Received October 24, 2019, accepted December 15, 2019, date of publication December 30, 2019, date of current version January 6, 2020.

Digital Object Identifier 10.1109/ACCESS.2019.2962222

Performance Evaluation of Machine Learning in Wireless Connected Robotics Swarms

QIAO TIAN[1], **HAOJUN ZHAO**[2], **(Student Member, IEEE), YUN LIN**[2], **(Member, IEEE), AND FENGJUN XIAO**[3]

[1]College of Computer Science and Technology, Harbin Engineering University, Harbin 150001, China
[2]College of Information and Communication, Harbin Engineering University, Harbin 150001, China
[3]College of Public Administration, Beihang University, Beijing 100191, China

Corresponding author: Yun Lin (linyun@hrbeu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61771154, and in part by the Fundamental Research Funds for the Central Universities under Grant HEUCFG201830.

ABSTRACT The transmission and negotiation of the robot swarms and the technical support provided by the communication technology are inseparable, and modulation recognition plays an important role in this transmission. Due to the diversity of current classifiers, choosing the correct classifier to improve the classification effect has become a key issue. To this end, this paper explores the performance of different classifiers in modulation recognition, selects six classifier families, and compares a total of 77 different single classifiers on the modulation dataset, which is implemented on the following three platforms: Weka, Python and MATLAB. The results show that the strong classifiers formed by the combination of weak classifiers is very effective, and Boosting, Bagging, and Random Forest are the three best classifier families. In addition, it was found that as the signal-to-noise ratio (SNR) increases, the overall performance of the classifier families gradually improves, but the ranking of the families performance remains consistent.

INDEX TERMS Modulation, machine learning, information entropy, performance evaluation.

I. INTRODUCTION

With the continuous improvement of communication and computer performance, intelligent robotic swarms are now one of the most popular research areas in artificial intelligence. The robot group originates from biologically-inspired robotics swarms, and the relevant research has mainly examined how to make a number of autonomous robots with relatively simple structures and functions engage in a certain kind of collective behavior to accomplish complex tasks via interaction, coordination, and control. Obviously, to achieve coordination between groups, the effectiveness of communication and negotiation of robotic swarms, and the effective sharing of information between groups must be ensured. Therefore, communication and coordination between robotic swarms are vital, and effective communication can greatly improve the performance of the system. Modulation recognition is an indispensable part of communication, and modulation recognition technology is one of the important means in electronic reconnaissance; it plays a significant role in the research of modern information technology, such as communication reconnaissance, anti-reconnaissance, and communication confrontation. Modulation recognition technology is an important technology in software radios, which are widely used in military, commercial, and civil applications, and has been the focus of research in recent years [1]. Lin et al. [2] completed individual identification of radio equipment based on machine learning and dimensionality reduction. Liu et al. [3] classified modulated signals based on ensemble learning. Tu et al. [4] used semi-supervised learning with generative adversarial networks to perform signal classification. Similarly, Shi et al. [5] used dynamic threshold settings to effectively identify RF signals. Zhang [6] completed D2D communication based on RF fingerprint authentication. Wang [7] used the factual complexity method to perform feature extraction and classify communication signals. Zhang et al. [8] investigated the capability of some evolutionary algorithms to achieve optimal solutions at an affordable complexity, and also globally optimized the power allocation between channel estimation and data transmission of user devices and their host remote radio units, greatly improving the forward capacity [9]. Wen et al. [10] used the block orthogonal matching pursuit (BOMP) algorithm to recover block-sparse signals from measurements.

The associate editor coordinating the review of this manuscript and approving it for publication was Jiankang Zhang.

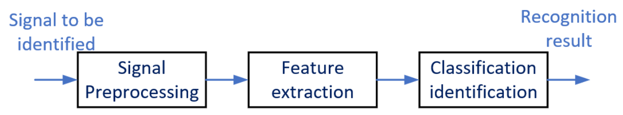


FIGURE 1. The recognition process of a statistical-patterns based recognition method.

More details about modulation recognition were presented in other previous works [11-15].

Currently, the most common methods used in modulation recognition technology are the statistical patterns-based recognition method and the decision theory-based recognition method. In the decision theory-based method, the test statistics of the signal (generally the likelihood ratio or average likelihood ratio) are obtained via theoretical analysis and calculation, and are then compared with the appropriate threshold to output the modulation signal types. Under the Bayesian minimum mispredence cost criterion, the recognition method based on decision theory can achieve the best recognition. The statistical patterns-based recognition method extracts the feature parameters of the received signal, and then classifies the feature parameters according to a certain rule, thereby realizing the recognition of different modulated signals. Considering that the decision theory-based recognition method is computationally expensive, difficult to process in real time, and sensitive to noise, among other shortcomings, a statistical patterns-based recognition method is used in the present study. The recognition process is illustrated in Figure 1.

According to Figure 1, the statistical patterns-based recognition method is mainly divided into two parts: feature extraction and pattern recognition [16]. Generally, signal preprocessing is needed before feature extraction. Common preprocessing methods include signal down-conversion, noise reduction, and filtering. Feature extraction refers to the finding of features that are easy to classify by processing the signal. There are many kinds of these features, such as transient features, constellation features, entropy features, and power spectrum features. Compared with other features, the entropy feature has the advantages of small calculation, a simple principle, and good anti-noise performance. After extracting the entropy feature of the signal, the classifier can be trained based on the extracted entropy feature dataset, and finally used to identify the modulation signal [17].

There are many types of classifiers that originate from different fields of computer science and mathematics. Some classifiers, such as linear discriminant analysis (DA) classifiers, are derived from the field of statistics; some, such as artificial neural network classifiers, are derived from artificial intelligence fields; others, such as proximity algorithm classifiers, are derived from data mining fields. Most researchers are only familiar with a certain class or a limited number of classifiers. It is customary to solve some new classification problems with several well-known classifiers, but this does not guarantee that the classifiers known to researchers can achieve the best classification results on a particular dataset, so it is of great significance for the performance study of all common classifiers.

It is evident that the development and prosperity of robotic swarms cannot be separated from the support and assistance of communication. Therefore, to efficiently classify and recognize the received signals, modulation technology provides technical support for a good communication environment, and effectively promotes the in-depth development and progress of robotic swarms. This paper focuses on the classifier families to explore the best individual and family classifiers under the modulation dataset. Specifically, this study aims to identify digital modulation signals, including 9ASK, 4ASK, 2FSK, 4FSK, 8FSK, BPSK, QPSK, 16QAM, and 32QAM. Eight entropy features of the signal are extracted as the classification basis, and the dataset of the combined entropy characteristics under mixed signal-to-noise ratios (SNRs) are constructed. Based on this dataset, the performance of the classifiers from different families is evaluated, and the classifier that optimizes the classification of the modulated signals is finally selected. The remainder of this paper is organized as follows. Section II studies the method of extracting modulation features of communication signals based on information entropy features, introduces different entropy feature extraction algorithms, and constructs a dataset of combined entropy features under SNRs of -10 dB to 2 dB. Section III introduces the classifiers of six families and prepares for the performance evaluation of subsequent classifiers. Section IV combines the dataset of entropy features based on the mixed SNR constructed in Section II, and compares the recognition effects of six classifier families for modulated signals under different SNRs. Section V further evaluates the performance in the Section VI concludes the paper.

II. ENTROPY FEATURE EXTRACTION METHOD

With the rapid development of information theory, it is possible to be used for the extraction of the features of digital communication signals. Entropy is a characteristic index used to measure the uncertainty of the signal distribution state and the complexity of the signal. Therefore, the information contained in the signal can be quantitatively described. This also provides a theoretical basis for the quantitative description of the characteristics of the signal via entropy analysis.

In this study, the power spectrum Shannon entropy, power spectrum exponential entropy, singular spectrum Shannon entropy, singular spectrum exponential entropy, wavelet energy spectrum entropy, and approximate entropy are used as the extraction characteristics of the modulated signal. A combined entropy feature dataset is constructed at -10 dB, -5 dB, 0 dB, and 2 dB, and is used as the basis for classifier simulation. Additionally, 800 samples are used as the training set, and 200 samples are used as the test set for each SNR.

A. POWER SPECTRUM ENTROPY

The power spectrum entropy is based on the frequency domain transform of the discrete signal, reflecting the distribution of the spectrum energy of the modulated signal. The larger the power spectrum entropy, the more dispersed the spectrum energy. The principle of power spectrum entropy is as follows: Let the signal sequence be: $X = \{x(1), x(2), \cdots x(n)\}$, First the fast Fourier transformation (FFT) is performed:

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j\frac{2\pi}{N}k}. \tag{1}$$

After the FFT, calculate the energy at each point in the spectrum as

$$E_i = \frac{1}{N}\sum_{k=0}^{N-1} |X(k)|^2. \tag{2}$$

Then the power spectrum of the intercepted signal is therefore

$$E = \sum_{i=1}^{N} E_i, \tag{3}$$

and the probability of each point on the power spectrum is:

$$p_i = \frac{E_i}{E}. \tag{4}$$

Combining the probability calculated by (4) with Shannon formula and exponential formula, the signal *Power Spectrum Shannon Entropy* and *Power Spectrum Exponential Entropy* of the signal can be obtained.

B. SINGULAR SPECTRUM ENTROPY

If the signal sequence is: $X = \{x(1), x(2), \cdots x(n)\}$, the delay embedding technique is first used to analyze the intercepted modulated signal sequence with a window of length M. Let the window's delay parameter be 1; the window of length M can divide the signal sequence into $(N - M)$ segment data to form matrix $\mathbf{A}$ with a dimension of $(N - M, M)$.

Singular value decomposition is performed on $\mathbf{A}$:

$$\mathbf{A} = \mathbf{UBV}^T, \tag{5}$$

where $\mathbf{B} = diag\{\lambda_1, \lambda_2, \ldots, \lambda_M\}, \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_M \geq 0$ is the singular value of $\mathbf{A}$.

Then the probability that each singular value occupies the entire singular spectrum is:

$$p_i = \frac{\lambda_i}{\sum_{i=1}^{m} \lambda_i}. \tag{6}$$

Combining the p_i obtained by (6) with Shannon formula and exponential formula, the signal *Singular Spectrum Shannon Entropy* and *Singular Spectrum Exponential Entropy* can be obtained.

C. WAVELET ENERGY SPECTRUM ENTROPY

Wavelet transform can be used to analyze non-stationary signals. Wavelet transform can obtain the basis function by shifting and scaling the mother wavelet and scale function. The wavelet basis function has both high-frequency and low-frequency components, and can simultaneously locate wavelets in the time domain and the frequency domain. Compared to Fourier transform, wavelet transform can more finely describe the local subtle features of the signal.

For the signal to be analyzed, its continuous wavelet transform is expressed as

$$Wf(a, b) = \langle f, \psi_{a,b}\rangle = \frac{1}{\sqrt{|a|}}\int_{\infty}^{\infty} f(t)\psi^*\left(\frac{t-b}{a}\right)dt. \tag{7}$$

The corresponding wavelet inverse transform is

$$f(t) = \frac{1}{C_\psi}\int_0^{\infty}\int_{\infty}^{\infty}\frac{1}{a^2}Wf(a, b)\psi\left(\frac{t-b}{a}\right)dadb. \tag{8}$$

The wavelet transform scale is selected as j, and Fourier transform is performed on the wavelet signal to obtain

$$X(k) = \sum_{n=1}^{N} d_i(n)W_N^{kn}, \tag{9}$$

where $W_N^{kn} = \exp\left(-j\frac{2\pi}{N}kn\right)$.

The power spectrum of the wavelet signal can be obtained by calculating the wavelet signal of each layer:

$$S(k) = \frac{1}{N}|X(k)|^2, k = 1, 2, \cdots, j+1. \tag{10}$$

After normalizing the power spectrum,

$$p_k = \frac{S(k)}{\sum_{i=1}^{N} S(i)}. \tag{11}$$

Therefore, the signal wavelet energy entropy can be obtained as:

$$H_{sw} = H(p_1, p_2, \ldots, p_N) = -\sum_{i=1}^{N} p_i \log_2 p_i. \tag{12}$$

D. APPROXIMATE ENTROPY

Approximate entropy describes the variation of the similarity pattern of a time series in its reconstruction dimension, and measures the nonlinear complexity of the time series. For a given point time series, the approximate entropy calculation steps are as follows. The sequence $u(i)$ is composed of m dimensional vectors $X(i)$ in order, where $(i = 1, 2, \ldots, N - m + 1)$:

$$X(i) = [u(i), \ldots, u(i+m-1)]. \tag{13}$$

Calculate the distance between the vector $X - i$ and the rest of the vector $X - j$ for each i value, where $(j = 1, 2, \ldots, N - m + 1)$:

$$D_{ij} = \max_{k\in[0,m-1]} |u(i+k) - u(j+k)|. \tag{14}$$

Given a threshold $r(r > 0)$, the ratio of the number of $d_{ij} < r$ to the total number for each i-value is recorded as $C_i^m(r)$, i.e.:

$$C_i^m(r) = num\left\{d_{ij} < r\right\}/(N - m + 1). \tag{15}$$

Roughly speaking, $C_i^m(r)$ reflects the probability that the m-dimensional modes in the sequence approximate each other in the sense of similar tolerance r. First take $C_i^m(r)$ as the logarithm, and then find the average of all i, denoted $\phi^m(r)$, namely:

$$\phi^m(r) = (N - m + 1)^{-1} \sum_{i=1}^{N-m+1} \ln C_i^m(r), \tag{16}$$

add 1 to m and repeat the previous steps to get $\phi^m(r)$:

$$ApEn(m, r, N) = \phi^m(r) - \phi^{m+1}(r). \tag{17}$$

In the above steps, m is a pre-selected mode dimension, usually taken as 2, and r is a pre-selected similar tolerance. According to experience, r is $0.1SD(u) - 0.25\ SD(u)$, SD represents the standard deviation:

$$SD = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}, \tag{18}$$

where μ is the average value.

III. DETAILS OF CLASSIFIERS

In this section, six representative classifier families are extracted, and 77 individual classifiers are compared. The classifier families all have good classification performance. The individual classifiers will end with -w, -p, and -m, indicating that they are respectively implemented on Weka, Python, or MATLAB platforms. The details of all the individual classifiers are explained in Appendix A.

A. DA

The basic idea of DA is to summarize the regularity of the classification of objective objects based on the data information of several samples of each category that has been mastered, to establish a discriminant function, and then to discriminate the category of the new sample function according to the discriminant function of the summary.

DA methods are mainly divided into three categories, namely distance discrimination, Fisher discrimination, and Bayes discrimination. The distance discrimination method is the simplest and most intuitive. It is applicable to the discrimination of continuous random variables, and has no limit on the probability distribution of variables. Fisher discriminant analysis (FDA) is a newly-developed dimensionality reduction technology. It is based on the criteria that the variance within the class is as small as possible, and the variance between classes is as large as possible, to determine the function. The basic idea of the method is projection, i.e., the combination of the independent variables of the original space is first projected into the space with lower dimensions, and then classified. In 2015, Mahmoudi et al. [18] proposed an improved Fisher discriminant function, Modified FDA, which makes the traditional function more sensitive to important instances, thus maximizing profits from fraudulent/legal classifiers; the results confirmed that Modified FDA can bring more profits. In 2016, Wu et al. [19] used the deep linear DA of the Fisher network to solve the problem of personnel classification and recognition. Bayes discriminant analysis (BDA) is a multivariate statistical analysis method based on Bayesian criteria for DA based on the cost decision with minimum risk or the maximum likelihood ratio. The BDA method is applied in many fields, including in the prediction of minimum risk [20], the prediction of coal and gas explosions [21], and the estimation of the default probability of loan applicants [22]. Zhang et al. [23] proposed an improved BDA method in which the discriminant function allows each class to have multiple Bayesian decision boundaries, and each Bayesian decision boundary is in its own subspace. The specific categories of all classifier families are detailed in Appendix A.

B. SUPPORT VECTOR MACHINE (SVM)

The SVM is one of the most powerful methods in machine learning algorithms. It can find a balance between model complexity and classification ability given limited sample information. Compared to other machine learning methods, the SVM has many advantages in that it can overcome the effects of noise and work without any prior knowledge. The SVM is a non-probabilistic binary linear classifier that predicts an input to one of two classes for each given input. It optimizes the linear analysis and classification of hyperplane formation techniques.

In recent years, SVM has primarily been used in the fields of text detection, medical treatment, human body recognition, and vehicle transportation. Omara et al. [24] used pairwise kernels combined with linear kernels to identify the pairwise SVM for the effective recognition of human ear images. The experimental results demonstrated that the recognition rate of the pairwise SVM was 98.3%, which is better than that of the traditional SVM. Elleuch et al. [25] designed an SVM-based deep learning model (DSVM) for handwriting recognition systems. By using dropout technology, it is possible to select key data points while efficiently identifying objects and avoiding overfitting classification. Bron et al. [26] improved the features according to the weight vector of the SVM, and then classified dementia. The highlight of this paper was the application of the weight vector method to the SVM for feature selection, and the use of SVM for target classification. The SVM parameters in the optimization decision stage further improved the recognition performance.

C. NEAREST NEIGHBOR (NN)

The NN algorithm is mainly used for classification and regression in machine learning. To determine the category of an unknown sample, all training samples are used as representative points, the distances between the unknown sample and all training sample points are calculated, and the NN is used. The category is the sole basis for determining the unknown sample category. Because the NN algorithm is particularly sensitive to noise data, the K-nearest neighbor algorithm (KNN) is introduced. The main concept of the KNN is that when the data and tags in the training set are known, the test data are input, the characteristics of the test data are compared with the features corresponding to the training set, and the most similar K in the training set is found.

Many scholars are committed to improving the research on KNN algorithms. The improvement methods are roughly divided into the following types. One type is the KNN algorithm for improving distance. The method of cosine KNN combined with entropy proposed by Wang et al. [27] in 2018 is an example of this type of improvement. The second type of improvement method is based on the feature-weighted KNN algorithm, as the original feature weighting algorithm was not sufficiently accurate [28]. In 2018, Huang et al. [29] improved the problem by using the DCT-KNN weighting algorithm; this third method is based on the central KNN. The improved algorithm was proposed by Wang et al. [30] in 2017. This method reduces the computational complexity of the original nearest feature line method, and improves the classification performance under the premise of ensuring the accuracy of the center vector. It is an improved algorithm based on sparse representation. In 2008, Wang [31] proposed a sparse representation classification method. The basic idea of sparse representation classification is to represent a given test sample as a sparse linear combination of training samples.

D. RANDOM FOREST (RF)

In machine learning, an RF is a classifier that contains multiple decision trees, the output of which is determined by the mode of the category of the individual tree output. In 2001, Breiman combined the Bagging integrated learning theory [32] with the random subspace method [33] to propose an RF algorithm. RF is an integrated learning model based on a decision tree. It contains multiple decision trees trained by Bagging integrated learning technology. When inputting samples to be classified, the final classification result is output by a single decision tree and vote to decide. The RF solves the performance problem of the bottleneck of the decision tree, has good tolerance to noise and outliers, and has good scalability and parallelism for high-dimensional data classification problems. The RF is a discriminant and non-linear model. It supports classification problems, regression problems, and multi-classification problems, which have good interpretability.

Scholars have also conducted much research on RF. Based on this research, rotation forests introduce the feature transformation of principal component analysis (PCA) [34], and stratified RF [35] uses the weight of Fisher discriminant projection. The feature is divided into two parts, namely strong information features and weak information features. Subspace selection RFs [36] apply a statistical criterion to divide the feature into three parts. The p-value is used to measure the importance of the feature, and the feature is divided into either information features or non-information features. PCA and stratified sampling-based RFs [37] are proposed as methods for classifying features into information and non-information features based on the results of PCA output. In addition, Wang et al. [38, 39] proposed a probabilistically optimized RF, called Bernoulli RF, by using two Bernoulli distributions to control the selection of segmentation features and segmentation values.

E. BOOSTING

The Boosting algorithm is an important integrated learning method. It becomes a strong learner with high accuracy by combining weak classifiers. The core of the algorithm is to use the weighted sample training classifier. In the process of each classifier training, if the samples are correctly classified, the weight of the samples can be reduced when constructing the next basic classifier training set. If the sample is incorrectly classified, its weight is increased in the next basic classifier training set.

In 1990, Schapire first constructed a polynomial-level algorithm, the original Boosting algorithm. This algorithm can transform weak classification rules into strong classification rules. In 1995, Freund and Schapire [40] proposed the AdaBoost (adaptive boosting) algorithm, which is as efficient as the original Boosting algorithm, but does not require any prior knowledge about the performance of the weak learner. It can be applied to practical problems very easily. In 2009, Pavan et al. [41] proposed a Boosting framework for semi-supervised learning called SemiBoost. The algorithm contains the underlying supervised algorithm and uses unlabeled data to improve its performance. In 2012, Shen et al. [42] designed a novel Boosting algorithm that utilizes the available Universum data, hence the name UBoost. In 2013, Chi et al. [43] improved AdaBoost and proposed a Boosting algorithm (CRBoosting) based on cooperative representation (CR).

F. BAGGING

The Bagging algorithm is directly based on the self-sampling method, i.e., sampling in the training set of capacity m using the method of put-back sampling to form a sampling set containing m samples. In the Bagging algorithm, each base classifier uses such a sample set formed by put-back sampling to train; this forms a plurality of base classifiers, and these base classifiers are combined to form a Bagging algorithm.

Stability is the key factor in Bagging for the improvement of the accuracy of prediction; Bagging can improve the accuracy of prediction for uncertain learning algorithms, but not for stable learning algorithms, and can sometimes even reduce the accuracy of prediction. In 2003, to improve Bagging performance, Bryll et al. [44] proposed Attribute Bagging (AB), a technique that can be used to improve the accuracy and stability of classifier sets caused by random subset features. In 2008, Cai et al. [45] proposed a weighted subspace method. By observing the diversity of classification boundaries in feature subspaces, the authors studied how to use different classification capabilities to improve bagging performance in classifier space. In 2018, Qian et al. [46] proposed an optimization method for weight coefficients. In the classic Bagging method, bootstrap resampling is used to generate different training sets, and the individual difference of the base classifier is increased, thereby improving the generalization ability. Additionally, Jin et al. [47] proposed in 2018 that the majority of voting methods are used in the traditional Bagging algorithm to make decisions, and the performance differences of the base classifiers are ignored; this results in the superior and inferior classifiers all having the same decision-making power.

G. OTHERS

To make a representative comparison between different classifier families and individuals, and to consider other classifiers that are not specifically discussed, three individual classifiers belonging to different families are introduced to make the comparison more comprehensive and balanced. The three individual classifiers are used for the experimental comparison discussed in Section IV. The specific parameters and configurations of the classifiers are included in Appendix A.

It is worth mentioning that although the selected 3 individual classifiers are relatively unpopular, they show excellent classification results in other classifier families. Therefore, these 3 individual classifiers can represent the best performance of other classifier families. However, since these three individual classifiers come from different families, we cannot consider them as whole to compare with other families for the sake of comprehensive consideration.

IV. INDIVIDUAL CLASSIFIERS ANALYSIS

A. METRICS

1) ACCURACY

In machine learning, three parameters reflect the results for the samples and are respectively predicted on the basis of precision, recall, and accuracy. Suppose there are two types in the original sample, wherein:

There is a total of P samples of category 1, and category 1 is assumed as a positive example.

There is a total of N samples of category 0, and category 0 is assumed as a negative example.

After classification:

TP is the samples with category 1 that were correctly identified as category 1 by the system, and FN is the samples with category 1 that were misjudged as category 0 by the system, Obviously $P = TP + FN$.

FP is the samples with category 0 that were correctly identified as category 1 by the system, and TN is the samples with category 0 that were misjudged as category 0 by the system, Obviously $N = FP + TN$.

The precision is defined as:

$$P = \frac{TP}{TP + FP}, \tag{19}$$

which reflects the proportion of real positive samples in the positive cases determined by the classifier. Recall is defined as

$$R = \frac{TP}{TP + FN} = 1 - \frac{FN}{T}, \tag{20}$$

which reflects the proportion of positive cases correctly classified to total positive cases. Finally, accuracy is defined as

$$A = \frac{TP + TN}{P + N} = \frac{TP + TN}{TP + FN + FP + TN}, \tag{21}$$

which reflects the classifier's ability to judge the entire sample, i.e., a positive decision can be positive, and negative determination is negative.

The accuracy rate reflects the magnitude of the judgement error between the result of the systematic determination by a group of samples and the original result. Because the training set in this study had only positive samples, and because the precision is equal to the accuracy, the accuracy was used as one of the indicators for whether the classifier correctly classified all the signals in the subsequent training and testing that were conducted.

2) KAPPA COEFFICIENT

In experiments, researchers generally need to investigate whether the different methods present consistency in results; the Kappa coefficient is therefore proposed to effectively measure this consistency. The Kappa coefficient is calculated based on the confusion matrix, which is a good way to change the direction of the grid and describe the changes of the data over time. However, the confusion matrix does not describe the degree of change from a statistical sense; therefore, there must be a way, namely the Kappa coefficient, to measure changes in the nominal variables in statistical methods.

Kappa coefficient differences are mainly used in the comparative analysis of two things; "accidental" factors or "inevitable" factors are often used to check the correctness of the extent of the test results to determine the real feature. The Kappa coefficient is a measure of the consistency of the results of statistical measurement, and is able to calculate the overall coherence and consistency of the classification index for consistency tests. It can also be used to measure the accuracy of the classification.

The value k is calculated as follows:

$$k = \frac{p_o - p_e}{1 - p_e}, \tag{22}$$

where p_o is the sum of the number of correctly classified samples for each class divided by the total number of samples, i.e., the overall classification accuracy.

The Kappa coefficient is calculated to be within -1 to 1, but usually falls between 0 and 1. It can be divided into five groups to indicate the consistency of different levels: 0.0-0.20 indicates very low consistency, 0.21-0.40 indicates general consistency, 0.41-0.60 indicates medium consistency, 0.61-0.80 indicates high consistency, and 0.81-1 indicates almost identicalness.

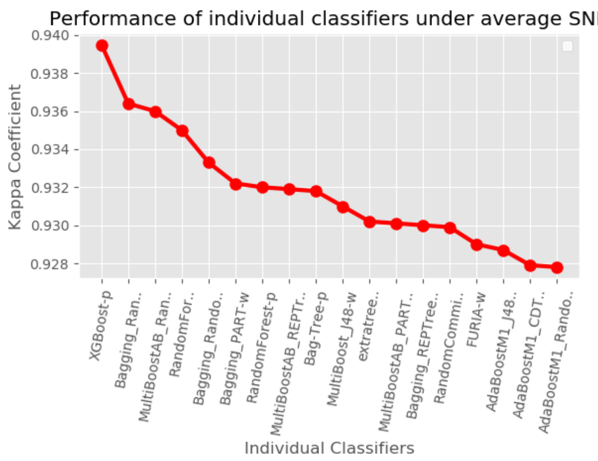


FIGURE 2. The top 18 classifiers with the best classification effect under the average SNR. Obviously, the individual classifiers with the best performance are mainly from the Boosting, Bagging and RF families. Among them, the top three are xgboost-p, Bagging-RandomForest-w and MuliBoostAB-RandomForest-w.

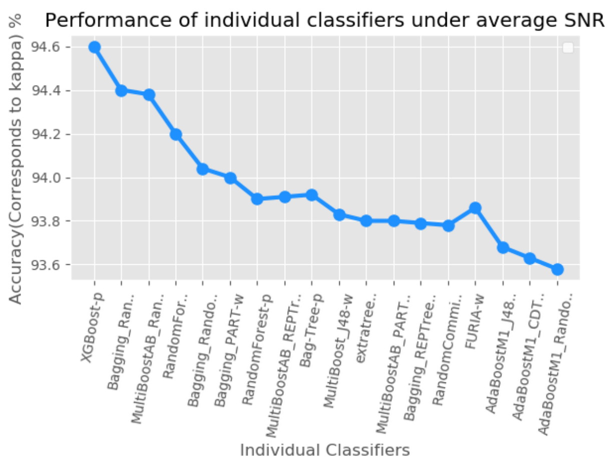


FIGURE 3. The top 18 classifiers with the best classification effects under the average SNR. In the ranking under the Kappa indicator, the overall trend of the individual classifiers with the best performance is basically consistent with the ranking of the accuracy rate. FURIA-w is an exception, and its accuracy ranking has been slightly improved. This also reveals the importance of using a comprehensive indicator evaluation.

B. PERFORMANCE ANALYSIS

To better compare the classification effects of the six families of classifiers under different SNRs, the classification effects of different classifiers were first analyzed at −10 dB, −5 dB, 0 dB, and 2 dB. Finally, the Kappa coefficient and accuracy of each classifier and family under the average SNR were obtained.

First, the top 18 classifiers with the best classification were analyzed and visualized. Figure 2 presents the performance analysis with the average Kappa coefficient as an indicator.

When the average Kappa coefficient is used as an index, 8 of the top 18 best classification results belong to the Boosting family, namely xgboost-p, MultiBoost AB-RandomForest-w, MultiBoostAB-REPTree-w, Multi BoostAB-PART-w, MultiBoostAB-J48-w, AdaBoost M1-J48-w, AdaBoostM1-CDT-w, and AdaBoostM1-Random SubSpace-w. As mentioned previously, the base classifiers of these Boosting methods have very good classification capabilities and are strong classifiers; they can therefore exert superior classification effects under comprehensive conditions. There are also 5 classifiers that belong to the Bagging family, namely Bagging-RandomForest-w, Bagging-RandomTree-w, Bagging-PART-w, bag-tree-p, and Bagging-REPTree-w. This demonstrates that the Boosting and Bagging families are the best classifiers, and present optimal performance on the entropy feature dataset.

It is worth noting that the RF family, which is one of the oldest methods, has a better classification effect than other new classifiers. It ranks in the 5th and 8th places in the top 18, which demonstrates that the RF family can better classify this dataset. The remaining classifiers in the top 18 are extratree-p, RandomCommittee-w, and Decorate-w. Of course, due to the large size and variability of the dataset, there may be some errors in the recognition accuracy and Kappa coefficient in the test, but it is always within a reasonable range.

To more intuitively compare the measurement effects of the two different indicators of Kappa coefficient and recognition accuracy, Figure 3 presents a schematic diagram of the classification effect of the classifier with accuracy as the vertical axis under the same abscissa as in Figure 2.

It can be seen from Figure 3 that under the average SNR, the accuracy and the sorting effect of Kappa as the index are basically consistent, i.e., the same measurement effect on the classification performance of different classifiers is obtained. However, it can be found that FURIA-w has increased, i.e., its accuracy is much higher than that of the surrounding classifiers, and the measurement effect is different from the Kappa coefficient. The reason for this is that the Kappa coefficient represents the ratio of the error reduction between the classification and the completely random classification, and its essence is the ratio of the actual consistency to the non-opportunity consistency. However, the accuracy rate reflects the magnitude of the error between the results of a set of samples and the original results. Therefore, considering the method of calculating the difference between the two, and because the number and variability of the dataset is relatively large, the accuracy of FURIA-w is thus increased.

In addition, in general, the RF family, represented by RandomForest-w, and the integrated classifier, represented by MultiBoostAB, Bagging, and AdaBoostM1, always maintain superior performance under different SNRs. These classifiers achieve good classification results whether they are analyzed in terms of the Kappa coefficient or accuracy, reflecting the superiority and universal applicability of their family classifiers. Therefore, these results can be combined to choose the best individual classifier in different application scenarios.

V. CLASSIFIER FAMILIES ANALYSIS

To measure the classification performance of different classifier families as a whole, and to compare and analyze them more comprehensively, Figures 4 and 5 show the box plots of the classifier families under the average SNR condition.

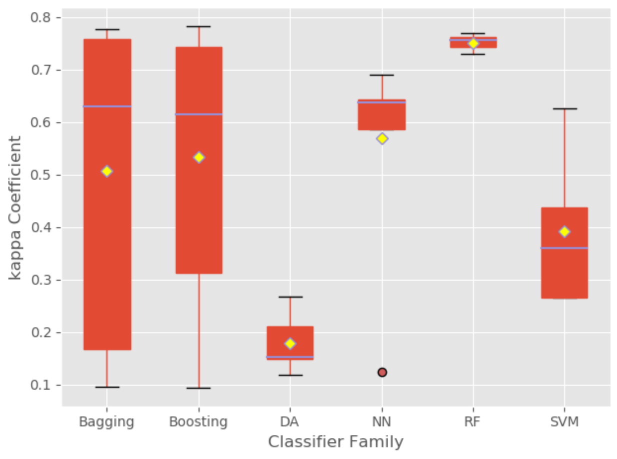


FIGURE 4. Box diagram of different classifier families at -10dB. It can be seen that RF has the largest mean, and the smallest individual difference. DA has the worst performance and has the lowest kappa value. In addition, due to the large number of Boosting and Bagging classifiers, they have large maximum and minimum differences.

There are three main parts in each box plot: the average of the Kappa coefficients of the classifiers in each family (denoted by a black square), and the maximum and minimum values of the Kappa coefficient. The highest horizontal line of each box plot represents the maximum value of the Kappa coefficient in the family, corresponding to the best-performing classifier in the family, and the lowest horizontal line represents the minimum Kappa coefficient, corresponding to the worst-performing classifier in the family. By observing the maximum, minimum, and mean values of the box plot, and the spacing between the maximum and minimum values, each family can be evaluated and analyzed comprehensively.

Figure 4 is a box plot of six families at -10 dB. It can be seen that RF is the best-performing classifier family; it has the largest mean value and a very small interval between the maximum and minimum values, which means that the classifiers in the RF family have smaller differences, i.e., each classifier has a better performance. Stacking is the worst performer among all families, and its mean and maximum Kappa coefficients are both zero. The maximum and minimum values of the box plots of the Bagging and Boosting families are almost the same, but the mean value of Boosting is slightly higher than that of Bagging. This demonstrates that, compared with the Bagging family, the Boosting family has a higher proportion of classifiers with superior performance, thus raising the average of the entire Boosting family. In the box diagram of the SVM, the mean value is in the middle and lower part of the interval line segment, which indicates that there is a large proportion of classifiers with poor performance in the family. In the box plot of the NN methods, the mean is at the top of the interval, indicating that most of the classifiers in this family perform well, so the mean is closer to the maximum of the Kappa coefficient in the classifier family.

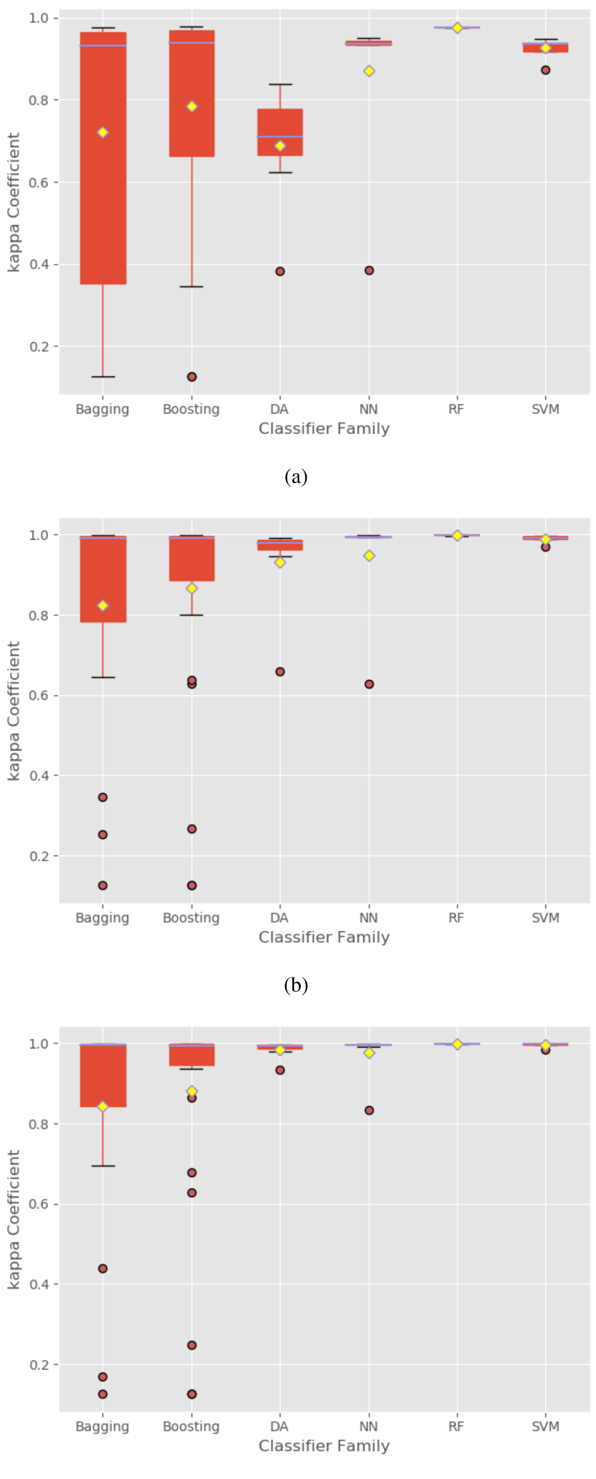


FIGURE 5. Box diagrams of different classifier families at −5 dB, 0 dB, and 2 dB. The overall performance of the RF is always optimal, with the highest Kappa coefficient and the smallest individual difference. The SVM, DA, and NN families perform poorly at a low SNR, but as the SNR increases, their overall performance is greatly improved, especially at 2 dB, and is comparable to the RF family. Additionally, there are always individual classifiers that pull down the overall performance of the Bagging and Boosting families, and their Kappa coefficients do not improve with the increase of SNR.

The three subfigures in Figure 5 present box plots for different classifier families at -5 dB, 0 dB, and 2 dB, respectively. As the SNR increases, a series of changes occurs in the box plots of each family. The average of the box plots of the RF family is also constantly increasing. Additionally, the box plot still maintains a very small interval, indicating that the performance of all classifiers in the family improves as the SNR changes. The Bagging and Boosting families are somewhat different from the RF family. Both the mean and maximum values increase with the increase of SNR, indicating that the performance of most of classifiers in these families is better with the increase of the SNR. However, the minimum of the two families does not increase with the increase of SNR, which means that there are always some classifiers that have very poor performance. This difference is reflected in two aspects: first, the recognition accuracy and Kappa coefficients of these classifiers are very low; second, the performance of these classifiers does not improve as the SNR increases.

Combining the box plots of all SNRs, the SVM, DA, and NN families exhibit the most obvious performance with the improvement of SNR. First, with the improvement of SNR, the mean, maximum, and minimum values of the Kappa coefficients of the three families continually increase. Secondly, the intervals of the three family box plots continuously decrease as the SNR is increased. These two points illustrate the performance of these three series of classifiers, especially those that are not effective at a low SNR, but that increase significantly as the SNR increases.

VI. CONCLUSION

With the rapid development of artificial intelligence, robotic swarms have been widely and highly regarded as one of the most popular branches of the field. To improve the communication quality, effectively identify and classify the modulated signals, and provide a good communication environment and technical support for robotic swarms, this paper studied the classification and recognition effects of different families of classifiers. Specifically, for the 9 different types of modulation signals, a hybrid entropy feature dataset was constructed by using the selected four entropy features. Based on this dataset, the performances of 77 classifiers from different series were then evaluated.

From the perspective of individual classifiers, the experimental results show that Boosting, Bagging, and RF are the top three classifier families. Among the top 18 classifiers with the best performances, Boosting, Bagging, and RF account for 8, 5, and 2 classifiers, respectively. In addition, the FURIA's Kappa coefficient ranking is different from its accuracy rate ranking, which results in the higher ranking of FURIA's performance. This also illustrates the importance of using a comprehensive indicator to measure and compare the performance of the classifier. From the perspective of family performance, whether the SNR is -10 dB, -5 dB, 0 dB, or 2 dB, RF always maintains the best performance, i.e., it has the largest mean and the smallest variance, and its overall performance is excellent. The performance of RF is followed by those of SVM and NN; although they perform poorly at -10 dB, their mean values gradually increase and their variances decrease as the SNR increases, demonstrating that their overall classification performances are improved with the increase of SNR. The Bagging and Boosting families have many members, so although their mean and maximum values increase with the increase of SNR, the minimum value does not change significantly; thus, there are always individual classifiers with very poor performance, which seriously lowers the overall performance.

The work of this paper also has some limitations. For example, we currently focus on the comparison and research of the most popular family classifiers at present, but ignore some of the classifiers that are very popular but have excellent classification performance. Moreover, our current comparison focuses on considering accuracy, but ignores the evaluation of the robustness of the classifier. In the follow-up work, we will further expand the research on individual classifiers, add some unpopular classifiers, and conduct more comprehensive and thorough exploration and evaluation.

APPENDIX A DETAILS OF INDIVIDUAL CLASSIFIERS

The specific parameters and configurations of the individual classifiers of the different families involved are as follows.

A. DA

- *LDA-eigen-p:* uses the LDA function in sklearns, and the solver is set to eigen.
- *LDA-lsqr-p:* LDA is solved by the least squares method (Ye et al., 2007). On-line DA is conducted using the least squares method (Wang et al., 2012).
- *LDA-svd-p:* uses the LDA function in sklearns; the solver is set to svd, even if it is solved by singular value decomposition. LDA is based on the kernel function and singular value decomposition (Park et al., 2005).
- *Quadratic Discriminant-p:* uses the quadratic discriminant function in sklearns. The quadratic discriminant method is used for classification of wrist EMG signals (Kang et al., 2011).
- *DA-diaglinear-m:* uses the fitcdiscr function in MATLAB; the discriminant mode is set to diaglinear.
- *DA-quadratic-m:* uses the fitcdiscr function in MATLAB; the discriminant mode is set to quadratic.
- *DA-linear-m:* uses the fitcdiscr function in MATLAB; the discriminant is set to linear. Local linear embedding and the linear DA algorithm are used for face recognition (Zhang et al., 2004).

B. SVM

- *LibSVM-Linear-w:* uses LibLinear library (Fan et al., 2008) to classify large-scale linear high-dimensional classifications. It uses the L2 loss (dual) solver, with parameters C = 1, tolerance = 0.01, deviation = 1.
- *LibSVM-Polynomial-w:* uses the Polynomial library for large-scale linear high-dimensional classification, and the parameter settings remain unchanged.

- *LibSVM-Radialbasis-w:* uses the Radialbasis library for large-scale linear high-dimensional classification, and the parameter settings remain unchanged.
- *LibLINEAR-w:* uses the LibLinear library (Fan et al., 2008) to classify large-scale linear high-dimensional classifications. It uses the L2 loss (dual) calculator, and parameters C = 1, tolerance = 0.01, deviation = 1.
- *SM-w:* a one-to-one classification method using sequential minimal optimization (Platt, 1998); C = 1, tolerance L = 0.001, rounding error 10-12. Data normalization and quadratic kernel SVM are used.
- *SVM-linear-p:* uses linear kernels; the penalty coefficient C of the error term is set to 1.
- *SVM-poly-p:* Kernel functions call polynomial kernels.
- *SVM-rbf-p:* calls the SVC function in sklearns. The type of the kernel function is set to the Gauss kernel function, the penalty coefficient C is set to 1, and the coefficient gamma of the Gauss kernel is set to 0.125.

C. NN

- *Mutilayer-Perceptron-w:* an MLP network with S-shaped hidden neurons. It has unlimited linear output neurons. Its learning rate is 0.3, momentum is 0.2, training period is 500, and the hidden neurons are equal.
- *Mutilayer-Perceptron-CS-w:* a classifier that uses back propagation to classify instances. The learning rate is 0.3, the momentum is 0.2, the training period is 500, and the hidden neurons are equal.
- *RBFNetwork-w:* uses K-means to select RBF centers and linear regression to learn classification functions, and uses symmetric multivariate Gauss and normalized input. Clustering (or hidden neurons) with a number equal to half of the training patterns was used in the present study, ridge = 10^{-8}.
- *MLPClassifier-relu-p:* uses one layer of a hidden layer; the number of neurons in the hidden layer is set to 10, the ReLU function is used as the activation function and L2 regularization, and random gradient descent is used as the optimization method. The maximum number of iterations is set to 1000, and the learning rate is 0.05.
- *MLPClassifierlogistic-p:* calls MLPClassifier in sklearns, uses the logistic function as an activation function, uses one layer of hidden layer. The number of hidden layer neurons is set to 10, random gradient descent is used as the optimization mode, the maximum number of iterations is set to 1000, and the learning rate is set to 0.05.

D. RF

- *Random Forest-w:* (Leo Breiman, 2001) uses blog C inputs and infinite depth trees to implement forests with 500 random tree basic classifiers.
- *Rotation Forest-w:* (Rodr guez et al., 2006) uses J48 as the basic classifier, the principal component analysis filter, and three sets of inputs; the pruning confidence C = 0.25, and there are two modes per leaf.
- *Random Forest-p:* calls the RF function in sklearns, sets the number of base classifiers to 10, and uses the Gini coefficient as the splitting index of the base classifiers.

E. BOOSTING

- *LogitBoost-w:* uses additive logic regression (decision stump) as the basic learner. Without cross-validation, it conducts 100% quality basic training and internal cross-validation once. The possibility of improving the threshold is 1.79, the shrinkage parameter is 1, and the number of iterations is 10.
- *Raced Incremental LogitBoost-w:* a competing Logitboost Committee, in which incremental learning and decision stump basic classifiers, ranging in size from 500 to 2000, utilize 1000 verification sets and logarithmic likelihood pruning.
- *Adaboost-tree-p:* uses the AdaBoost algorithm in sklearns; the decision tree is selected as the base classifier, and the number of base classifiers is set to 50.
- *Adaboost-tree-m:* uses the AdaBoost.M2 algorithm in MATLAB, and 50 decision trees are upgraded.
- *LPBoost-tree-m:* uses the decision tree as the base classifier, the maximum number of tree splitting is no more than 6 times, and the LPBoost algorithm in MATLAB is called.
- *RUSBoost-tree-m:* he base classifier is set to 50, which is upgraded using a decision tree.
- *TotalBoost-tree-m:* 50 decision trees are upgraded to be more robust than LPBoost.
- *GBDT-p:* the number of base classifiers is set to 50 and the learning rate is 0.3. The Friedman mean square error is used as the node splitting index of the decision tree.
- *XGboost-p:* calls the XGboost function package and uses the decision tree with the maximum depth of 4 as the base classifier. The shrinkage step is set to 0.3.
- *AdaBoostM1-Decision Stump-w:* uses the decision stump basic classifier to implement the same AdaBoost. M1 method.
- *AdaBoostM1-J48-w:* set of Adaboost.M1 that combines J48 basic classifiers.
- *AdaBoostM1-RandomSubSpace-w:* an Adaboost.M1 set that combines RandomSubSpace basic classifiers.
- *AdaBoostM1-RandomTree-w:* an AdaBoost.M1 set that combines RandomTree basic classifiers.
- *AdaBoostM1-REPTree-w:* a set of AdaBoost.M1 that combines REPTree basic classifiers.
- *AdaBoostM1-CDT-w:* an AdaBoost.M1 set that combines CDT basic classifiers.
- *AdaBoostM1-ExtraTree-w:* a set of AdaBoost.M1 that combines ExtraTree basic classifiers.
- *AdaBoostM1-HoeffdingTree-w:* a set of AdaBoost.M1 that combines the basic classifiers of HoeffdingTree.
- *AdaBoostM1-LADTree-w:* a set of AdaBoost.M1 that combines the basic classifiers of LADTree.
- *MuliBoostAB-Decision Stump-w:* a set of MultiBoost. It uses the decision stump basic classifier,

three sub-committees, 10 training iterations, and 100% quality to perform basic training based on AdaBoost and Wagging. The following MultiBoostAB ensemble uses the same option.

- *MuliBoostAB-DecisionTable-w:* combines MultiBoost and DecisionTable, both of which have the same parameter settings as above.
- *MuliBoostAB-IBK-w:* uses MultiBoostAB with IBK basic classifiers.
- *MuliBoostAB-J48-w:* trains a set of J48 decision trees using pruning confidence C = 0.25 and two training modes per leaf.
- *MuliBoostAB-LibSVM-w:* uses the LibSVM-based classifier with the best C selected by the SVM-C classifier and the Gauss kernel distribution. It is compared with previous reports, although strong classifiers are not recommended as base classifiers in principle.
- *MuliBoostAB-Logistic-w:* is combined with the Logistic base classifier.
- *MuliBoostAB-Multilayer Perceptron-w:* uses the same basic MLP classifier as Multilayer Perceptron-w (another strong classifier).
- *MuliBoostAB-NaiveBayes-w:* uses the NaiveBayes basic classifier.
- *MuliBoostAB-OneR-w:* uses OneR basic classifier.
- *MuliBoostAB-PART-w:* is combined with the basic classifier of PART.
- *MuliBoostAB-RandomForest-w:* is combined with a RF base classifier. Although RF itself is a whole, it does not seem to be very useful to compare the MultiBoostAB sets of RF sets.
- *MuliBoostAB-RandomTreew:* uses a random tree with the same options as above.
- *MuliBoostAB-REPTree-w:* uses the REPTree basic classifier.

F. BAGGING

- *Bagging-Decision Stump-w:* uses 10 bagging iterations of decision stump basic classifiers.
- *Bagging-DecisionTable-w:* uses DecisionTable and BestFirst, and searches forward for input selection, retaining one validation and maximizing accuracy.
- *Bagging-HyperPipes-w:* uses the HyperPipes basic classifier.
- *Bagging-IBK-w:* uses the IBK-based classifier, which uses the cross-validation of linear NN search and Euclidean distance to develop KNN classification adjustment K.
- *Bagging-J48-w:* the basic classifiers of J48.
- *Bagging-LibSVM-w:* used for the Gauss kernel of LibSVM and has the same options as a single LibSVM classifier.
- *Bagging-Logistic-w:* has infinite iterations and logarithmic likelihood ridges 10^{-8} in the Logistic basic classifier.
- *Bagging-LWL-w:* uses a local weighted learning basic classifier with linear weighted kernel shape and Decision Stump basic classifier.
- *Bagging-Multilayer Perceptron-w:* has the same configuration as a single Multilayer Perceptron-w.
- *Bagging-NaiveBayes-w:* has a NaiveBayes classifier.
- *Bagging-OneR-w:* uses the OneR basic classifier, with at least six objects per Bucket.
- *Bagging-PART-w:* has at least two training modes per leaf, trimming confidence C = 0.25.
- *Bagging-Random Forest-w:* has 500 trees of forest, infinite tree depth and blog C inputs.
- *Bagging-RandomTree-w:* uses random tree basic classifier without backfilling. blog2 C random input, infinite tree depth and two training modes per leaf are investigated.
- *Bagging-REPTree-w:* uses REPTree with two patterns per leaf, with a minimum category difference of three times, reducing error pruning and infinite tree depth.
- *Bagging-knn-p:* uses KNN as the base classifier, n-neighbors = 5, and set the number of base classifiers to 10.
- *Bagging-tree-p* 10 Bagging iterations were performed on decision tree classifiers using Gini coefficients as splitting indices.
- *Bagging-logistic-p:* the logistic regression classifier is used as the base classifier for 10 Bagging iterations, in which the logistic regression classifier uses L2 regularity and the logarithmic likelihood loss is not higher than that.
- *Bagging-tree-m:* set the maximum number of splits of the decision tree to no more than 6.

G. OTHERS

- *Extratree-p:* integrates 10 decision tree-based classifiers.
- *RandomCommittee-w:* is a collection of random trees (each tree is constructed with a different seed) whose output is the average of the basic classifier output
- *FURIA-w:* is short for "Fuzzy Unordered Rule Induction Algorithm", it learns fuzzy rules instead of conventional rules and unordered rule sets instead of rule lists.

REFERENCES

[1] A. A. Khan, M. H. Rehmani, and M. Reisslein, "Cognitive radio for smart grids: Survey of architectures, spectrum sensing mechanisms, and networking protocols," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 860–898, 1st Quart., 2016.

[2] Y. Lin, X. Zhu, Z. Zheng, Z. Dou, and R. Zhou, "The individual identification method of wireless device based on dimensionality reduction," *J. Supercomput.*, vol. 75, no. 6, pp. 3010–3027, Jun. 2019.

[3] T. Liu, Y. Guan, and Y. Lin, "Research on modulation recognition with ensemble learning," *EURASIP J. Wireless Commun. Netw.*, vol. 2017, no. 1, p. 179, 2017.

[4] Y. Tu, Y. Lin, J. Wang, and J.-U. Kim, "Semi-supervised learning with generative adversarial networks on digital signal modulation classification," *Comput. Mater. Continua*, vol. 55, no. 2, pp. 243–254, 2018.

[5] C. Shi, Z. Dou, Y. Lin, and W. Li, "Dynamic threshold-setting for RF-powered cognitive radio networks in non-Gaussian noise," *EURASIP J. Wireless Commun. Netw.*, vol. 2017, no. 1, p. 192, Nov. 2017.

[6] Z. Zhang, X. Guo, and Y. Lin, "Trust management method of D2D communication based on RF fingerprint identification," *IEEE Access*, vol. 6, pp. 66082–66087, 2018.

[7] H. Wang, J. Li, L. Guo, Z. Dou, Y. Lin, and R. Zhou, "cFractal complexity-based feature extraction algorithm of communication signals," *Fractals*, vol. 25, no. 4, pp. 1740008-1–1740008-3, Jun. 2017.

[8] J. Zhang, S. Chen, X. Mu, and L. Hanzo, "Evolutionary-algorithm-assisted joint channel estimation and turbo multiuser detection/decoding for OFDM/SDMA," *IEEE Trans. Veh. Technol.*, vol. 63, no. 3, pp. 1204–1222, Mar. 2014.

[9] J. Zhang, S. Chen, X. Guo, J. Shi, and L. Hanzo, "Boosting fronthaul capacity: Global optimization of power sharing for centralized radio access network," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1916–1929, Feb. 2019.

[10] J. Wen, Z. Zhou, Z. Liu, M.-J. Lai, and X. Tang, "Sharp sufficient conditions for stable recovery of block sparse signals by block orthogonal matching pursuit," *Appl. Comput. Harmon. Anal.*, vol. 47, pp. 948–974, Nov. 2019.

[11] J. Wen and X.-W. Chang, "On the KZ reduction," *IEEE Trans. Inf. Theory*, vol. 65, no. 3, pp. 1921–1935, Mar. 2019.

[12] Z. Dou, C. Shi, Y. Lin, and W. Li, "Modeling of non-gaussian colored noise and application in cr multi-sensor networks," *EURASIP J. Wireless Commun. Netw.*, vol. 2017, no. 1, p. 192, Nov. 2017.

[13] M. Liu, J. Zhang, Y. Lin, Z. Wu, B. Shang, and F. Gong, "Carrier frequency estimation of time-frequency overlapped MASK signals for underlay cognitive radio network," *IEEE Access*, vol. 7, pp. 58277–58285, 2019.

[14] Y. Lin, Y. Li, X. Yin, and Z. Dou, "Multisensor fault diagnosis modeling based on the evidence theory," *IEEE Trans. Rel.*, vol. 67, no. 2, pp. 513–521, Jun. 2018.

[15] H. Wang, L. Guo, Z. Dou, and Y. Lin, "A new method of cognitive signal recognition based on hybrid information entropy and DS evidence theory," *Mobile Netw. Appl.*, vol. 23, no. 4, pp. 677–685, Aug. 2018.

[16] W. Wei and J. M. Mendel, "Maximum-likelihood classification for digital amplitude-phase modulations," *IEEE Trans. Commun.*, vol. 48, no. 2, pp. 189–193, Feb. 2000.

[17] A. Swami and B. M. Sadler, "Hierarchical digital modulation classification using cumulants," *IEEE Trans. Commun.*, vol. 48, no. 3, pp. 416–429, Mar. 2000.

[18] N. Mahmoudi and E. Duman, "Detecting credit card fraud by modified Fisher discriminant analysis," *Expert Syst. Appl.*, vol. 42, no. 5, pp. 2510–2516, Apr. 2015.

[19] L. Wu, C. Shen, and A. van den Hengel, "Deep linear discriminant analysis on Fisher networks: A hybrid architecture for person re-identification," *Pattern Recognit.*, vol. 65, pp. 238–250, May 2017.

[20] S. Srivastava and M. R. Gupta, "Distribution-based Bayesian minimum expected risk for discriminant analysis," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2006, vol. 1, no. 1, pp. 2294–2298.

[21] W. Chao, "Bayesian discriminant analysis for prediction of coal and gas outbursts and application," *Int. J. Mining Sci. Technol.*, vo1. 20, no. 4, pp. 520–523, Jul. 2010.

[22] R. Mileris, "Estimation of loan applicants default probability applying discriminant analysis and simple Bayesian classifier," *Econ. Manage.*, vo1. 1, no. 1, pp. 1078–1084, May 2010.

[23] J. Zhang, "Multiple Bayesian discriminant functions for high-dimensional massive data classification," *Data Mining Knowl. Discovery*, vol. 31, no. 2, pp. 465–501, Mar. 2017.

[24] I. Omara, "Learning pairwise SVM on deep features for ear recognition," in *Proc. Int. Conf. Comput. Inf. Sci.*, May 2017, vol. 1, no. 1, pp. 341–346.

[25] M. Elleuch, R. Mokni, and M. Kherallah, "Offline Arabic handwritten recognition system with dropout applied in deep networks based-SVMs," in *Proc. Int. Joint Conf. Neural Netw.*, Jul. 2016, vol. 1, no. 1, pp. 3241–3248.

[26] E. E. Bron, M. Smits, W. J. Niessen, and S. Klein, "Feature selection based on the SVM weight vector for classification of dementia," *IEEE J. Biomed. Health Inform.*, vol. 19, no. 5, pp. 1617–1626, Sep. 2015.

[27] X. Wang, "Lithology identification using an optimized KNN clustering method based on entropy-weighed cosine distance in Mesozoic strata of Gaoqing field, Jiyang depression," *J. Petroleum Sci. Eng.*, vol. 166, no. 1, pp. 157–174, Jul. 2018.

[28] D. P. Vivencio, E. R. Hruschka, M. do Carmo Nicoletti, E. B. dos Santos, and S. D. C. O. Galvao, "Feature-weighted k-nearest neighbor classifier," in *Proc. IEEE Symp. Found. Comput. Intell.*, Apr. 2007, vol. 1, no. 1, pp. 481–486.

[29] J. Huang, Y. Wei, J. Yi, and M. Liu, "An improved kNN based on class contribution and feature weighting," in *Proc. 10th Int. Conf. Measuring Technol. Mechatronics Automat. (ICMTMA)*, Feb. 2018, vol. 1, no. 1, pp. 313–316.

[30] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 210–227, Feb. 2009.

[31] S. W. Kwok and C. Carter, "Multiple decision trees," *Mach. Intell. Pattern Recognit.*, vol. 1, no. 1, pp. 327–335, 1990.

[32] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 8, pp. 832–844, Aug. 1998.

[33] J. J. Rodriguez, L. I. Kuncheva, and C. J. Alonso, "Rotation forest: A new classifier ensemble method," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 10, pp. 1619–1630, Oct. 2006.

[34] Y. Ye, Q. Wu, J. Z. Huang, M. K. Ng, and X. Li, "Stratified sampling for feature subspace selection in random forests for high dimensional data," *Pattern Recognit.*, vol. 46, no. 3, pp. 769–787, Mar. 2013.

[35] T. T. Nguyen, H. Zhao, and J. Z. Huang, "A new feature sampling method in random forests for predicting high-dimensional data," *Adv. Knowl. Discovery Data Mining*, vol. 1, no. 1, pp. 459–470, May 2015.

[36] Y. Wang and S. T. Xia, "A novel feature subspace selection method in random forests for high dimensional data," in *Proc. Int. Joint Conf. Neural Netw.*, Jul. 2016, vol. 1, no. 1, pp. 4383–4389.

[37] Y. Wang, "Bernoulli random forests: Closing the gap between theoretical consistency and empirical soundness," in *Proc. Int. Joint Conf. Artif. Intell.*, vol. 1, no. 1, pp. 2167–2173, Jan. 2016.

[38] B. S. Rota and P. Kontschieder, "Neural decision forests for semantic image labelling," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, vol. 1, no. 1, pp. 81–88.

[39] Y. Freund and E. Robert, "A decision-theoretic generalization of online learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, Feb. 1997.

[40] P. K. Mallapragada, R. Jin, A. K. Jain, and Y. Liu, "SemiBoost: Boosting for semi-supervised learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 11, pp. 2000–2014, Nov. 2009.

[41] C. Shen, P. Wang, F. Shen, and H. Wang, "*u*Boost: Boosting with the Universum," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 4, pp. 825–832, Apr. 2012.

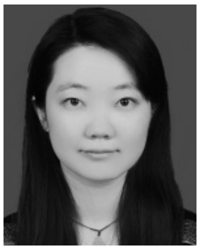
[42] Y. Chi and F. Porikli, "Classification and boosting with multiple collaborative representations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 8, pp. 1519–1531, Aug. 2014.

[43] R. Bryll, R. Gutierrez-Osuna, and F. Quek, "Attribute bagging: Improving accuracy of classifier ensembles by using random feature subsets," *Pattern Recognit.*, vol. 36, no. 6, pp. 1291–1302, Jun. 2003.

[44] Q.-T. Cai, C.-Y. Peng, and C.-S. Zhang, "A weighted subspace approach for improving bagging performance," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Mar./Apr. 2008, vol. 1, no. 1, pp. 3341–3344.

[45] Y. Qian, X. Guan, S. Wu, B. Yun, and D. Re, "An approach of Bagging ensemble based on feature set and application for traffic classification," *Telecommun. Sci.*, vol. 34, no. 4, pp. 41–48, Dec. 2018.

[46] Z. Jin, Y. Han, and Q. Zhu, "A sentiment analysis model with the combination of deep learning and ensemble learning," *J. Harbin Inst. Technol.*, vol. 50, no. 11, pp. 236–246, Jul. 2018.



QIAO TIAN was born in Harbin, China, in 1991. She received the B.S. degree from the College of Computer Science and Technology, Harbin Engineering University, Harbin, where she is currently pursuing the Ph.D. degree. Her main research interests include system parallel optimization, computer architecture, and communication.

HAOJUN ZHAO (Student Member, IEEE) received the B.S. degree from the College of Information and Communication Engineering, Harbin Engineering University, Harbin, China, in 2018, where he is currently pursuing the M.S. degree in communication engineering. His research interests include communication technology, machine learning, and security analysis.

YUN LIN received the B.S. degree from Dalian Maritime University, in 2003, the M.S. degree from the Harbin Institute of Technology, in 2005, and the Ph.D. degree from Harbin Engineering University, in 2010. From 2014 to 2015, he was a Visiting Scholar with Wright State University, Dayton, OH, USA. He is currently an Associate Professor with Harbin Engineering University. His current research interests include communication technology, signal processing, information fusion, cognitive radio, software-defined radio, and artificial intelligence.

FENGJUN XIAO received the B.S. degree in economics from Beihang University, in 2009, and the master's degree in technology policy under the supervision of Prof. S. Li, in 2014. He is currently pursuing the Ph.D. degree under the supervision of Prof. C. Li and began his research on the network security and emergency management.