UAV Flight State Recognition Using AC-GAN Based Method

Yaqing Xu[1], Jun Liang[1,2], Benkuan Wang[1,2], Datong Liu[1,2,*], Yu Peng[1,2], Xiyuan Peng[1,2]
1 Automatic Test and Control Institute, Harbin Institute of Technology, Harbin, China
2 School of Electronics and Information Engineering, Harbin Institute of Technology, Harbin, China
E-mail: liudatong@hit.edu.cn

***Abstract*—Unmanned Aerial Vehicle (UAV) has become a widely used tool in both military and civilian applications. As the UAV capabilities and technology complexity rapidly increase, UAV reliability and security are particularly essential. In order to improve the operational security, fault diagnosis, flight quality assessment and fuel consumption assessment are implemented. However, under different flight states, such as takeoff, climbing, level flight, etc. the possible failures, the meaning of flight quality and fuel consumption levels are different, so flight state recognition is the basis of the above work. Aiming at the problem that the flight time of various flight states is different, which leads to the imbalance of recognition samples, this paper proposes an Auxiliary Classifier Generative Adversarial Network (AC-GAN) based method. The method supplements samples by generating samples that enjoy the same distribution with the real data, thereby improving the accuracy of the recognition. It also segments the time series by sliding window method, to solve the classification problem of unequal time series. The performance of the proposed method is illustrated using actual UAV sensor data. The results show that the proposed method can effectively improve the recognition accuracy.**

Keywords-AC-GAN; flight state recognition; UAV

I. INTRODUCTION

UAV has been widely used in both military and civilian applications due to its unique advantages [1]. As UAV has developed rapidly, the UAV system becomes more and more complex, and their operational monitoring data are continuously accumulated. Based on monitoring data, condition monitoring [2]-[4], anomaly detection [5], fault diagnosis [6], and fuel consumption assessment could be realized. The flight, however, is a complicated process and it is hard to summarize the flight law from the complete UAV landing process. The possible failures and failure rates of UAVs under different flight states are different [7][8], for example, takeoff, approach and landing have the highest failure rate. In [9], it also refers that there is a large difference in fuel consumption levels under various flight states. A complete flight can be considered as a combination of a series of flight states. After the flight state is recognized, more targeted fault diagnosis, fuel consumption assessment, etc. can be performed. Therefore, flight state recognition is essential and necessary. Situation recognition and segmentation, on the one hand, can provide the necessary preprocessing for subsequent modeling of each state, the discovery of anomalous patterns, and analysis of external environment interference. On the other hand, it is possible to effectively evaluate the completion of the UAV mission by comparing every situation with the scheduled track.

The flight states are mainly divided into basic states, such as taking off, climbing, cruising, turning and landing, and the special states in the battle. One flight of the UAV contains a large amount of flight data, which contain lots of state parameters. They are the main basis for the ground station to monitor the state of the UAV. Based on these data, currently, existing methods for state recognition mainly fall in three categories: knowledge-based, classification-based and similarity-based.

The knowledge-based method is based on limited expert knowledge and in dynamic changing conditions, there is a problem of low recognition accuracy [10]. The similarity-based method measures the similarity of time series using Mahalanobis distance [11], grey correlation analysis, and dynamic time warping [12], etc. and then the sequence to be identified can be matched with the standard sequence. This method, however, is time-consuming and inefficient in calculating the distance between sequences. The classification-based method has become the most popular method, which includes neural network [13], Support Vector Machine (SVM) [14], Bayesian network [15], decision tree [16], etc. In [13], a fuzzy neural network based method is proposed for flight state recognition, considering the randomness and fuzziness of flight motion. In [10], the fuzzy least squares support vector machine is used to identify the flight modes. Those mentioned methods, however, do not consider the difference in flight time among different states of the UAV, so that the number of samples in different states differs significantly, resulting in unbalanced classification samples, which reduces recognition accuracy. The current main solutions to solve this problem in bi-classification include oversampling, under-sampling, threshold-moving and combined classifiers. In multi-classification problem, however, these methods are not very effective [17]. To mitigate this problem, this paper considers making for those small samples by generating samples with similar data characteristics. The Generative Adversarial Network (GAN) is the most popular method for generating samples in recent years [18], which enables the generated samples and the real samples share the same distribution. As a derivative model of the GAN framework, Auxiliary Classifier Generative Adversarial

This work is supported partly by Natural Science Foundation of China under Grant No. 61571160, 61803121, and 61771157.

978-1-7281-0860-5/19/$31.00 ©2019 IEEE

Network (AC-GAN) could implement multi-classification, so this paper utilizes the Auxiliary Classifier Generative Adversarial Network (AC-GAN) to generate small samples, and then classifies all samples by the auxiliary classifier. The proposed method can effectively solve the problem of unbalanced classification samples, which is suitable for UAV flight state recognition.

The remainder of this paper is organized as below. Section II illustrates the principle of AC-GAN algorithm. The proposed flight state recognition method is presented in Section III. Section IV presents a numerical experiment and provides the computational results to illustrate the effectiveness of the proposed method. Finally, conclusions and future work are summarized in Section V.

II. AC-GAN ALGORITHM

Generative Adversarial Network (GAN) is structurally inspired by the zero-sum game in game theory. A GAN framework consists of at least two parts, one is generator (G) and the other is discriminator (D) and both are usually formatted using a neural network. The generator captures the potential distribution of real data samples and generates new data samples, which are delivered to the discriminator together with those real data samples during the training period. The discriminator is a two-class classifier to identify the real samples and pick out the generated samples as accurately as possible, while the target of the generator is exactly opposite, which needs to reduce the probability of being identified by the discriminator as much as possible, thus forming a confrontation training mechanism. Then G and D form a mini-max game [19]. The loss function of GAN can be described as:

$$\min_G \max_D \{f(D,G) = \mathrm{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathrm{E}_{z\sim p_z(z)}[\log(1-D(G(z)))]\} \quad (1)$$

where x represents real data and z is random noise. p_{data} is the distribution of real data samples and $G(z)$ are generated data that obey real data distribution p_{data}, which are also called fake samples. The principle of GAN is shown in Fig. 1.

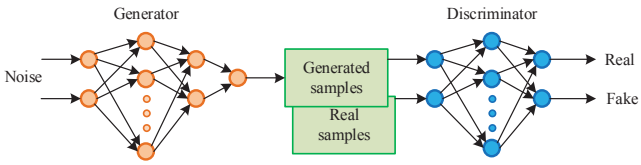

Figure 1. Flow chart of GAN

Generator and discriminator are constantly optimizing themselves during the constant iteration of the training process, so that the performance of the two can be improved together until reaching balance, which means the discriminator D cannot correctly discriminate the source of data, that is, when the fake samples are completely indistinguishable from the true samples, it is considered that the generator G has learned the sample distribution of the actual data.

As a derivative model of the GAN framework, Auxiliary Classifier GAN (AC-GAN), can implement multi-classification, which uses the label information of the input to generate corresponding samples. In the AC-GAN framework, except noise z, each generated sample has a corresponding label c and the generator utilizes both to generate samples $X_{fake} = G(c,z)$. The discriminator gives the probability distribution of the samples whether they are real or fake and their belonging categories $P(S \mid X)$, $P(C \mid X) = D(X)$. The objective function has two parts: the log likelihood L_s of the correct source and the log likelihood of the correct category L_c.

$$L_s = E[\log P(S = real \mid X_{real})] + E[\log P(S = fake \mid X_{fake})] \quad (2)$$

$$L_C = \mathrm{E}[\log P(C = c \mid X_{real})] + \mathrm{E}[\log P(C = c \mid X_{fake})] \quad (3)$$

where X_{real} means real samples and E() represents the mean value.

The discriminator outputs the probability of the corresponding category label, and then changes the loss function. Since it contains the likelihood of the real data source and the likelihood of the correct classification label, it increases the probability whether the category prediction is correct, no longer adjusting parameters only by backpropagation of the binary loss. The flow chart of AC-GAN is as follow:

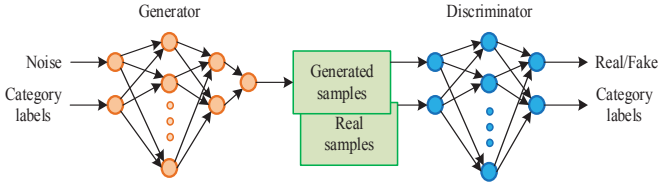

Figure 2. Flow chart of AC-GAN

The generator generates corresponding new data samples according to input sample labels $X_{fake} = \{\boldsymbol{x}_{fake}^k\}_{k=1}^K$, which are input to the discriminator together with the original samples $X_{real} = \{\boldsymbol{x}^m\}_{m=1}^M$, then the authenticity labels and category labels of the output could be obtained. Afterwards, the two labels errors are taken as loss functions, alternating iteration to train generators and discriminators and the process of generating samples is as follow:

The random vectors $\{\boldsymbol{z}^k\}_{k=1}^K$ sampled randomly from the Gauss noise are input to the generator, then mapped to the hidden layer vectors $\boldsymbol{h}_z^k$, and regenerated new samples $\{x_{fake}^k\}_{k=1}^K$ with corresponding class labels $\boldsymbol{y}_{fake}^k$. The expressions are as follows:

$$\boldsymbol{h}_z^k = f_{\theta_z}(\boldsymbol{W}_z \boldsymbol{z}^k + \boldsymbol{b}_z) \quad (4)$$

$$x_{fake}^{k} = f_{\theta_z'}(W_z' h_z^k + b_z') \tag{5}$$

where $\theta_z = \{W_z, b_z\}$, $\theta_z' = \{W_z', b_z'\}$ are the set of parameters from the input layer to the hidden layer and from the hidden layer to the output layer, respectively. W_z and W_z' are weight matrixes, while b_z and b_z' are bias matrixes. The activation function is the Relu function.

III. The Proposed Flight State Recognition Method

One flight process of UAV includes several flight states, including take-off, accelerated climb, uniform climb, level flight, left turn, right turn, descent, approach and landing, etc. By assigning digitized labels, the states are divided into 10 categories, and the labels are 0-9. Table I below lists the flight states of two flights and their corresponding flight time (this time refers to the sum of the flight time of the state in one flight). As illustrated from Table I, the UAV experienced 9 states from take-off to landing, but the flight time of these 9 states is quite different. For example, in 01 sorties, level flight experiences 450.4 seconds, while descent only lasts for 21.6 seconds. That is, the training samples of level flight are more than 20 times lower than those of descent. Obviously, the classification samples are not balanced, which will seriously affect the accuracy of classification.

TABLE I. Flight States And Labels

Flight states	labels	01 flight time(s)	02 flight time(s)
Waiting on the ground	0	3700	2120
Take off	1	3.2	3.2
Accelerated climb	2	54.4	87.2
Uniform climb	3	0	44
Level flight	4	450.4	1200.8
Left turn	5	61.6	288
Right turn	6	157.6	156
descent	7	21.6	112
approach	8	183.2	193.6
landing	9	27.2	43.2

To improve the accuracy, this paper utilizes the Generative Adversarial Network (GAN) to generate fake samples that share the same distribution as the real samples, and then inputs them into the discriminator network with auxiliary classifiers to realize state recognition. The framework of this method is as follow:

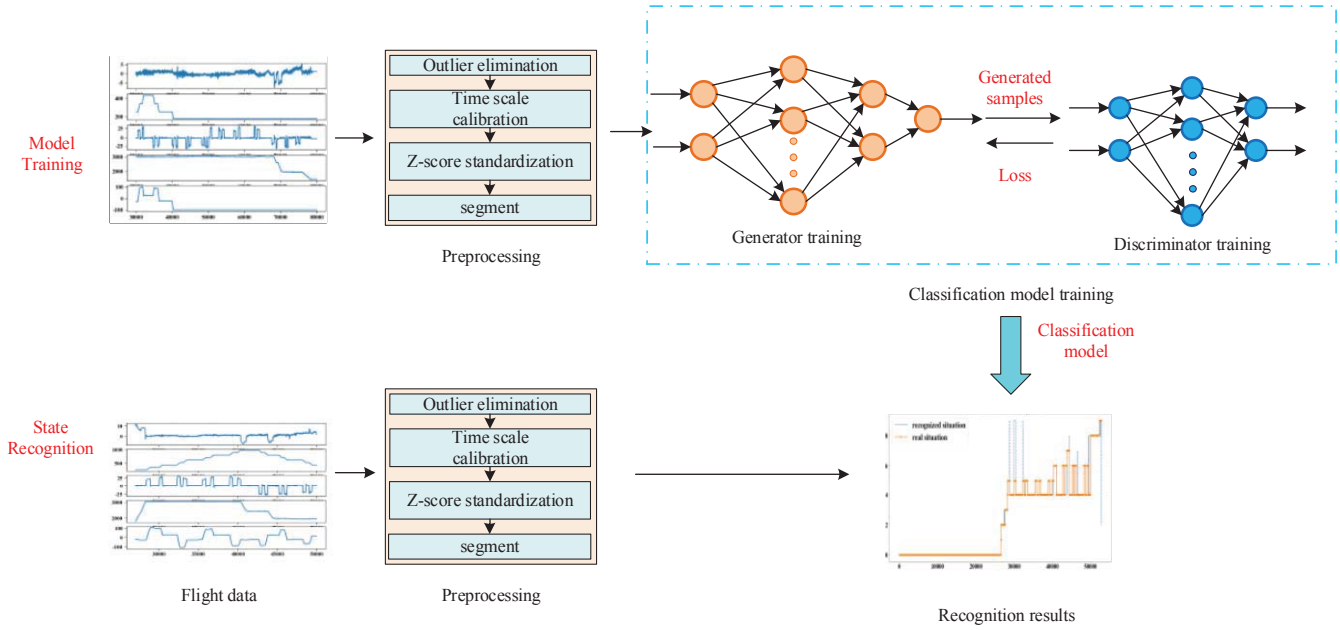

Figure 3. The framework of AC-GAN based method

The method is divided into two parts: model training and state recognition. The specific steps are as follow:

1) Model training process

a) Select flight parameters and preprocess the data

i. Delete repeat sampling. The data with a time interval less than the minimum sampling interval are considered resampling data

ii. Time scale calibration. The data with different sampling rates are unified to a standard sampling rate by interpolation or sampling.

iii. Z-score standardization. Mark X represents raw data, the mean value of data is μ, the standard deviation is σ, the formula of standardization is as follow:

$$(X - \mu)/\sigma \tag{6}$$

iv. Data segment. Divide the data into small segments of equal length and treat each segment as a sample of classification.

b) Generator and discriminator training

Mark the generated samples $\{x_{fake}^k\}_{k=1}^K$ as 0, whose corresponding category labels are $\boldsymbol{y}_{fake}^k$ and let the original samples $\{\boldsymbol{x}^m\}_{m=1}^M$ marked as 1, whose corresponding category labels are $\boldsymbol{y}^m$. The generated samples and the real samples jointly input to the discriminator for authenticity discrimination and classification, and then the discriminator could output the corresponding authenticity labels d_{real}^m, d_{fake}^k and category labels $\boldsymbol{c}_{real}^m$, $\boldsymbol{c}_{fake}^k$. GAN completes the training of the discriminator network by minimizing the error of two labels. The cross entropy loss function is calculated as follow:

$$L_c = -\frac{1}{M}\sum_{m=1}^{M}[\boldsymbol{y}^m \ln \boldsymbol{c}_{real}^m + (1-\boldsymbol{y}^m)\ln(1-\boldsymbol{c}_{real}^m)] - \frac{1}{K}\sum_{k=1}^{K}[\boldsymbol{y}_{fake}^k \ln \boldsymbol{c}_{fake}^k + (1-\boldsymbol{y}_{fake}^k)\ln(1-\boldsymbol{c}_{fake}^k)] \tag{7}$$

$$L_d = -\frac{1}{M}\sum_{m=1}^{M}\ln d_{real}^m - \frac{1}{K}\sum_{k=1}^{K}\ln(1-d_{fake}^k) \tag{8}$$

$$L_D = \arg\min_{\boldsymbol{\theta}}(L_c + L_d) \tag{9}$$

where L_c and L_d represent the error of the entropy loss of category labels and authenticity labels, separately. L_D is the loss function of the discriminator and $\boldsymbol{\Theta}$ is parameter set.

For training the generator, $\{\boldsymbol{x}_{fake}^k\}_{k=1}^K$ are marked as 1 and input to the discriminator. When the output authenticity labels are 0, it means the new samples generated by the generator failed to fool the discriminator, so the generator needs to adjust training through minimizing (10).

$$L_g = -\frac{1}{K}\sum_{k=1}^{K}\ln d_{fake}^k \tag{10}$$

$$L_G = \arg\min_{\boldsymbol{\theta}'}(L_c + L_g) \tag{11}$$

where L_g is the error of the entropy loss of authenticity labels and L_G is the loss function of the generator. $\boldsymbol{\Theta}'$ is the parameter set.

c) Adjust parameters by confrontation learning

When the original samples $\{\boldsymbol{x}^m\}_{m=1}^M$ are input to GAN, the target of the discriminator is to make the output value d_{real}^m close to 1 while when the generated samples $\{\boldsymbol{x}_{fake}^k\}_{k=1}^K$ are input to GAN, the target of the discriminator is to make the output value d_{fake}^k close to 0, that is, the discriminator can correctly judge that the generated samples are false. Meanwhile, the goal of the generator is to let d_{fake}^k close to 1, which means generated samples successfully fool the discriminator. Thus, the two form of confrontation. Through this confrontation learning mechanism, the discriminator and generator are alternately optimized: the generator is fixed first, and the discriminator is optimized by the random gradient descent method to maximize the discrimination accuracy of the discriminator. Then the discriminator is fixed, and the same method is also used to optimize the generator. During the training process, the two are continuously optimized to improve the classification and sample generation capabilities until the discriminator and generator reach the Nash equilibrium [20].

2) State recognition process

a) Select flight parameters and preprocess the data

b) State recognition. Input real samples and category labels into the classification model to obtain classified labels.

c) Calculate classification accuracy. Mark X_{true} is the number of correct recognition samples and X_{all} is the number of all samples. The accuracy is calculated as follow:

$$accu = X_{true} / X_{all} \tag{12}$$

IV. Experimental Description and Results Analysis

A. Experimental data description

Actual UAV flight data of our own fixed-wing drone for verifying drone PHM technology are utilized in the experiment, including standard pressure altitude, pitch angle, yaw angle, roll angle, yaw rate, vertical velocity, northward velocity, eastward velocity, Mach and airspeed. Since there is a jump between 360-degree and 0-degree in the yaw angle, it will trouble the subsequent calculation. Therefore, in the experiment, the amplitude of (0~360) is transformed to $(-\infty \sim +\infty)$ and the result is shown as follow:

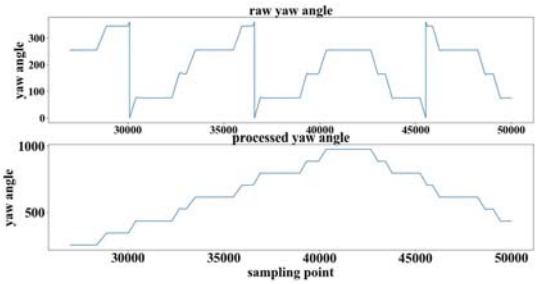

Figure 4. Yaw angle processing result

As shown in Fig.4, the 0~360 jump disappears, and the data become more continuous. Since the flight data are time series, if each point is regarded as a sample, not only the data volume is large, but also timing characteristics could not be reflected. Therefore, this paper segments the data using a sliding window,

and takes the data in one window as the same state. Given that the sampling interval is 0.08s while the duration of one state is at least 3.2s, the segment length is set 10, and the time interval in a window is 0.8s, which guarantees the data in one window belong to the same state.

B. Experimental results analysis

1) Experimental results using AC-GAN based method

Convolutional Neural Network (CNN) is utilized in both generator and discriminator network in the experiment due to its excellent feature learning capability. The learning rate is set 0.0002, the batch size is 128 and the iteration number is 50. The loss of generator and discriminator decreases gradually as the iteration number increases, and the result is shown in Fig.5.

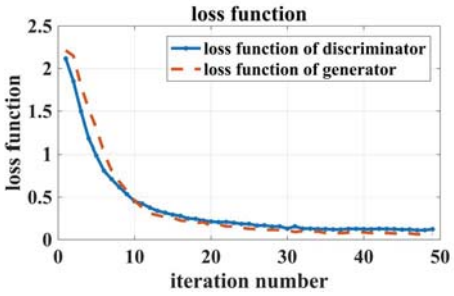

Figure 5. Loss function of generator and discriminator

The above figure shows the change of loss function in the iteration process. The blue line is the discriminator loss function and the orange dotted line is the generator loss function. The loss function of the generator and the discriminator decreases rapidly at the beginning, gradually stabilizes afterwards, and finally approaches 0, indicating that both the generator and the discriminator have reached a stable and optimal state, where the state recognition achieves the best result that is shown as follow:

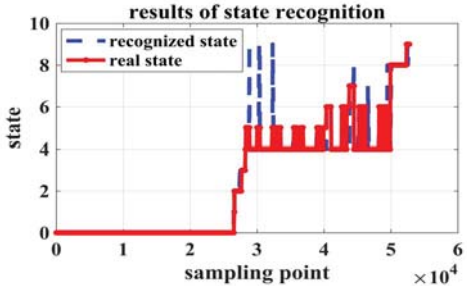

Figure 6. State recognition result using AC-GAN-based method

In the above figure, the dotted line represents the recognized state, and the solid line indicates the actual flight state. As can be seen from the figure, the recognition accuracy is high, reaching 94.33%.

2) Comparative experimental results

In order to verify the superiority of this method, comparative experiments are designed in this paper. In classification problems, SVM, decision trees, KNN and neural networks are common classifiers. Among these, SVM aims at the problem of bi-classification. To achieve multi-classification, it is necessary to establish multiple bi-classifiers, which is not suitable for complex and multi-state classification. The AC-GAN-based method is based on the neural network, so the neural network based method is not considered in the comparative experiment. Comparative experiments based on KNN and decision tree are designed. Similar to the AC-GAN based experiment, the segment method is also applied. Owing to lack of feature learning in deep learning, the slope and mean value of each segment data are adopted as features to perform subsequent classification. The same parameters as in the AC-GAN experiment are selected. The difference is that the mean value of each parameter and the slope of the pitch angle, altitude, airspeed, and yaw angle are calculated. The slope of the pitch angle is used to determine the take-off time. The slope of the altitude plays an important role in the identification of climbing, levelling and descent. The slope of the airspeed exerts an important influence on the determination of the velocity change. The slope of the yaw angle affects the judgment of the direction. The slope calculation results are shown in Fig.7

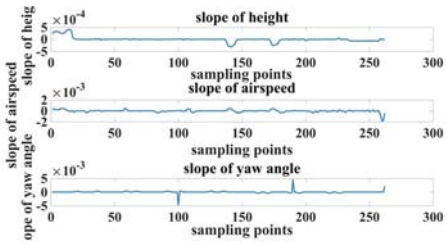

Figure 7. Slope calculation results

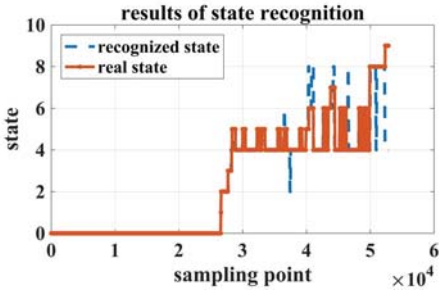

Figure 8. State recognition results using KNN method

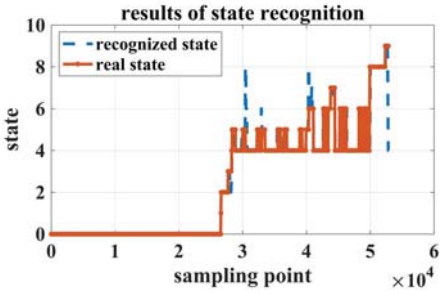

Figure 9. State recognition results using decision tree method

In Fig. 8 and Fig. 9, the state recognition results compared with the actual state have been shown in the two graphs, where the orange line represents the actual state label, and the dotted

blue curve is the recognized state label. As indicated in Fig. 8, KNN-based method makes good performance on state recognition, the accuracy of which reaches 89.21%. Some flight states, however, due to the short of training samples, the classifier could not effectively learn the potential features, resulting in poor recognition performance. For example, in the initial stage, the classifier fails to accurately divide accelerated climb and subsequent uniform climb, and also confuses the landing state with the levelling state. From Fig.9, the calculated accuracy based on decision tree based method is higher than that based on KNN, which hits 90.07%, but there is still no good distinction between accelerated climb and uniform climb. The accuracy of the state with fewer samples in three methods are shown in Table II.

TABLE II. RECOGNITION ACCURACY COMPARISON

Flight states	KNN	Decision tree	AC-GAN
Take off	0.0	0.50	0.50
Accelerated climb	1	0.37	0.78
Uniform climb	0	0.81	1
descent	0.58	0.88	0.94
approach	0.95	0.88	0.98
landing	0.02	0.68	0.61
overall	0.89	0.90	0.94

It can be seen from Table II that for six flight states with fewer samples, the recognition accuracy of AC-GAN based method is higher than that of KNN-based method in five states, and is higher than decision tree based method in four states, which demonstrates that the supplement of small samples alleviates the problem of imbalance samples and effectively improves the recognition accuracy.

V. CONCLUSION AND FUTURE WORK

In this work, considering the problem of imbalance classification samples due to the difference in flight time among various flight states, an AC-GAN based flight state recognition method is proposed. The method balances the training samples by generating fake samples that share the same distribution with the real samples, which improve the performance of flight state recognition, especially for those states with small samples. The actual flight data are utilized in the experiments. Except for the experiment of AC-GAN based method, this paper also implements comparative experiments of KNN-based and decision tree based methods, in order to identify the effectiveness of small samples supplement. Experimental results illustrate that the proposed method makes better performance on flight state recognition, the accuracy of which hits 94.33% while that of the other two methods just reaches 89.21% and 90.07%, respectively. Also, for those states with small samples, the AC-GAN based method shows higher recognition accuracy in most states. The proposed method not only improves the recognition accuracy, but also supports for subsequent fault diagnosis, fuel consumption assessment, etc.

Our future study will focus on establishing a baseline model of each state from a large number of historical data, which can be applied for flight quality assessment, anomaly detection, etc.

REFERENCES

[1] Z. Gong and Y. Guo, "Investigation of aero-engine fault diagnosis based on principal component analysis," Computer Measurement & Control, vol.20, no.8, pp. 2017-2023, 2012.

[2] B. Wang, D. Liu, W. Wang, and X. Peng, "A hybrid approach for UAV flight data estimation and prediction based on flight mode recognition," Microelectronics Reliability, vol. 84, pp. 253-262, 2018/05/01/ 2018.

[3] D. Liu, Y. Song, L. Li, H. Liao, and Y. Peng, "On-line life cycle health assessment for lithium-ion battery in electric vehicles," Journal of Cleaner Production, vol. 199, pp. 1050-1065, 2018.

[4] Y. Song, D. Liu, C. Yang, and Y. Peng, "Data-driven hybrid remaining useful life estimation approach for spacecraft lithium-ion battery," Microelectronics Reliability, vol. 75, pp. 142-153, 2017.

[5] L. Liu, Q. Guo, D. Liu, and Y. Peng, "Data-Driven Remaining Useful Life Prediction Considering Sensor Anomaly Detection and Data Recovery," IEEE Access, vol. 7, pp. 58336-58345, 2019.

[6] B. Wang, Y. Chen, D. Liu, and X. Peng, "An embedded intelligent system for on-line anomaly detection of unmanned aerial vehicle," Journal of Intelligent & Fuzzy Systems, vol. 34, pp. 3535-3545, 2018.

[7] N. C. Oza, K. Tumer, I. Y. Tumer, and E. M. Huff, "Classification of aircraft maneuvers for fault detection," in Multiple Classifier Systems, Proceeding. vol. 2709, T. Windeatt and F. Roli, Eds., ed Berlin: Springer-Verlag Berlin, 2003, pp. 375-384.

[8] D. Liu, L. Li, Y. Song, L. Wu, and Y. Peng, "Hybrid state of charge estimation for lithium-ion battery under dynamic operating conditions," International Journal of Electrical Power & Energy Systems, vol. 110, pp. 48-61, 2019.

[9] Y. Zhang and G. Fei, "Research on flight state recognition method based on track data," Aeronautical Computing Technique, vol. 47, no. 6, pp. 45-51, 2017.

[10] N. Shihong, S. Zhongke, and X. Chuan, "Establishment of Avion Inflight Maneuver Action Recognizing Knowledge Base," Computer Simulation, vol. 22, pp. 23-26, 2005.

[11] R. D. Maesschalck, D. Jouan-Rimbaud and D. L. Massart, "The Mahalanobis distance," Chemo metrics & Intelligent Laboratory Systems, vol. 50, no. 1, pp.1-18, 2000

[12] H. Li., Z. Shan, and H. Gou, "Flight action recognition algorithm based on MDTW," Computer Engineering and Applications, vol. 51, no. 9, 2015.

[13] W. Xu, "The method of recognizing carrier-based aircraft landing maneuver based on fuzzy neural network," Applied Science and Technology, vol. 40, no.2, pp. 26-29, 2013.

[14] J. Yang and X. Sheng, "Fuzzy support vector machines based recognition for aeroplane flight action," Journal of Projectiles, Rockets, Missiles and Guidance, vol.26, no.6, pp. 738-742, 2005.

[15] Y. Shen, N. Shi, and Z. Peng, "Flight action recognition method based on Bayesian network," Computer Engineering and Applications, vol.53, no.4, pp. 161-167, 2017.

[16] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," IEEE transactions on systems, man, and cybernetics, vol. 21, pp. 660-674, 1991.

[17] J. Han, J. Pei, and M. Kamber, Data mining: concepts and techniques: Elsevier, 2011.

[18] L. Tran, X. Yin, and X. Liu, "Disentangled representation learning gan for pose-invariant face recognition," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1415-1424.

[19] X. Liang, Z. Hu, H. Zhang, C. Gan, and E. P. Xing, "Recurrent topic-transition gan for visual paragraph generation," in Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 3362-3371.

[20] L. J. Ratliff, S. A. Burden, and S. S. Sastry, "Characterization and computation of local nash equilibria in continuous games," in 2013 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton), 2013, pp. 917-924.