an Jiaotong Liverpool University for the first two years of his integrated B.Eng degree before joining University of Liverpool. Now he is pursuing his Ph.D. degree at the University of Liverpool. His research interests include high order statistics, machine learning, statistical signal processing and their application in signal estimation and classification.

Professor Asoke K. Nandi received the degree of Ph.D. in High Energy Physics from the University of Cambridge (Trinity College), Cambridge, UK, in 1979. He held several research positions in Rutherford Appleton Laboratory (UK), European Organisation for Nuclear Research (Switzerland), Department of Physics, Queen Mary College (London, UK) and Department of Nuclear Physics (Oxford, UK). In 1987, he joined the Imperial College, London, UK, as the Solartron Lecturer in the Signal Processing Section of the Electrical Engineering Department. In 1991, he jointed the Signal Processing Division of the Electronic and Electrical Engineering Department in the University of Strathclyde, Glasgow, UK, as a Senior Lecturer; subsequently, he was appointed a Reader in 1995 and a Professor in 1998. In March 1999 he moved to the University of Liverpool, Liverpool, UK, to take up his appointment to the David Jardine Chair of Signal Processing in the Department of Electrical Engineering and Electronics. Professor Nandi is a Finland Distinguished Professor.

In 1983 he was a member of the UA1 team at CERN that discovered the three fundamental particles known as W^+, W^- and Z^0, providing the evidence for the unification of the electromagnetic and weak forces, which was recognized by the Nobel Committee for Physics in 1984. Currently, he is the Head of the Signal Processing and Communications Research Group with interests in the areas of signal processing, machine learning, and communications research. With his group he has been carrying out research in developments and applications of machine learning, biomedical signal processing, bioinformatics, machine condition monitoring, communications signal processing, and blind source separation. He has authored or co-authored over 400 technical publications; these include two books, entitled *Automatic Modulation Recognition of Communications Signals* (Boston, MA: Kluwer Academic, 1996) and *Blind Estimation Using Higher-Order Statistics* (Boston, MA: Kluwer Academic, 1999), and over 180 journal papers. The h-index of his publications, according to the Web of Science, is 38.

Professor Nandi was awarded the Mounbatten Premium, Division Award of the Electronics and Communications Division, of the Institution of Electrical Engineers of the U.K. in 1998 and the Water Arbitration Prize of the Institution of Mechanical Engineers of the U.K. in 1999. He is a Fellow of the Institute of Electrical and Electronics Engineers (USA), the Cambridge Philosophical Society, the Institution of Engineering and Technology, the Institute of Mathematics and its applications, the Institute of Physics, the Royal Society for Arts, the Institution of Mechanical Engineers, and the British Computer Society. In 2010 he received Glory of Bengal Award for his outstanding achievement in scientific research.