An Automatic Modulation Classification Network for IoT Terminal Spectrum Monitoring under Zero-sample Situations

Quan Zhou
Beijing University of Posts and Telecommunications
https://orcid.org/0000-0003-1547-3065

Ronghui Zhang
Beijing University of Posts and Telecommunications

Fangpei Zhang
Information Science Research Institute of China Electronic Technology Group Corporation

Xiaojun Jing (✉ jxiaojun@bupt.edu.cn)
Information Science Research Institute of China Electronic Technology Group Corporation

Research

Keywords: Internet of Things, automatic modulation classification, zero-shot learning, generative adversarial network

Posted Date: October 5th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-927160/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License
An Automatic Modulation Classification Network for IoT Terminal Spectrum Monitoring under Zero-sample Situations

Quan Zhou¹, Ronghui Zhang¹, Fangpei Zhang² and Xiaojun Jing¹*  

Abstract  
Rely on powerful computing resources, a large number of internet of things (IoT) sensors are placed in various locations to sense the environment around where we live and improve the service. The proliferation of IoT end devices has led to the misuse of spectrum resources, making spectrum regulation an important task. Automatic modulation classification (AMC) is a task in spectrum monitoring, which senses the electromagnetic space and is carried out under non-cooperative communication. However, DL-based methods are data-driven and require large amounts of training data. In fact, under some non-cooperative communication scenarios, it is challenging to collect the wireless signal data directly. How can the DL-based algorithm complete the inference task under zero-sample conditions? In this paper, a signal zero-shot learning network (SigZSLNet) is proposed for AMC under the zero-sample situations firstly. Specifically, for the complexity of the original signal data, SigZSLNet generates the convolutional layer output feature vector instead of directly generating the original data of the signal. The semantic descriptions and the corresponding semantic vectors are designed to generate the feature vectors of the modulated signals. The generated feature vectors act as the training data of zero-sample classes, and the recognition accuracy of AMC is greatly improved in zero-sample cases as a consequence. The experimental results demonstrate the effectiveness of the proposed SigZSLNet. Simultaneously, we show the generated feature vectors and the intermediate layer output of the model.

Keywords: Internet of Things; automatic modulation classification; zero-shot learning; generative adversarial network

1 Introduction
The rapid development of the Internet of Things (IoT) and mobile networks is to meet users' demands for performance, speed, seamless connectivity, Intelligence, and portability [1]. The 6th Generation Mobile Communication Technology (6G) is the next generation of telecommunication standards to ensure the needs of ever-growing intelligence communication applications. The IoT is a potential candidate for leveraging the resources of communication networks. It is estimated that there are currently over 50 billion devices connected to the Internet. Based on these IoT devices to sense the surrounding environment and provide better services. Spectrum is the most valuable resource in wireless networks, and the increase of IoT end devices may lead to spectrum resource abuse. Therefore, regulating the spectrum becomes an important task, and it is a non-cooperative communication method. Consequently,
it is imperative to classify receiver modulation types under non-cooperative communication conditions. Some researchers have proposed an automatic classification modulation scheme called automatic modulation classification (AMC), which contributes to signal recognition, threat assessment, and spectrum monitoring [2].

For the urgent needs of spectrum regulation in IoT, AMC has attracted much attention in recent years. The conventional modulation recognition algorithms can be divided into likelihood-based (LB) methods, and feature-based (FB) methods [3]. LB determined the modulation mode of the received signal by the likelihood comparison, which infers the labels by the Bayesian optimization. FB regarded AMC as a pattern recognition issue, and it yielded suboptimal classification accuracy although it was less sensitive to the uncertainties. The algorithm extracts the statistical features [4, 5, 6], instantaneous time features [7] and wavelet features [8] from the original modulated signal, and these features are used as input to the machine learning algorithm, such as Support Vector Machine (SVM), Lightgbost, Decision Tree (DTree), K-Nearest Neighbors (KNN), etc.[9, 10, 11, 12], and infer the class to which they belong as a consequence. However, LB methods are usually of considerable computational complexity, prior knowledge-based accompanying sensitivity to channel noise, and poor robustness. FB methods are expert feature-based and suboptimum classification accuracy. Accordingly, both LB and FB are poorly adapted to non-cooperative communication environments.

Recently, Convolutional Neural Networks (CNNs) have been successfully applied to AMC by learning valid representations from complex data [13]. [14] leveraged CNN network on AMC, and high classification accuracy was obtained, which proved the CNN-based methods are better than traditional feature-based methods. [15] employed CNN, residual network (ResNet), Inception network, and Convolutional Long Short Term Memory Deep Neural Network (CLDNN) to process modulated signals, respectively, which demonstrated that the temporal feature was effective in improving the classification accuracy. [16] considered feature interactions and combined the advantages of CNN and Long Short Term Memory (LSTM) to solve the AMC issue. Researchers propose a CNN-LSTM dual-stream structure to explore feature interactions and the spatial-temporal properties of the original signal. Simultaneously, the signal constellation was applied to AMC for the classification accuracy in [17]. Progressively, for better robustness and decent inference, a grid constellation matrix-based algorithm was considered in [18]. For the few-sample conditions, a host of researchers have made some explorations. Bu et al. [19] proposed an adversarial transfer learning architecture (ATLA) to reduce the differential on sampling rates, with 10% of the dataset for training, and achieved a favorable classification result. Li et al. [20] exploited a capsule network (CapsNet) to extract features from signals, with only 3% of the training data set, and obtained a promising inference result. Nevertheless, the schemes mentioned above work on the assumption that all categories of training sets have data, whether sufficient or few. In fact, these methods can not work under class-imbalance situations, in which the training set misses data for some classes.

The data augmentation method that based on Generative Adversarial Networks (GAN) [21] is a possible way to address the issue of missing data for some classes, where the generator generates a specified data to fool the discriminator, and then
the discriminator attempts to distinguish the real data from the generated data accurately. In the literature [22], the unstable convergence of GAN was solved, and an efficient approximation of the Wasserstein distance was provided for WGAN. For WGAN-GP [23], the gradient disappearance and the weight clipping were considered by an improved gradient penalty. Note that GAN-based data augmentation methods still require the corresponding data for the training and the augmentation. Under the zero-sample conditions, GAN-based data augmentation methods are not applicable any more for the data imbalance issue. For the classes with entirely missing data, the zero-sample learning scheme is always considered by expert-set linguistic descriptions, transfer learning, or matrix transformations [24, 25]. However, the linear matrix transformation methods have a particularly low inferential accuracy under complex data scenes. Furthermore, the modulated signal do not have a semantic expert dataset. As a result, the methods based on the semantic space mapping do not have the modulated semantics.

In this paper, in order to better regulate the spectrum usage of IoT devices, a novel signal zero-shot learning network (SigZSLNet) is proposed to solve the issue of zero-sample mentioned above for AMC firstly, where the mapping relations between the classes are established by the semantic space mapping, and the expert-set linguistic descriptions are conducted for the modulated modes. GAN drives the generation of unseen classes, which enriches the training set. In summary, the contributions of this article can be concluded as

1) The semantic descriptions are designed for different modulated signals by their properties, and the semantic vectors are obtained based on the generation module of the semantic attribute vector.

2) The WGAN-GP based method is employed to generate the feature vectors of the modulated signals under the guidance of the semantic vectors. After that, a complete training dataset is constructed with the WGAN-GP module, including the real feature vector and the synthetic feature vector.

3) The complete training dataset is input to the classifier, and the estimation result of the AMC is obtained. Experimental results indicate that the proposed SigZSLNet can solve the zero-sample issue for AMC effectively.

To better present this paper, the remainder of this paper is organized as follows. Section II introduces the motivation and the description of the model. Section III presents the experiments of various groups, and the result is analyzed in detail. We discuss the result in section IV. Finally, Section V presents the conclusion of this article.

2 Method
2.1 Motivation
With the rapid development of 5G technology, more and more sensors and devices are connected to the internet, and the internet is transmitted through the wire and air network, satellite network, etc. The application areas are becoming more and more extensive. As show in Figure 1, the scenarios of wireless spectrum management, non-cooperative communication are considered in this paper, where the receiver does not have a prior knowledge of the modulated signal at the transmitter. As a result, the classes of the modulated signal at the receiver may be different from
the available data. The training data collected in advance are called seen classes, and training data that are not available are called unseen classes. In this way, the available classes of the modulated signal at the receiver is $C_{\text{seen}} = \{c_1, c_2, \ldots, c_{m-k}\}$, and the unavailable classes $C_{\text{unseen}} = \{c_{m-k+1}, c_{m-k+2}, \ldots, c_m\}$. The wireless signal at receiver can be represented as

$$ R(t) = s(t) * G(t) + n(t) $$

where $s(t)$ is the modulated signal from the transmitter, $G(t)$ is the channel, and $n(t)$ is the Additive White Gaussian Noise (AWGN).

Generally, the common DL-based methods [13] can infer the received data with an adequate training data set. To enable the DL-based method work under zero-sample conditions, this paper mainly concerns how to generate the missing data $C_{\text{unseen}}$ accurately with WGAN-GP and how to conduct AMC on the basis of the real data and the generated data.

Algorithm 1 Semantic Vector Generation

Input: $C_{\text{seen}}, D_{\text{seen}}, D_{\text{unseen}}$

Output: $S_{\text{seen}}, S_{\text{unseen}}$

for $i = 0$ to $n_1$ do
    $y_1 = f_{\text{cov}}(C_{\text{seen}})$;
    $y_2 = f_{\text{enc}}(D_{\text{seen}})$;
    Optimizing $Loss = L_A(y_1, y_2)$
end for

$y_{\text{seen}} = f_{\text{enc}}(D_{\text{unseen}})$;
$y_{\text{unseen}} = f_{\text{enc}}(D_{\text{unseen}})$;
$S_{\text{seen}} = y_{\text{seen}}$
$S_{\text{unseen}} = y_{\text{unseen}}$
end loop

Algorithm 2 Algorithm for Unseen Classes Generation

Input: $C_{\text{seen}}, S_{\text{seen}}, S_{\text{unseen}}$

Output: $y_{\text{unseen}}$

1: for $i = 0$ to $n_2$ do
2: $y_G = f_G(S_{\text{seen}}, G_{\text{noise}})$;
3: $y_{\text{seen}} = f_{\text{CNN}}(C_{\text{seen}})$;
4: $\{0, 1\} = f_D(y_{\text{seen}}, y_G)$;
5: end for
6: loop
7: $y_{\text{unseen}} = f_G(D_{\text{unseen}})$;
8: $C_{\text{unseen}} = y_{\text{unseen}}$
9: end loop

Algorithm 3 Algorithm for Classifier Training

Input: $\{C_{\text{seen}}, C_{\text{unseen}}\}, \{l_{\text{seen}}, l_{\text{unseen}}\}$

Output: $\theta_{\text{classifier}}$

1: for $i = 0$ to $n_3$ do
2: $y_{\text{label}} = f_{\text{classifier}}(C_{\text{seen}}, C_{\text{unseen}} | \theta_{\text{classifier}})$;
3: $loss = L_G(y_{\text{label}}, \{C_{\text{seen}}, C_{\text{unseen}}\})$;
4: Optimizing $loss$;
5: end for

2.2 Model Description

The generation module of the semantic attribute vector is shown in Figure 2. The network architecture of the proposed SigZSLNet is exhibited in Figure 3.
Figure 3, SigZSLNet contains three modules, the CNN module, the WGAN-GP module, and the Classifier module.

Table 1: Layers of semantic generation module and output dimensions of each layer

| Modules                  | Layers                  | Output         |
|--------------------------|-------------------------|----------------|
| **Convolutional Part**   | Input                   | 2×1024×1       |
|                          | Conv1D(filters 64, size 1×5) | 64×1016×1      |
|                          | Maxpooling(size 1×5)     | 64×204×1       |
|                          | Conv1D(filters 64, size 1×5) | 64×200×1       |
|                          | Maxpooling(size 1×5)     | 64×40×1        |
|                          | Conv1D(filters 64, size 1×5) | 64×36×1        |
|                          | Maxpooling(size 1×5)     | 64×8×1         |
|                          | Conv1D(filters 64, size 1×5) | 64×4×1         |
|                          | Maxpooling(size 1×5)     | 64×1×1         |
|                          | Flatten                 | 64×1           |
|                          | Fully Connected         | 64×1           |
|                          | Fully Connected         | 64×1           |
| **Encoding Part**        | Encoding                | 32×47×1        |
|                          | Conv1D(filters 128, size 1×5) | 128×43×1       |
|                          | Maxpooling(size 1×5)     | 128×9×1        |
|                          | ReLU                    | /              |
|                          | Conv1D(filters 128, size 1×5) | 128×5×1        |
|                          | Maxpooling(size 1×5)     | 128×1×1        |
|                          | ReLU                    | /              |
|                          | Flatten                 | 128×1          |
|                          | Fully Connected         | 64×1           |

2.2.1 Semantic Generation

As it is shown in Figure 2, the generation module of the semantic attribute vector contains the convolutional part and the encoding part, where the convolutional part is to extract the signal data features, and the convolutional encoding part is to encode semantic descriptions and extract semantic features. The layer of semantic generation module and output dimensions of each layer are represented in Table 1. More specifically, various modulated signals are related to their property descriptions. As a result, the property descriptions of different modulated signals are quantified with one-hot encoding in this letter, and the quantification results form the specific semantic vectors, which can guide the WGAN-GP to generate specific data. After the loss function optimization, a semantic feature vector is obtained that can be used as the input to the generator to propel the generation of specific data.

The measured relations between the semantic description and the signal are optimized by Joint Embedding Loss [26], which is formulated as

\[
L_A = \frac{1}{N} \sum_{n=1}^{N} \max(0, \Delta(y_n, y)) + \mathbb{E}[F(s, t) - F(s, t_n)] + \max(0, \Delta(y_n, y) + \mathbb{E}[F(s, t) - F(s, t_n)])
\]  

(2)
Table 2: Layers of SigZSLNet and the output dimensions of each layer. G is the generator module, and D is the discriminator module.

| Module | Layer | Output |
|--------|-------|--------|
| CNN    | Input Layer | 1024×2 |
|        | Conv1D(filters 64, size 1×8) | 1024×64 |
|        | Maxpooling(size 1×2) | 512×64 |
|        | Conv1D(filters 64, size 1×8) | 512×64 |
|        | Maxpooling(size 1×2) | 256×64 |
|        | Conv1D(filters 64, size 1×8) | 256×64 |
|        | Maxpooling(size 1×2) | 128×64 |
|        | Conv1D(filters 64, size 1×8) | 128×64 |
|        | Maxpooling(size 1×2) | 64×64 |
|        | Conv1D(filters 64, size 1×8) | 64×64 |
|        | Maxpooling(size 1×2) | 32×64 |
|        | Flatten | 2048 |
|        | Fully Connected Layer | 128 |
| G      | Input Layer | 64 |
|        | Fully Connected Layer | 256 |
|        | Leaky.relu | / |
|        | Fully Connected Layer | 128 |
| D      | Input Layer | 64 |
|        | Fully Connected Layer | 256 |
|        | Leaky.relu | / |
|        | Fully Connected Layer | 128 |
| Classifier | Input Layer | 128 |
|          | Fully Connected Layer | 128 |
|          | PReLU | / |
|          | SoftMax | 7 |

\[
\Delta(a, b) = \begin{cases} 
1, & \text{if } a \neq b \\
0, & \text{if } a = b 
\end{cases}
\]  

(3)

where \( N \) denotes the total number of the signal and text pairs; \( \Delta \) signifies the 0-1 loss; \( y \) denotes the class label; \( t \) is the subset of text descriptions; \( s \) signifies the subset of signals; \( \mathbb{E}[\cdot] \) denotes the mean operation; \( F(s, t) = e(s)^T \varphi(t) \), where \( e(s) \) is the encoder of the input signals, and \( \varphi(t) \) is the encoder of the input attribute description. The algorithm for semantic generation module is shown in Algorithm 1, where \( C_{seen} \) is the signal of seen classes, \( D_{seen} \) is the semantic description of \( C_{seen} \), \( D_{unseen} \) represents the semantic description of unseen classes. \( S_{seen} \) and \( S_{unseen} \) are the output semantic vector of seen classes and unseen classes.

2.2.2 CNN Module

The CNN module is leveraged to obtain the feature vectors, which consists of five convolutional layers, one fully connected layer. In detail, the convolution layer includes a convolution with the stride of \( 1 \times 1 \) and the convolution kernel of \( 1 \times 8 \), an activation function \( f_{relu} = \max(0, x) \), and a maximum pooling with the stride
of $1 \times 2$ and the kernel of $1 \times 2$. The output of the last fully connected layer is 128 dimensions. The details are depicted in Table 2. In general, the main role of CNN module is to extract spatial features of modulated signal data. Similarly, for CNN’s ability to extract spatial features, the open source pre-trained CNN model with convolutional module can be used as the CNN module in the proposed SigZSLNet. The pre-training weights of the model proposed in [14] can be found at GitHub. In this way, we utilize this model as a CNN module of SigZSLNet, using the output of the first fully connected layer as the feature vector of the signal.

### 2.2.3 GAN Module

The main motivation of the GAN module is to generate the feature vectors of the unseen (missing) classes, which contains the generator (G) part and the discriminator (D) part. The generator consists of two fully connected layers with 256 neurons in the middle layer and 128 outputs, and the activation function in each layer is *leaky relu*. Similarly, the discriminator consists of two fully connected layers, with 256 neurons, and the activation function in the first layer is *leaky relu*, while the second layer has no activation function. Table 2 shows the detailed structure of WGAN-GP.

The input to the generator is a semantic attribute vector of the modulated signal, which is generated in semantic generation module. The input of the discriminator is a 128-dimensional attribute vector generated by the generator. The generator and the discriminator are playing a game called Min-Max, where the generator tries to generate data from the semantic attribute vector that can be judged as true by the discriminator. Simultaneously, the discriminator tries to distinguish between the real feature data from the CNN module and the synthetic data generated by the generator. After several epochs, the generator generates modulated signal data that makes the discriminator difficult to distinguish. In this way, the generated feature data can substitute for the real data. This process can be formulated as follows

$$
\min_G \max_D L_{GAN} = -E_{x \sim p_r}[D(x)] + E_{x \sim p_g}[D(x)] \\
+ \lambda E_{x \sim p_g}[\| \nabla_x D(x) \|_2 - 1]^2
$$

(4)

where $x \sim p_r$ denotes the real data and $x \sim p_g$ signifies the generated data from the generator. $\| \nabla_x D(x) \|_2 - 1]^2$ is the gradient penalty while $\lambda$ serves as the penalty coefficient with the default $\lambda = 10$. **Algorithm 2** describes the generation process of the missing signal classes, where $f_G(\cdot)$ denotes the generator, $f_D(\cdot)$ denotes the discriminator, $f_{CNN}(\cdot)$ denotes the CNN part, $y_{unseen}$ is the generated feature vector of unseen classes.

### 2.2.4 Classifier Module

The classifier module contains a fully connected layer and a *Softmax* layer. Specifically, the fully connected layer has 128 neurons. *PReLU* is used as the activation functions in the fully connected layer, and *Softmax* is the judicial function of the last layer, which are respectively formulated as

$$
f_{PReLU}(x_i) = \begin{cases} 
  x_i, & \text{if } x_i > 0 \\
  \lambda x_i, & \text{if } x_i \leq 0
\end{cases}
$$

(5)
\[ f_{\text{Softmax}}(x_i) = \frac{e^{x_i}}{\sum_i e^{x_i}} \]  \hspace{1cm} (6)

where \( \lambda \in (0,1) \) is a variable that can be learned by a backpropagation algorithm, and adjust to the most appropriate slope value.

The training stage focuses on training the classifier to have the ability to classify seen and unseen classes. First, the semantic attribute vector is generated based on the semantic description. In this way, the semantic vectors of the seen and unseen categories are obtained. Then, WGAN-GP is trained by the seen classes and their semantic vectors. Consequently, synthetic feature vectors for the unseen categories are generated from the generator based on their semantic attribute vectors. Finally, both the real feature vectors and the synthetic feature vectors are input to the Classifier module. After processing by the full connection layer and the Softmax layer, the prediction result of AMC is obtained. The final loss function is formulated as

\[ L_C = h(y, f(x_{\text{seen}}, x_{\text{unseen}} | \theta)) \]  \hspace{1cm} (7)

where \( h \) is the cross entropy, which is defined as \( h(p, q) = -\sum_i p_i \log(q_i) \). \( y \) denotes the true label and \( f \) signifies the classifier; \( \theta \) is the weight of the fully connected layer, and \( x_{\text{seen}} \) is the CNN feature of the seen classes; \( x_{\text{unseen}} \) denotes the unseen class.

The test stage is concerned with determining of the category to which the received modulated signal data belongs. Generally, In an end-to-end AMC system, the modulated signals received by the receiver may belong to the seen class or the unseen class of modulated signals. The feature vector is obtained after feature extraction by the CNN module, and then inferred by the classifier module to obtain the inference results.

3 Experimental

3.1 Dataset and Settings

The experimental dataset of the in-phase and quadrature (IQ) samples is obtained based on the MATLAB 2019b platform. The dataset consists of seven modulated modes, \( \text{Group} = \{ \text{BPSK, QPSK, 2ASK, 4ASK, 16QAM, 32QAM, 64QAM} \} \) (SNRs range from 0 dB to 10 dB, with the stride 1 dB). For each class, there are 400 modulated signals for each SNR, where 300 of them are for training and the rest are for the test. To sum up, the training set contains 23,100 modulated signals, while the test set contains 7,700 modulated signals. The semantic description of each modulated mode is shown in Table 3, where the “statistical peaks” is the peak number of the modulated signal.

Four groups of the seen classes are considered in the experiments, which are presented in Table 4. For Group 1, we do not have the class of BPSK for training. Consequently, we utilize the data of seen classes \{QPSK, 2ASK, 4ASK, 16QAM, 32QAM, 64QAM\} and the description semantic of all classes to generate unseen classes. Group 2 is similar. Group 3 and Group 4 are tested under two unseen classes conditions. Similarly, for Group 3, we do not have \{QPSK, 4ASK\}, and leverage \{BPSK, 2ASK, 16QAM, 32QAM, 64QAM\} to obtain unseen classes. For Group 4, we do not have \{QPSK, 4ASK\},
Table 3: The semantic description of each category

| Classes | statistical peaks | phase modulation | amplitude modulation | order |
|---------|-------------------|------------------|----------------------|-------|
| BPSK    | 2                 | yes              | no                   | 2     |
| QPSK    | 2                 | yes              | no                   | 2     |
| 2ASK    | 4                 | no               | yes                  | 2     |
| 4ASK    | 4                 | no               | yes                  | 4     |
| 16QAM   | 4                 | yes              | yes                  | 16    |
| 32QAM   | 6                 | yes              | yes                  | 32    |
| 64QAM   | 8                 | yes              | yes                  | 64    |

Table 4: The semantic description of each category

| Groups | Seen Classes | Unseen Classes |
|--------|--------------|----------------|
| Group 1| QPSK, 2ASK, 4ASK, 16QAM, 32QAM, 64QAM | BPSK |
| Group 2| BPSK, QPSK, 4ASK, 16QAM, 32QAM, 64QAM | 2ASK |
| Group 3| BPSK, 2ASK, 16QAM, 32QAM, 64QAM | QPSK, 4ASK |
| Group 4| BPSK, QPSK, 2ASK, 16QAM, 32QAM | 4ASK, 64QAM |

The attribute feature vector is normalized in this letter because it can accelerate the training and prevent overfitting. Simultaneously, the Gaussian Noise is added to the input of the generator because the Gaussian data is easier to map into the CNN feature distribution. Additionally, the harmonic mean is considered to reflect the inference ability, and it is defined as

\[ H = \frac{2su}{s + u} \]  

(8)

where \( s \) denotes the average classification accuracy for the seen classes, and \( u \) is for the unseen classes. The implementation of the SigZSLNet is based on Tensorflow.

3.2 Results

Table 5: The performance comparison among various methods

| Methods | Groups | Seen | Unseen | \( H \) | Accuracy |
|---------|--------|------|--------|---------|----------|
| 1       |        | 0.8924 | 1 | 0.9431 | 0.9078 |
| SigZSLNet | 2   | 0.9364 | 0.8515 | 0.8919 | 0.9243 |
| 3       |        | 0.88 | 0.7114 | 0.7868 | 0.8318 |
| 4       |        | 0.7991 | 0.6932 | 0.7424 | 0.7688 |

Table 5 provides the performance comparisons of SigZSLNet among various groups. VGGNet, ResNet, and CLDNN are pre-trained on RML2018a [14]. In addition, there is one missing class in Group 1 and Group 2, while Group 3 and Group 4 have two classes of data missing. The recognition accuracy in Table 5 is the average value of five experiments. Firstly, VGGNet, ResNet, and CLDNN separately serve as the CNN module, and the test data is input to the three pre-trained models. The corresponding recognition accuracy is extremely low because the dataset in
this letter is not the same distribution as RML2018a. It states that the pre-trained model does not affect the classification results, and the model pre-trained based on the RML2018a dataset can be employed as the CNN part of the model. Consequently, the pre-trained ResNet serves as the CNN module of the SigZSLNet. Secondly, the average classification accuracy of the proposed SigZSLNet exceeds 76%, even above 85% for Group 1 and Group 2, which indicates that SigZSLNet can effectively conduct AMC under zero-sample situations. Thirdly, compared with the recognition accuracy of Group 1 and Group 2, the performance of Group 3 and Group 4 significantly declines. The reason for this situation is that the data generation difficulty of the GAN module increases with the rise of the missing classes. The modulated signal is highly susceptible to the signal-to-noise ratio. Under low SNR conditions, the signal is strongly disturbed, the features are not obvious, and the features learned by the generator are not obvious. As a result, the recognition accuracy decreases due to the fall of the data generation quality. In total, Table 5 states that the proposed SigZSLNet is an AMC scheme more applicable to real scenarios.

The test accuracy comparisons under various SNRs are shown in Figure 4, where the settings of Group 1-4 are the same as that of Table 5. From Figure 4, the recognition accuracy improves with the rise of SNR firstly and then slightly declines with the rise of SNR when the SNR is above 6 dB. It indicates that SNR is an important influence factor for the recognition accuracy of AMC. Particularly, the average classification accuracy exceeds 85% of various groups when SNR varies from 6 dB to 9 dB because GAN balances the quality of the generated feature vector. In addition, as discussed in Table 5, the recognition accuracy of the proposed SigZSLNet gradually deteriorates with the rise of the unseen classes.

In Figure 5, 50 synthetic feature vectors and 50 real feature vectors of the random 2ASK modulation signal are visualized. From Figure 5, although there is much difference between the synthetic feature vector and the real feature vector, their similarity is at a high level, especially for the waveform variation trend. The generated feature vectors have great similarity with the real feature vectors. This means that SigZSLNet can generate the feature vectors of zero-sample classes accurately.

As a supplement, the confusion matrix comparisons of various groups are made under SNR = 6 dB in Figure 6, where the BPSK signal is missing in Group 1 and exists in Group 2-4. Figure 6 indicates that the classification accuracy of Group 1 is obviously higher than that of Group 2-4 for the BPSK signal, which manifests the generation quality of the synthetic feature vectors by the GAN module outperforms that of the real feature vectors by the CNN module. This states that the proposed SigZSLNet can effectively fill the missing classes and improve the accuracy of AMC under zero-sample situations.

In Figure 7, we show the visualized results of the feature vectors before input to the classifier module. The 128-dimensional features are downscaled into two-dimensional coordinate vectors by the t-SNE algorithm [27]. Figure 7 (a)-(d) are the visualization of the seen and unseen classes of Group 4 with different signal-to-noise ratios for the feature vectors. The classification accuracy of the classifier is highest at the SNR of about 6 dB, as described in the previous experimental results. For the results of dimensionality reduction visualization, at a signal-to-noise ratio of 6dB,
different classes are clustered together, and each class is easier to distinguish. When \( SNRs = \{0 \text{ dB}, 4 \text{ dB}, 10 \text{ dB}\} \), the reduced-dimensional features of BPSK and QPSK are mixed together and not easily distinguished. However, for 4ASK and 64QAM, which are unseen categories, the generated data are distinguishable. It is concluded that the algorithm has the highest classification accuracy at a signal-to-noise ratio of 6 dB because the CNN feature vectors of BPSK and QPSK are easily confused. Thus, the result of difficulty distinguishing between BPSK and QPSK is that the pre-trained CNN feature vector outputs low-quality feature vectors.

### 4 Discussions

Simulation results show that the SigZSLNet proposed in this paper can generate data of missing modulation signal to make up for that data in the training set and solve zero-sample in AMC. In the groups where one category was missing and two categories were missing, the accuracy rate exceeded 76% in the task of classifying seven categories. However, it is known from the simulation results that the generated modulated signals lack diversity and can only generate data of modulated signals with a single SNR. The model needs to be further improved in terms of SNR diversity to generate rich modulated signal SNRs. In addition, the poor robustness of the open-source CNN model, which is only trained with 24 categories of modulated signals, makes it difficult to excel in our experiments. In future work, we will collect rich modulated signal data, making a dataset like Imagenet [28] and CoCo [29], to train more robust pre-trained models.

### 5 Conclusion

The increasing number of IoT devices means that more traffic will occupy scarce available spectrum in the future, so it becomes extremely important to regulate and recognize the observed signals. In this paper, for the urgent need of spectrum regulation in IoT terminals, we first propose the method SigZSLNet to implement AMC under zero-sample conditions. Based on the semantic feature vector, the feature vectors of the missing modulated signals are generated with GAN, and the accuracy of AMC is greatly improved due to the generated feature vectors. The various groups experimental results validate the effectiveness of the proposed SigZSLNet. The proposed method obtains an average accuracy of over 85% in the missing category when one modulation signal is missing, corresponding to an accuracy of over 76% for the classification over seen and unseen classes. While two modulation signals are missing, the proposed algorithm obtains an average accuracy of more than 69% in the missing category, corresponding to an accuracy of more than 76% for the seven classification tasks. We visualized the generated feature vectors for comparison, and also visualized the data clustering for each category based on PCA algorithm and t-SNE algorithm to prove the validity of the generated data. In conclusion, the experimental results show that our proposed method effectively solves the AMC task of spectrum resource management for IoT terminal devices.

### Abbreviations

AMC: Automatic modulation classification; IoT: Internet of things; SigZSLNet: Signal zero-shot learning network; GAN: Generative adversarial network; 6G: The 6th
Declarations
Authors’ contributions
Q. Z was in charge of the major theoretical analysis, algorithm design, experimental simulation, and paper writing. XJ. J and Q. Z had contributions to paper writing. FP. Z and RH. Z had contributions to theoretical analysis and gave suggestions of the organization. All authors read and approved the final manuscript.

Funding
This research was solely the work of the authors, funded by no authority.

Availability of data and materials
Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

Competing interests
The authors declare no conflict of interest.

Author details
1 School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing, China. 2 Information Science Research Institute of China Electronics Technology Group Corporation, Beijing, China.

References
1. Yuanhao, C., Fan, L., Xiaojun, J., Junsheng, M.: Integrating sensing and communications for ubiquitous iot: Applications, trends and challenges. arXiv preprint arXiv:2104.11457 (2021)
2. Junsheng, M., Xiaojun, J., Hai, H., Ning, G.: Subspace-based method for spectrum sensing with multiple users over fading channel. IEEE Communications Letters 22(4), 848–851 (2018)
3. Octavia, A., Dobre, Menguc, O., Srierraman, R., Robert J, I.: Cyclostationarity-based robust algorithms for qam signal identification. IEEE Communications Letters 16(1), 12–15 (2012)
4. Wei, S.: Feature space analysis of modulation classification using very high order statistics. IEEE Communications Letters 17(9), 1688–1691 (2013)
5. David, B., Balu, S.: A hybrid ica-svm approach to continuous phase modulation recognition. IEEE Signal Processing Letters 16(5), 402–405 (2009)
6. Sai, H., Yuanyuan, Y., Zhiqing, W., Zhiyong, F., Zhang, P.: Low complexity automatic modulation classification based on order-statistics. In: IEEE Transactions Wireless Communications, vol. 16, pp. 400–411 (2017). IEEE
7. E.E., A., A.K., N.: Automatic identification of digital modulation types. Signal Process 47(1), 55–69 (1995)
8. Stepanne, M., Sifen, Z.: Characterization of signals from multiscale edges. IEEE Transactions on Pattern Analysis and Machine Intelligence 14(7), 710–732 (1992)
9. Corinna, C., Vladimir, V.: Support-vector network. Machine Learning 20(3), 273–297 (1995)
10. Leo, B.: Random forests. IEEE transactions on broadcasting 45(1), 5–32 (2001)
11. T.M., C., P.E., H.: Nearest neighbor pattern classification. IEEE transactions on broadcasting 13(1), 21–27 (1967)
12. N, F., Geiger, D., Moises, G.: Bayesian network classifiers. Machine Learning 29(2), 131–163 (1997)
13. Yann, L., Yoshua, B., Geoffrey, H.: Deep learning. Nature 521(7553), 436–444 (2015)
14. Tim, O., Tamoghna, R., T. Charles, C.: Over-the-air deep learning based radio signal classification. IEEE Journal of Selected Topics in Signal Processing 12(1), 168–179 (2018)
15. Nathan, W., Timothy, O.: Deep architectures for modulation recognition. IEEE International Symposium on Dynamic Spectrum Access Networks, 1–6 (2017)
16. Zufan, Z., Hao, L., Chun, W., Chengquan, G., Yong, X.: Automatic modulation classification using cnn-lstm based dual-stream structure. IEEE Transactions on Vehicular Technology 69(11), 13521–13531 (2020)
17. Yu, W., Miao, L., Jie, Y., Guan, G.: Data-driven deep learning for automatic modulation recognition in cognitive radios. IEEE Transactions on Vehicular Technology 68(4), 4074–4077 (2019)
18. Sai, H., Yizhou, J., Yue, G., Zhiyong, F., Ping, Z.: Automatic modulation classification using contrastive fully convolutional network. IEEE Wireless Communications Letters 8(4), 1044–1047 (2019)
19. Ke, B., Yuan, H., Xiaojuan, J., Jindong, H.: Adversarial transfer learning for deep learning based automatic modulation classification. IEEE Signal Processing Letters 27, 880–884 (2020)
20. LiXin, L., Junsheng, H., Qianqian, C.: Automatic modulation recognition: A few-shot learning method based on the capsule network. IEEE Transactions on Broadcasting (99), 1–1 (2020)
21. Ian, G., Jean, P.-A., Bing, X., David, W.-F.: Generative adversarial nets. Neural Information Processing Systems, 2672–2680 (2014)
22. Martin, A., Soumith, C., Bottou, L.: Wasserstein gan (2017)
23. Ishaan, G., Faruk, A., Martin, A., Vincent, D., Aaron, C.: Improved training of wasserstein gans. Neural Information Processing Systems 30, 5767–5777 (2017)
24. Yanwei, F., Timothy, H., Tao, X., Shaogang, G.: Transductive multi-view zero-shot learning. IEEE Transactions on Pattern Analysis and Machine Intelligence 37, 2332–2345 (2015)
25. Bernardino, R.-P., Philip, T.: An embarrassingly simple approach to zero-shot learning 37, 2152–2161 (2015)
26. Scott, R., Zeynep, A., Honglak, L., Bernt, S.: Learning deep representations of fine-grained visual descriptions, 49–58 (2016)
27. Laurens, M., Geoffrey, H.: Visualizing data using t-sne. Journal of Machine Learning Research 9(86), 2579–2605 (2008)
28. Jia, D., Wei, D., Richard, S., Lijia, L., Kai, L., Feifei, L.: Imagenet: A large-scale hierarchical image database, 248–255 (2009)
29. Lin, T., Michael, M., Serge, Belongie: Microsoft coco: Common objects in context, 740–755 (2014)

Figure legends
Figure 1. AMC application scenarios in the Internet of Things.
Figure 2. The generation module of semantic attribute vector. The left part is the convolutional module and the right part is the convolutional encoding module.
Figure 3. The network architecture of SigZSLNet. WGAN-GP can generate the feature data of the unseen (missing) classes, D is the discriminator, and G is the generator. The classifier consists of a fully connected layer with the neuron number 128 and a softmax layer with the class 7.
Figure 4. The test accuracy comparisons under various SNRs (Group 1 and Group 2 have one class missing while Group 3 and Group 4 have two class missing).
Figure 5. The visualized feature vector: (a)The synthetic feature vector, (b)The real feature vector.
Figure 6. The confusion matrix comparisons of different groups under SNR=6 dB.
Figure 7. The visualized t-SNE of Group 4 under different SNRs.

Figures
Figure 1: AMC application scenarios in the Internet of Things.

Figure 2: The generation module of semantic attribute vector. The left part is the convolutional module and the right part is the convolutional encoding module.
Figure 3: The network architecture of SigZSLNet. WGAN-GP can generate the feature data of the unseen (missing) classes, D is the discriminator, and G is the generator. The classifier consists of a fully connected layer with the neuron number 128 and a softmax layer with the class 7.

Figure 4: The test accuracy comparisons under various SNRs (Group 1 and Group 2 have one class missing while Group 3 and Group 4 have two class missing).

Figure 5: The visualized feature vector: (a) The synthetic feature vector, (b) The real feature vector.
Figure 6: The confusion matrix comparisons of different groups under SNR=6 dB.

Figure 7: The visualized t-SNE of Group 4 under different SNRs.