Received 11 July 2024; accepted 24 July 2024. Date of publication 1 August 2024; date of current version 9 August 2024.

Digital Object Identifier 10.1109/OJCOMS.2024.3436640

Deep Learning-Based SNR Estimation

SHILIAN ZHENG[1], SHURUN CHEN[2], TAO CHEN[3], ZHUANG YANG[2], ZHIJIN ZHAO[2], AND XIAONIU YANG[1]

[1]Innovation Studio of Academician Yang, National Key Laboratory of Electromagnetic Space Security, Jiaxing 314033, China

[2]College of Communication Engineering, Hangzhou Dianzi University, Hangzhou 310000, China

[3]College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China

CORRESPONDING AUTHOR: S. ZHENG (e-mail: lianshizheng@126.com)

This work was supported in part by the National Natural Science Foundation of China under Grant U19B2016, and in part by the National Basic Scientific Research of China under Grant JCKY2023110C099.

ABSTRACT The signal-to-noise ratio (SNR) is an important metric for measuring signal quality and its estimation has received widespread attention in various application scenarios. In this paper, we propose an SNR estimation framework based on deep learning classification. Power spectrum input is proposed to reduce the computational complexity. We also propose an SNR estimation method based on deep learning regression to overcome the inevitable estimation error problem of classification-based methods in dealing with signals with SNR not within the training label set. We conduct a large number of simulation experiments considering various scenarios. Results show that the proposed methods have better estimation accuracy than two existing deep learning-based SNR estimation methods in different noises and multipath channels. Furthermore, the proposed methods only need to be trained under one modulation signals to adapt to SNR estimation of other modulation signals, with superior transfer performance. Finally, the method using the average periodogram as input has stronger adaptability in few-shot scenario and requires lower computational complexity compared to the method with in-phase and quadrature (IQ) input.

INDEX TERMS Signal-to-noise ratio, deep learning, convolutional neural network, classification, regression.

I. INTRODUCTION

SIGNAL-to-noise ratio (SNR) plays a crucial role in wireless communication systems. Compared to bit error rate (BER) and symbol error rate (SER) [1], SNR is a more direct parameter that reflects the signal quality or the channel quality. Additionally, a large number of channel decoding algorithms [2] require channel state information such as SNR to perform soft decoding [3], thus making the performance of SNR estimation critical for subsequent information recovery. Accurate SNR estimation can reduce the error rate of information recovery from the received signals and thus improve communication reliability.

Over the past few decades, many methods for estimating SNR have been proposed. These methods can be divided into two types: data-assisted (DA) estimators [4] and non-data-assisted (NDA) estimators [5]. DA estimators rely on prior knowledge of the transmitted data while NDA estimators obtain SNR estimates by analyzing the unknown received signal. As noted in [6], these SNR estimation methods possess certain limitations, such as restricted adaptability to various modulation types, constrained range of SNR, and a requirement for precise frequency and timing synchronization which are not always met in practical scenarios.

With the rapid development of deep learning (DL) in recent years, it has been widely used in the field of radio signal processing [7], including signal detection [8], signal recognition [9], information recovery [10], etc. Given the strong feature learning ability of deep learning, researchers also begin to apply deep learning to the challenging NDA SNR estimation [6], [11], [12]. These DL-based algorithms utilize classification methods to solve the SNR estimation task under additive white Gaussian noise (AWGN). Compared to traditional SNR estimation methods, DL-based SNR estimation methods significantly improve estimation performance and no longer rely on manually extracting signal features. However, the aforementioned DL-based

SNR estimation methods share some common limitations. Firstly, these methods do not consider complexity issues, and the signal input forms or network structures used result in high complexity, which may limit the practical implementation of these methods. For example, in [12] converting the input signal to a constellation diagram results in a high-dimensional input. The use of deep GoogleNet requires a high computational burden when used for SNR estimation. Secondly, they all use deep learning classification to solve the SNR estimation problem in continuous spaces. For example, in [11], the authors overlooked scenarios where the SNR of the actual signal doesn't align with the SNR categories of the training data. The simulated dataset contains fewer SNR categories, with intervals set at 2 dB or 4 dB, potentially resulting in an overestimation of accuracy. The SNR label resolution of classification methods will inevitably affect the estimation accuracy. Finally, the factors considered in simulation performance analysis are not comprehensive enough. For example, they only considered the basic AWGN scenario [6], [11], [12], without involving non-AWGN situations or considering adaptability to new signals. To overcome these problems, in this paper, we propose to use power spectrum as input to the adopted residual network to reduce computational complexity without compromising SNR estimation accuracy, which is especially important to reduce the energy consumption in mobile and remote communication systems. We also propose a solution based on deep learning regression to overcome the drawback of the estimation accuracy of classification based methods being affected by label resolution. We conduct a large number of simulation experiments considering a variety of nonideal scenarios to verify the performance of the proposed methods. Specifically, the main contributions of this paper are as follows.

- We propose an SNR estimation framework based on deep learning classification, which uses Residual Network (ResNet) as the classification network. In terms of network input, in addition to utilizing in-phase and quadrature (IQ) as input, we also propose using power spectrum as input. Compared to the IQ input method, it can reduce computational complexity without compromising estimation accuracy.
- We propose an SNR estimation method based on deep learning regression. This method overcomes the inevitable estimation error problem of classification-based methods in dealing with signals with SNR not within the training label set. Compared to classification-based methods, the regression-based methods require fewer training labels and have superior estimation performance when faced with untrained SNRs.
- We conduct a large number of simulation experiments, considering various scenarios such as different noise distributions including white noise and colored noise, multipath channels including Rayleigh and Rician distributions, and adaptability to new signals. The results indicate that the proposed classification-based methods perform better in terms of average SNR and mean squared error (MSE) compared to two existing deep learning-based SNR estimation methods, i.e., CDG-GoogleNet and IQ-CNN-LSTM. In addition, the proposed regression-based method has better estimation performance for untrained SNR compared to the classification-based method. Furthermore, in the case of different noise distributions including AWGN and additive general Gaussian noise (AGGN), the proposed method based on power spectrum input has better adaptability than the method based on IQ input. Finally, the proposed methods only need to be trained under one modulation signals to adapt to SNR estimation of other modulation signals, with superior transfer performance.
- We also conduct performance analysis in few-shot scenario, and the results show that the methods with power spectrum input have significant advantages over IQ input in very few sample situations, and is far superior to existing deep learning based methods, further demonstrating the superiority of our methods.
- We analyze the complexity of the proposed methods, including time complexity (floating-point operations, i.e., FLOPs) and space complexity (number of model parameters, i.e., Params). The results show that the FLOPs and Params of our proposed methods are much smaller than those of the existing deep learning-based SNR estimation methods, i.e., CDG-GoogleNet and IQ-CNN-LSTM. Furthermore, the proposed method using the average periodogram as input can further reduce the computational complexity to close to $1/L$ of that of the IQ input method, where L is the average number of times.

The remainder of this paper is organized as follows. In Section II, we introduce related work of SNR estimation. In Section III, we discuss the mathematical models of received signal and SNR estimation. In Section IV, the general framework of the proposed deep learning classification-based method is introduced in detail. In Section V the proposed regression-based method is discussed. Performance analysis, including simulation results and complexity analysis is given in Section VI. Finally, conclusions are drawn in Section VII.

The abbreviations used in this paper are summarized in Table 1.

II. RELATED WORK

A. TRADITIONAL SNR ESTIMATION METHODS

Traditional SNR estimation methods can be divided into two types: DA estimators [4] and NDA estimators.

DA estimators rely on prior knowledge of the transmitted data, examples of which include the minimum mean-square error estimation (MMSE) method [13], maximum likelihood (ML) estimation method [14], and squared signal-to-noise variance (SNV) estimator [15]. Based on ML estimation theory [16], the ML SNR estimator was introduced by Gagliardi and Thomas [17]. They derived the ML SNR

TABLE 1. Summary of abbreviations.

Abbreviations	Definitions
SNR	Signal-to-Noise Ratio
SER	Symbol Error Rate
BER	Bit Error Rate
DL	Deep Learning
DA	Data-Assisted
NDA	Non-Data-Assisted
MMSE	Minimum Mean-Square Error estimation
ML	Maximum Likelihood
SNV	Squared Signal-to-Noise Variance
PSK	Phase Shift Keying
BPSK	Binary Phase Shift Keying
MPSK	Multiple Phase Shift Keying
AWGN	Additive White Gaussian Noise
AGGN	Additive General Gaussian Noise
M2M4	Fourth-order Moment
SVR	Signal-to-Variation Ratio
SSME	Split-Symbol Moment Estimator
SOS	Second-Order Sections
PSD	Power Spectral Density
CNN	Convolutional Neural Network
IQ	In-phase and Quadrature
LSTM	Long Short Term Memory
CDG	Constellation Diagram
FLOPs	Floating Point Operations
Params	Number of Model Parameters
PG	Periodogram
APG	Average Periodogram
SGDM	Stochastic Gradient Descent with Momentum
MAE	Mean Absolute Error
MSE	Mean Squared Error
RMSE	Root Mean Squared Error
DFT	Discrete Fourier Transform

estimator for a binary phase shift keying (BPSK)-modulated signal in AWGN. The original formulation of the SNV estimator for BPSK modulation was first introduced in [18]. The authors then extended the application range of SNV estimator from BPSK modulation to high-order modulation in complex channel.

In contrast, NDA estimators obtain SNR estimates by analyzing the unknown received signal. Examples include second and fourth-order moment (M2M4) estimation method [5], split-symbol moment estimator (SSME) [19] and signal-to-variation ratio (SVR) estimator [20]. Specifically, Benedict et al. [21] proposed applying M2M4 to the noise intensity estimation of the actual AWGN channel, and Matzner [22] gave a detailed derivation of the M2M4 SNR estimator and obtained a similar expression to [22]. Simon and Mileant proposed the SSME algorithm by assuming that the BPSK signal is used in a wideband AWGN channel [23]. Brandão proposed the SVR estimator [24] which is a moment-based method that can measure channel quality in AWGN channels. The SVR estimator is designed to process arbitrary phase shift keying (PSK) modulation signals, and it is generally not suitable for other digital modulation schemes.

As pointed out in [6], there are several limitations of the traditional SNR estimation methods. Firstly, the types of signal modulation that these methods can apply to are limited. For example, the maximum likelihood estimator is effective for its target modulation, while the M2M4 estimator only applies to multiple phase shift keying (MPSK) signals. Secondly, the range of SNR that can be effectively estimated is narrow. For instance, the subspace-based estimation algorithm may not be optimal under low SNR conditions, as the dimension of the signal subspace is susceptible to overestimation in such scenarios. Finally, these methods usually assume that the receiving system is perfectly synchronous, that is, there is no frequency and timing offsets between the transmitter and the receiver which are not always met in practical scenarios.

B. DEEP LEARNING-BASED SNR ESTIMATION METHOD

Existing DL-based SNR estimation methods primarily utilize IQ signals or signals transformed from IQ signals as inputs to neural networks. Specifically, in [6], the authors proposed a convolutional neural network (CNN)-based SNR estimation method which uses the raw IQ signal as the input. Simulation results shown that the method is more robust and has a wider application range of modulation types than traditional SNR estimation method. However, the CNN used in this method is a basic CNN with several stacked convolutional layers which may restrict its performance. Furthermore, the method assigns the signal's amplitude as the label, estimates this amplitude using deep learning techniques, and subsequently computes the estimated SNR. Since it relies on an analytical expression to derive the SNR from the estimated amplitude, it may struggle to adapt effectively to complex channel environments.

In [11], the authors adopted a combination of CNN and long short term memory (LSTM) to estimate the SNR in long term evolution (LTE) and five-generation (5G) systems. Raw IQ input is also considered in this method. Simulation results shown that the CNN-LSTM based prediction of SNR in LTE and 5G systems has better accuracy and latency than traditional estimation methods. However, the method addressed the SNR estimation problem using a classification approach. It overlooked scenarios where the SNR of the actual signal doesn't align with the SNR categories of the training data. The simulated dataset contains fewer SNR categories, with intervals set at 2 dB or 4 dB, potentially resulting in an overestimation of accuracy.

In [12], a deep learning SNR estimation algorithm with constellation diagram (CDG) input was proposed, which improves the SNR estimation performance. Simulations shown that the proposed CDG-GoogleNet method is significantly better than the traditional SNR estimation algorithm

TABLE 2. Comparison of different SNR estimation methods.

Algorithm	IQ-CNN	IQ-CNN-LSTM	CDG-GoogleNet
Problem Formulation	Classification	Classification	Classification
Input Format	IQ	IQ	CDG
Network Structure	CNN	CNN-LSTM	GoogleNet
Output	Amplitude	SNR	SNR
AWGN	✔	✔	✔
AGGN	×	×	×
Colored Noise	×	×	×
Multipath Channels	×	×	×
Few-shot Scenario	×	×	×
New Signals	×	×	×
Complexity	High	Medium	High
FLOPs	1.95184×10^8	1.25015×10^7	1.58182×10^9
Params	4.25701×10^5	2.49219×10^6	6.07708×10^6

in the accuracy of SNR estimation and the adaptability of changing noise background environment. However, it's important to note that the simulations were conducted within a relatively narrow SNR range. This limitation raises questions about the generalizability of the method across broader SNR ranges encountered in real-world scenarios. Furthermore, converting the input signal to a constellation diagram and then using GoogleNet is computational expensive which may pose a potential obstacle to its real-world implementation.

For convenience we will refer to the methods proposed in [6], [11] and [12] as IQ-CNN, IQ-CNN-LSTM and CDG-GoogleNet respectively in the rest of the paper. We summarize the various factors considered by these methods in Table 2. It is evident that all of them utilize classification techniques to tackle the SNR estimation challenge. Furthermore, the considered scenarios are notably restricted, predominantly focusing on AWGN conditions. Lastly, these approaches tend to exhibit relatively high complexity levels which may restrict their real-world deployment. The FLOPs and Params are obtained with a signal length of 1024.

III. SYSTEM MODEL

SNR estimation is usually performed on the received signal. In a typical communication system, the signal sent by the transmitter is transmitted through the channel to the receiving end, and the signal received by the receiving end can be represented as

$$y(n) = h(n) \otimes b(n)e^{j(2\pi \Delta fn+\theta)} + w(n), n \in [0, N-1], \quad (1)$$

where the transmitted signal is represented by $b(n)$, the channel response for the transmitted signal is denoted by $h(n)$, the parameter Δf represents the carrier frequency offset, θ represents the random phase shift, $w(n)$ represents the noise, N is the signal length, and $\otimes$ represents convolution operation. Traditional methods face challenges in accurately separating the signal component from noisy signals and precisely estimating SNR due to the impact of channel propagation and the uncertainty regarding the noise type in advance.

According to the received signal $y(n)$ with additive noise, the noise-removed part of the received signal is defined as the effective signal $x(n)$, which can be represented as

$$x(n) = h(n) \otimes b(n)e^{j(2\pi \Delta fn+\theta)}. \quad (2)$$

The power of signal can be defined as

$$P_x = \lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} |x(n)|^2, \quad (3)$$

and the power of the noise can be defined as

$$P_w = \lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} |w(n)|^2. \quad (4)$$

Thus we define SNR of $y(n)$ as

$$\text{SNR} = \frac{P_x}{P_w}. \quad (5)$$

In general, decibels (dB) is used to express the signal-to-noise ratio:

$$\text{SNR}_{\text{dB}} \triangleq \rho = 10\log_{10}\left(\frac{P_x}{P_w}\right). \quad (6)$$

The SNR estimation problem is to estimate the SNR of the received signal $y(n)$, which can be expressed as

$$\widehat{\rho} = \max_{\rho} \Pr\{\rho|y(n)\}, \quad (7)$$

where $\widehat{\rho}$ is the estimated result and $\Pr\{\cdot\}$ represents the probability.

Generally, AWGN is considered to model the noise distribution. However, non-ideal noise exists in real world systems. In this paper, we also consider another white noise, AGGN, which can better represent the "pulse" noise. The probability density function of AGGN [10] is

$$p(\omega) = \frac{\kappa}{2\delta\Gamma\left(\frac{1}{\kappa}\right)} \exp\{-|\frac{\omega-\mu}{\delta}|^{\kappa}\}, \quad (8)$$

where ω is a random variable, μ is the mean, κ is the shape parameter, δ is the standard deviation, and $\Gamma(\cdot)$ is the Gamma function. When $\kappa = 2$, (8) becomes traditional AWGN distribution:

$$p(\omega) = \frac{1}{\sqrt{2\pi}\cdot\delta} \exp\left\{-\left|\frac{\omega-\mu}{\delta}\right|^2\right\}. \quad (9)$$

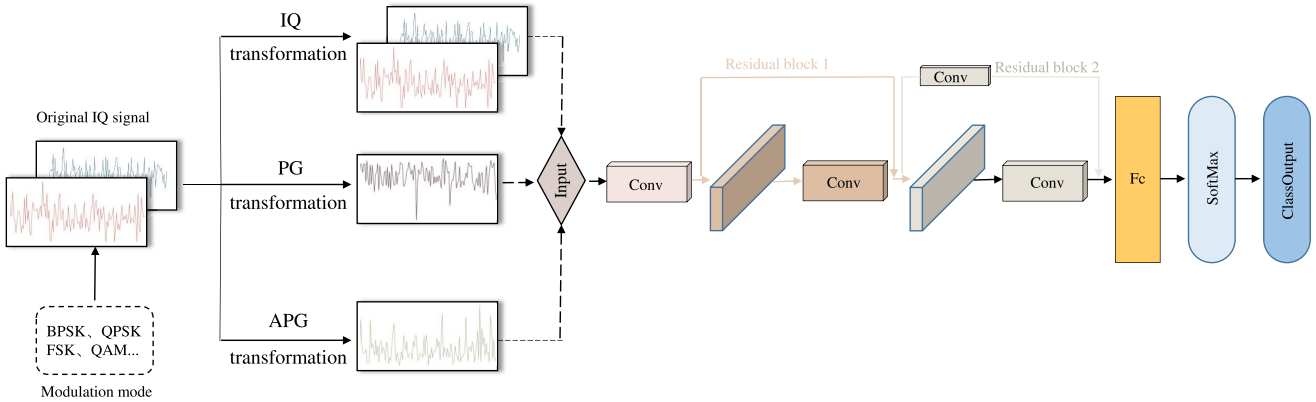

FIGURE 1. The overall framework of deep learning classification-based SNR estimation. The dotted lines represent the selection of a preprocessing method to obtain the network input, Conv represents the convolution layer, and Fc represents the fully connected layer. This framework includes two residual blocks, a SoftMax layer, and a ClassOutput layer.

Beyond white noises, our consideration extends to colored noise, specifically pink noise. This type of colored noise is produced by subjecting uniformly distributed random numbers to a sequence of randomly initialized second-order sections (SOS) filters. The amplitude distribution of the resulting pink noise is quasi-Gaussian, confined within the range of -1 and 1. Furthermore, the power spectral density (PSD) of this pink noise is inversely proportional to frequency:

$$P(f) \propto \frac{1}{f}, \tag{10}$$

where f is frequency.

In a wireless communication channel environment, the signal undergoes transmission to the receiver through various paths, such as reflection, refraction, and scattering. The total signal intensity follows the Rayleigh distribution as a result, leading to what is known as Rayleigh fading. If the received signal comprises not only reflections and scattering but also a direct transmission from the transmitter to the receiver, the total signal intensity adheres to the Rician distribution, and this phenomenon is termed Rician fading. The probability density function of Rayleigh distribution [25] is

$$p(\omega) = \frac{\omega}{\delta^2} \exp\left\{-\frac{\omega^2}{2\delta^2}\right\}, \tag{11}$$

where ω is a random variable, δ is the standard deviation. The probability density function of Rician distribution [26] is

$$p(\omega) = \frac{\omega}{\delta^2} \exp\left\{-\frac{\omega^2 + v^2}{2\delta^2}\right\} I_0\left(\frac{wv}{\delta^2}\right), \tag{12}$$

where ω is a random variable, δ is the standard deviation and I_0 is the modified zero-order Bessel function of the first kind. When $v = 0$, the Rician distribution degenerates into the Rayleigh distribution.

IV. PROPOSED CLASSIFICATION-BASED METHOD

A. OVERALL FRAMEWORK

SNR estimation is essentially a parameter estimation problem in continuous space. One way to solve it is to discretize the SNR according to a specific range and convert it into a classification problem in a finite set. This is advantageous for training the SNR estimation model borrowing inspiration from computer vision methods. The classification model exerts a compelled classification influence on the SNR. The SNR is categorized into the nearest pre-set SNR category, facilitating convenient statistical analysis and observation. The discretized set can be represented as

$$\boldsymbol{\rho} = \{\rho_{\min}, \rho_{\min} + \rho_\alpha, \ldots, \rho_{\max} - \rho_\alpha, \rho_{\max}\}, \tag{13}$$

where $\boldsymbol{\rho}$ is the set of all possible SNRs, $\rho_{\min}$ is the minimum SNR, $\rho_{\max}$ is the maximum SNR, and ρ_α is the resolution of the SNR estimate. In this way, we can consider SNR estimation as a classification problem, where the number of categories is

$$C = (\rho_{\max} - \rho_{\min})/\rho_\alpha + 1. \tag{14}$$

In this paper, we adopt deep learning to solve the classification problem. The overall framework of our proposed SNR estimation method is shown in Fig. 1 where we consider a variety of input data formats, including raw IQ input and power spectrum input (with average and without average). The dotted lines represent the selection of a preprocessing method to obtain the network input, Conv represents the convolution layer, and Fc represents the fully connected layer. This framework includes two residual blocks, a SoftMax layer, and a ClassOutput layer.

Selecting one of the three input preprocessing methods to preprocess the raw signal, and use the processed signal as the input to the CNNs. These input formats have different sizes, which may lead to different computational complexity in the inference phase. Therefore, we can flexibly select the appropriate input forms according to specific application scenarios. We build corresponding CNNs for these input

formats. Except for the first layer, which is designed to match the dimensions of its processed data, the rest of the CNNs share the same architecture. For each input, The neural network model is trained by using the constructed training set. In the test stage, test data is fed into the model obtained after training to obtain the estimation results of SNR.

B. INPUT FORMAT

1) RAW IQ INPUT

Firstly, the IQ components of the received signal $y(n)$ can be extracted as

$$I(n) = \text{real}(y(n)), \tag{15}$$

$$Q(n) = \text{imag}(y(n)), \tag{16}$$

where the I and Q components of the received signal are denoted by $I(n)$ and $Q(n)$, respectively, and real$(\cdot)$ and imag$(\cdot)$ represents the extraction of the real and imaginary components from the received signal $y(n)$. Afterwards, the extracted $I(n)$ and $Q(n)$ are concatenated to form a matrix

$$\mathcal{A} = \begin{bmatrix} I(0) & I(1) & \cdots & I(N-1) \\ Q(0) & Q(1) & \cdots & Q(N-1) \end{bmatrix} \tag{17}$$

where N is the length of the sequences. The matrix $\mathcal{A}$ with size $2 * N$ can be used as the input of the CNN.

2) POWER SPECTRUM INPUT

Since the power spectrum of white noise is often flat, and the power spectrum of signals is often not flat due to the influence of pulse shaping, signals with different SNR will behave differently on the power spectrum. Based on this, we can use the power spectrum as the input of the CNN to learn and extract features to distinguish different SNRs. The power spectrum of the signal can be estimated by first computing the discrete Fourier transform (DFT) of the received signal $y(n)$:

$$Y(k) = \sum_{n=0}^{N-1} y(n) e^{-jk\frac{2\pi}{N}n}, k = 0, 1, \ldots, N-1. \tag{18}$$

Then the resulting sequence is shifted with the frequency zero moved to the center of the spectrum. After that, the amplitude is calculated and the resulting power spectrum is

$$R(k) = 10\log_{10}|\hat{Y}(k)|^2, k = 0, 1, \ldots, N-1, \tag{19}$$

where $\hat{Y}(k)$ denotes the shifted sequence of $Y(k)$, $|\cdot|$ means getting the absolute value. The purpose of $10\log_{10}(\cdot)$ computation is to make the calculated power spectrum more match the SNR expressed in dB. This process follows the procedure of PG. With $R(k)$, we can construct the matrix as the input of the CNN as

$$\mathcal{B} = \begin{bmatrix} R(0) & R(1) & \cdots & R(N-1) \end{bmatrix}, \tag{20}$$

where matrix $\mathcal{B}$ is with size $1 * N$ which is smaller than that of matrix $\mathcal{A}$. For simplicity, we refer to this input as PG input.

In order to reduce the variance of periodograms in power spectrum estimation, APG has been proposed. The N points of the signal are divided into L segments, each segment of data length M, and the power spectrum is estimated respectively for each segment of data, and then the average value is calculated. The average periodogram changes the original variance to $1/L$. When $L = 1$, the average periodogram method is equal to the previously discussed periodogram method. As the number of segments L increases, the variance decreases, but the resolution decreases. The average periodogram can be expressed as follows:

$$Y_i(k) = \sum_{n_i=(i-1)\ \times M}^{i\times M-1} y(n_i) e^{-jk\frac{2\pi}{N}n_i}, i = 1, \ldots, L, \tag{21}$$

$$S(k) = 10\log_{10}\left(\frac{1}{L}\sum_{i=1}^{L}\left|\hat{Y}_i(k)\right|^2\right), k = 0, \ldots, M-1, \tag{22}$$

where $\hat{Y}_M(k)$ represents the shifted sequence of $Y_M(k)$ and $S(k)$ is the obtained APG. Similar as PG, the purpose of $10\log_{10}(\cdot)$ computation is to make the calculated power spectrum more match the SNR expressed in dB. With the obtained $S(k)$, we can construct the matrix as the input of the CNN as

$$\mathcal{C} = \begin{bmatrix} S(0) & S(1) & \cdots & S(M-1) \end{bmatrix}. \tag{23}$$

It can be seen that the size of $\mathcal{C}$, i.e., $1*M$, obtained from the average periodogram is further reduced. The computational overhead associated with performing inference tends to scale with the size of the input signals, so the computational complexity using $\mathcal{C}$ as the input is reduced by L times compared with that of using $\mathcal{B}$ as the input. Similarly, we refer to this input as APG input for simplicity in the rest of the paper.

C. ADOPTED NETWORKS

1) BASIC CNN

Basic structure of CNN includes the convolutional layer, the pooling layer, and the fully connected layer. The convolution kernel in the convolution layer realizes the convolution function. It is postulated that the m-th layer in the network corresponds to a convolutional layer, the input feature map is $a_{m-1} \in \mathbb{R}^{P\times Q\times D}$, and the output feature is $o_m \in \mathbb{R}^{P'\times Q'\times U'}$, then the u-th element of the output feature map is

$$o_m^u = \sum_{d=1}^{D} w_m^{(u,d)} \otimes a_{m-1}^d + b_m, \tag{24}$$

where $w_m^{(u,d)}$ is the weight of the convolution kernel, and b_m stands for bias of the convolution kernel. Convolution is similar to the filtering function, which can eliminate the influence of channel environment in (1) and is beneficial for SNR estimation.

The pooling layer is used to alleviate the excessive sensitivity of the convolution layer to the position. The

pooling layer computes the output at a time on elements in a fixed region (also known as the pooling window) of the input data. The output of the maximum pooling layer used is defined as:

$$\text{MaxPool}\left(R_{p,q}^{u}\right) = \max_{i \in R_{p,q}^{u}} a_i, \tag{25}$$

where $R_{p,q}^{u}$ is the target area in the input feature map of the pooling layer. There is also average pooling defined as:

$$\text{AveragePool}\left(R_{p,q}^{u}\right) = \frac{1}{|R_{p,q}^{u}|} \sum_{i \in R_{p,q}^{u}} a_i, \tag{26}$$

where $|R_{p,q}^{u}|$ represents the number of neurons in region $R_{p,q}^{u}$.

There is usually a nonlinear layer between the convolution layer and the pooling layer, which is used to increase the nonlinear expression ability of the network. The commonly used nonlinear function is rectified linear unit (ReLU) which is defined as

$$\text{ReLU}(a) = \max(a, 0). \tag{27}$$

Fully connected layer is generally located at the end of the network for mapping the extracted feature map to a fixed number of neurons. Each neuron in the layer is connected with all elements of the previous layer.

Classification problems require discrete predictive outputs, and there are two problems with using outputs directly from the output layer. On the one hand, since the range of output values in the output layer is uncertain, it is difficult to intuitively judge these values' meaning. On the other hand, since the real labels are discrete values, the error between these discrete values and the output values of the uncertain range is difficult to measure. Softmax operator can effectively solve the above two problems by

$$\text{SoftMax}(c_i) = \frac{\exp(c_i)}{\sum_k \exp(c_k)}, \tag{28}$$

where c_i represents the i-th element of the output array with C elements. It transforms the output into a probability distribution with positive values and a sum of "1". Via this operation, we change from the initial observation to the confidence of each predicted value. The higher the confidence, the higher the probability that the signal belonging to this category.

2) ADOPTED RESNET

With the development of DL, AlexNet [27], VGGNet [28], GoogleNet [29] and other image recognition network models have been proposed one after another. Upon comparing these advanced neural networks with their predecessors, researchers identified three key factors that influence the performance of CNNs: convolution kernel size, network width, and network depth. Generally, increasing the depth of the network layers enhances overall performance. As the network layers deepen, the accuracy of the model steadily improves until it reaches a peak. However, there exists a threshold for performance improvement by deepening the layers of the network. Excessive layer depth leads to a significant decrease in accuracy, a phenomenon known as "degradation".

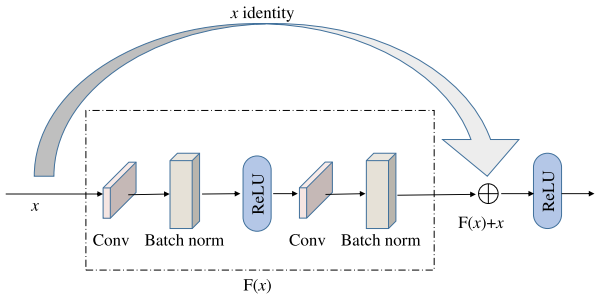

FIGURE 2. The basic structure of the residual block.

ResNet [30] has been proposed to solve this problem by introducing the residual block as shown in Fig. 2. On one hand, the input x normally passes through various network layers. On the other hand, there is also a shortcut route that directly connects the input x to the output $F(x)$ and taking the resulting $F(x) + x$ as input to the next activation function ReLU. The residual network enhances the efficiency of information transmission and reduces the number of network parameters by incorporating residual connections into the non-linear convolutional layer. This enables the direct transmission of low-level features from the network to the high-level, ensuring that even if certain intermediate layers do not contribute meaningful transformations, they can still convey information from the preceding layer. When the number of network layers surpasses the optimal count, the surplus layers are transformed into identity mappings. As a result, the network performance remains unaffected despite the excessive deepening of the layer count.

We construct the adopted ResNet shown in Fig. 3 based on the basic ResNet18, which consists of a convolution layer, two residual blocks, and a fully connected layer. The input matrix first passes through a convolution layer and a maximum pooling layer. The size of the convolution kernel in the convolution layer needs to be set according to different input formats, i.e., when using IQ input, the size of the convolutional kernel is $2 * 15$, while when using PG and APG input, the size is $1 * 15$. Then the output of the maximum pooling layer passes through two residual blocks. Details of the residual blocks are shown in Table 3 without the discussion of the normalized layer and the ReLU layer for the sake of simplicity. It is worth mentioning that the inclusion of the "conv5" layer is aimed at addressing the issue of inconsistent dimensions observed before and after the residual block. Next, the output of the second residual block passes through the average pooling layer and the fully connected layer. Finally, softmax is used to obtain the confidence that the signal belongs to a specific category of SNR. We considered both enhancing model estimation performance and reducing computational complexity, so the constructed network employs only two residual blocks.

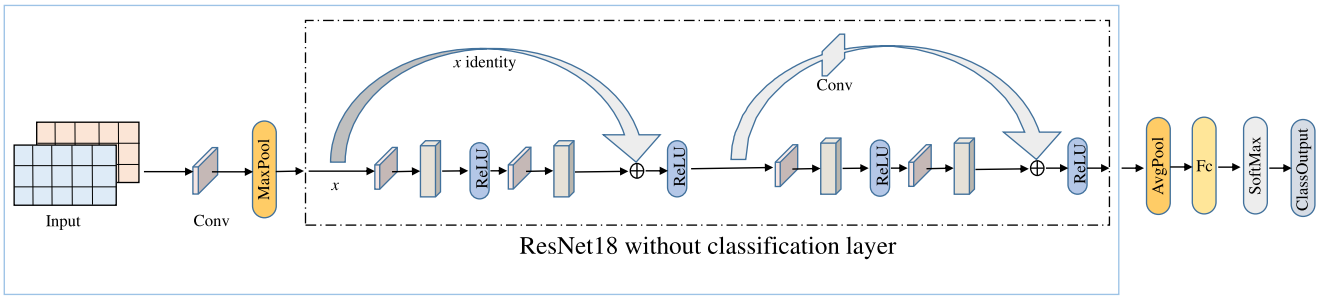

FIGURE 3. Adopted ResNet network structure. "MaxPool" represents the maximum pooling layer, "AvgPool" represents the average pooling layer and "Fc" represents the fully connected layer.

TABLE 3. Parameter setting of residual blocks.

Block \ Parameter	Layer	KernelSize	Stride	KernelNumber
Residual-block1	conv1	1×3	(1,1)	64
	conv2	1×3	(1,1)	64
Residual-block2	conv3	1×3	(1,2)	128
	conv4	1×3	(1,1)	128
	conv5	1×1	(1,2)	128

In practical applications, signals are often more complex, necessitating deeper network layers to achieve satisfactory training outcomes. Consequently, the number of residual blocks is increased based on actual requirements. The ResNet network effectively mitigates the 'degradation phenomenon,' ensuring that an excessive number of network layers does not result in poor training results.

D. TRAINING AND INFERENCE

The parameters of the constructed network model need to be optimized based on the training set. Assuming that there are m pairs of data in the training set, which can be represented as

$$\left\{\left(x_{train}^{(1)}, z_{train}^{(1)}\right), \left(x_{train}^{(2)}, z_{train}^{(2)}\right), \ldots, \left(x_{train}^{(m)}, z_{train}^{(m)}\right)\right\}, \quad (29)$$

where $x_{train}^{(i)}$ represents the i-th signal sample and $z_{train}^{(i)}$ is the corresponding label. In the case of forward propagation, the final output of the network using $x_{train}^{(i)}$ as the input is

$$\hat{z}_{train}^{(i)} = \mathcal{G}_W\left(x_{train}^{(i)}\right), \quad (30)$$

where W is the set of the parameters of the network represented by $\mathcal{G}_W(\bullet)$, and $\hat{z}_{train}^{(i)}$ represents the output of $x_{train}^{(i)}$ after passing through the network. Given a mini-batch of training samples, cross-entropy is used as the loss function for optimizing the parameters of the network

$$\text{Loss}_{\text{CE}} = -\frac{1}{B}\sum_{i=1}^{B}\sum_{j=1}^{C} z_{train,j}^{(i)} \log \hat{z}_{train,j}^{(i)}, \quad (31)$$

where $z_{train,j}^{(i)}$ is the true SNR label of the i-th signal sample and B is the mini-batch size.

During the training of the neural network, the optimization algorithm is used to iteratively adjust the model parameters to minimize the value of the Loss function. To update the set of parameter W, we use stochastic gradient descent with momentum (SGDM) algorithm [31]. Momentum is a method that helps accelerate SGD in the relevant direction and dampens oscillations. This involves randomly selecting a mini-batch of samples in each iteration, which is then used to calculate the gradient of the Loss function with respect to the parameters as

$$\vartheta_t = \gamma \vartheta_{t-1} + \eta \nabla_\theta \text{Loss}_{\text{CE}}, \quad (32)$$

$$\theta \leftarrow \theta - \vartheta_t. \quad (33)$$

where γ is the momentum, θ is the parameter in W to be updated and η is the learning rate. After the model training is completed, in the inference stage, when a new signal arrive, it is fed into the network in the corresponding form, and the class with the highest confidence output is used as the SNR estimation result. The training and inference process of classification-based SNR estimation is specifically described in Algorithm 1.

V. PROPOSED REGRESSION-BASED METHOD

In classification-based solution, in the inference stage, when input a signal, the trained model will output an SNR in the predefined SNR set $\boldsymbol{\rho}$ as the estimated result. As shown in Fig. 4, when the true SNR of the input signal is not in the set $\boldsymbol{\rho}$, the network will forcibly select the SNR category with the highest output confidence from the training set $\boldsymbol{\rho}$ as the estimated value. Obviously, if the number of SNR in the predefined set $\boldsymbol{\rho}$ is small and the resolution of SNR is insufficient, even if the classification performance is perfect, there will inevitably be a large deviation in the estimation of SNR in the reasoning stage.

To solve this problem, we propose SNR estimation based on deep learning regression. Unlike the classification method that mandates output within preset categories, the regression method produces a continuous fitting curve. When employing the regression method to train the SNR estimation model, as the estimation performance improves, the curve progressively converges towards the actual SNR of the signal, which can give an estimate close to the true SNR even

Algorithm 1 Training and Inference Algorithms of Classification-Based SNR Estimation Methods

Training procedure

Input: Training set $\mathcal{D} = \{(x_{train}^{(1)}, z_{train}^{(1)}), (x_{train}^{(2)}, z_{train}^{(2)}). ., \ldots, (x_{train}^{(m)}, z_{train}^{(m)})\}$, minibatch size B, learning rate η, maximum iterations λ, momentum variable γ.

Prepossess the input data according to the specific input format, i.e, IQ, PG or APG.
Initialize the parameters of the network.
for $1 \leq t \leq \lambda$ **do**
Choose B samples from the training set $\mathcal{D}$;
Estimate loss using (31);
Update the parameters of the network using (32)(33).
end for

Output: The trained model $\mathcal{G}_W(\bullet)$.

Inference procedure

Input: Signal $y(n)$, the trained model $\mathcal{G}_W(\bullet)$.

Prepossess the signal according to the specific input format, i.e, IQ, PG or APG.
Compute $\mathcal{G}_W(\mathcal{A})$, $\mathcal{G}_W(\mathcal{B})$, or $\mathcal{G}_W(\mathcal{C})$ according to the specific input format.
Choose the category with the maximum confidence as the estimated SNR.

Output: The estimated SNR $\hat{\rho}$.

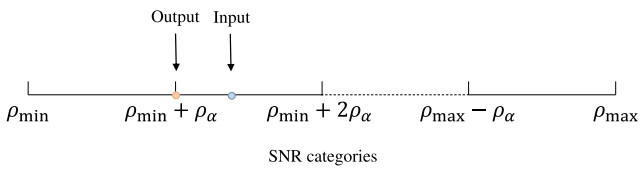

FIGURE 4. The inference of SNR not within the training label set with the classification-based methods.

if the SNR of the input signal is untrained. The most significant difference between the regression model and the classification model is the output layer. The total number of neurons in the last fully connected layer of classification model is set equal to the number of SNR categories, say C in this paper. Softmax layer is then used to obtain the confidences. Differently from this, in the regression model, there is only one neuron in the last fully connected layer, and the output can be used directly as the estimated SNR, without the need for the softmax layer. The rest of the framework is kept unchanged, which is shown in Fig. 5. There are also differences in the training of the model. Firstly, the label used by the classification model is the one-hot encoded categories, while the label used by the regression model is the SNR value itself. Secondly, the loss function used in training the regression model is the root mean squared error (RMSE) which can be represented as

$$\text{Loss}_{\text{RMSE}} = \sqrt{\frac{1}{B}\sum_{i=1}^{B} |\rho_i - \hat{\rho}_i|^2}, \tag{34}$$

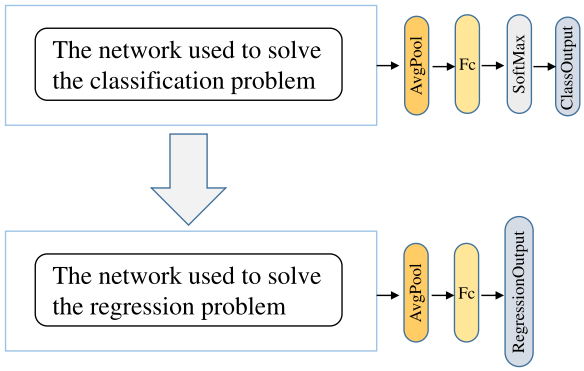

FIGURE 5. Transferring classification-based network to regression-based network.

Algorithm 2 Training and Inference Algorithms of Regression-Based SNR Estimation Methods

Training procedure

Input: Training set $\mathcal{D} = \{(x_{train}^{(1)}, z_{train}^{(1)}), (x_{train}^{(2)}, z_{train}^{(2)}). ., \ldots, (x_{train}^{(m)}, z_{train}^{(m)})\}$, minibatch size B, learning rate η, maximum iterations λ, momentum variable γ.

Prepossess the input data according to the specific input format, i.e, IQ, PG or APG.
Initialize the parameters of the network.
for $1 \leq t \leq \lambda$ **do**
Choose B samples from the training set $\mathcal{D}$;
Estimate loss using (34);
Update the parameters of the network using (35)(33).
end for

Output: The trained model $\mathcal{G}_W(\bullet)$.

Inference procedure

Input: Signal $y(n)$, the trained model $\mathcal{G}_W(\bullet)$.

Prepossess the signal according to the specific input format, i.e, IQ, PG or APG.
Compute $\mathcal{G}_W(\mathcal{A})$, $\mathcal{G}_W(\mathcal{B})$, or $\mathcal{G}_W(\mathcal{C})$ according to the specific input format.
Gradually fitting curves to represent the mapping relationship between input signals and SNR.

Output: The estimated SNR $\hat{\rho}$.

where ρ_i represents the true SNR and $\hat{\rho}_i$ represents the predicted SNR. The regression loss function used to update the parameters as

$$\vartheta_t = \gamma \vartheta_{t-1} + \eta \nabla_\theta \text{Loss}_{\text{RMSE}}, \tag{35}$$

The training and inference process of regression-based SNR estimation is specifically described in Algorithm 2.

VI. PERFORMANCE ANALYSIS

A. SIMULATION SETTING

1) DATASET

We generate signals with different modulations through simulation to construct a basic dataset for SNR estimation. Details of the dataset simulation is shown in Table 4. Each

TABLE 4. The dataset information.

Item	Value
SNR range	$[-20, 30]$ dB with interval 0.5 dB
Modulations	BPSK, QPSK, 4PAM, 16QAM
Data dimension	2*1024
Training set	50500
Test set	10100
Noise	AWGN/AGGN/Colored
Channel	Rayleigh/Rician

TABLE 5. Parameter setting of Rayleigh channel.

Item	Value
SampleRate	10×10^3
PathDelays	$[0, 1.5 \times 10^{-4}]$
AveragePathGains	$[2, 3]$
MaximumDopplerShift	3
PathGainsOutputPort	True
NormalizePathGains	True

generated signal has a length of 1024 and an oversampling ratio of 8, implying that each signal comprises 128 symbols. A root raised-cosine filter with a 6-symbol truncated length is employed as the pulse-shaping filter, and the roll-off factor is chosen randomly from the interval $[0.2, 0.7]$. Moreover, the normalized frequency offset of each signal normalized is randomly selected from the range $[-0.2, 0.2]$. We consider three types of noise: AWGN, AGGN and pink. In AGGN, the arbitrary position parameter is set to 0, the shape parameter is set to 1.5 and the inverse scale parameter is set to 1. We also consider multipath channels. The parameter settings of Rayleigh channel and Rician channel are shown in the Table 5 and Table 6, respectively.

2) HYPERPARAMETER SETTING

The training of the model was conducted on a computer system equipped with an Intel Core i7-9750H CPU operating at 2.60 GHz, 32 GB of RAM, and a GeForce RTX 2080 GPU. All experiments were performed on Matlab2020b. Firstly, the deep neural network is trained using data from the training set. The training parameters are shown in Table 7. After completion of training, the final deep neural network is saved as the best-trained network model, and the performance of the SNR estimator based on deep learning is evaluated using test set data.

TABLE 6. Parameter setting of Rician channel.

Item	Value
SampleRate	1×10^6
KFactor	2.8
PathDelays	$[0, 1.5, 1.2] \times 10^{-6}$
AveragePathGains	$[0.1, 0.5, 0.2]$
DirectPathDopplerShift	5
DirectPathInitialPhase	0.5
MaximumDopplerShift	50
PathGainsOutputPort	True

TABLE 7. Training parameter setting.

Parameter	Value
Max epochs	25
Mini-batch size	64
Initial value of learning rate	0.005
Learning rate change period	6 epochs
Learning rate change factor	0.1
Optimizer	sgdm
Shuffle	Every-epoch
Validation Frequency	$50500/128 = 394$

3) PERFORMANCE METRICS

We use mean SNR, mean absolute error (MAE) and MSE to evaluate the performance of the methods. The mean SNR is defined as the mean value of the predicted SNR of the test samples

$$\text{MeanSNR} = \frac{1}{K}\sum_{i=1}^{K}\hat{\rho}_i, \tag{36}$$

where $\hat{\rho}_i$ is the predicted SNR of the i-th sample and K is the number of samples. MAE represents the mean of absolute error between the predicted value and the actual value which is defined as

$$\text{MAE} = \frac{\sum_{i=1}^{K}|\rho_i - \hat{\rho}_i|}{K}. \tag{37}$$

MSE represents the sample variance of the difference between the predicted value and the actual value, which can amplify the difference to some extent and make it easier to observe. MSE is defined as

$$\text{MSE} = \frac{\sum_{i=1}^{K}\left(\rho_i - \hat{\rho}_i\right)^2}{K}. \tag{38}$$

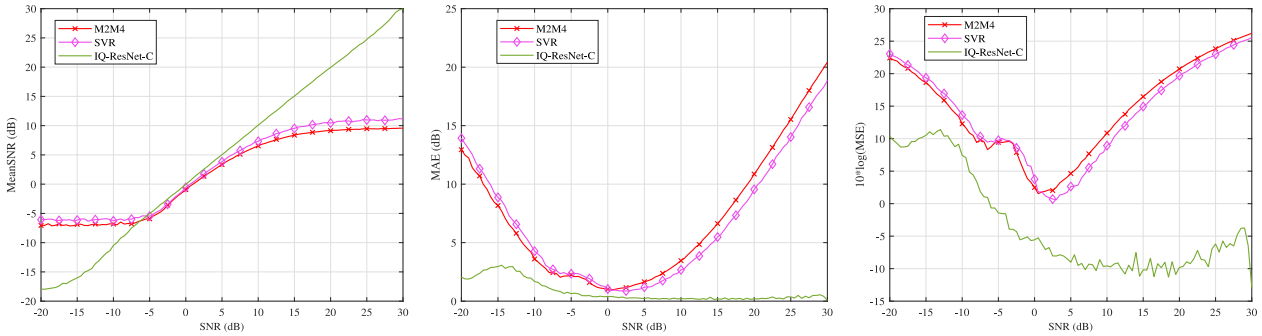

FIGURE 6. Comparison of Performance between Deep Learning-Based Algorithms and Traditional Estimation Algorithms. (a) Mean SNR, (b) MAE, and (c) MSE.

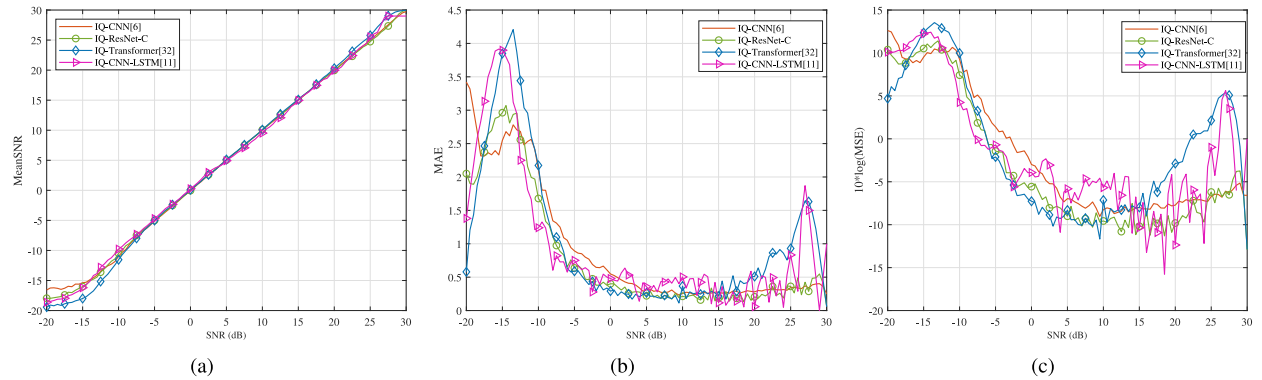

FIGURE 7. Performance of different networks. (a) Mean SNR, (b) MAE, and (c) MSE.

B. COMPARISON WITH TRADITIONAL ESTIMATION ALGORITHMS

M2M4 and SVR are two widely used traditional SNR estimation algorithms as mentioned in [18]. As shown in the Fig. 6, we compare the performance of our proposed deep learning-based SNR estimation algorithm with two traditional SNR estimation algorithms. As observed from Fig. 6(a), the average SNR estimated by the traditional SNR estimation algorithms M2M4 and SVR is close to the actual SNR only within the range of $[-5, 5]$ dB. The average SNR estimated by these traditional methods shows a significant deviation from the actual SNR in other regions. Specifically, traditional SNR estimation methods tend to overestimate SNR in the $[-20, -7]$ dB range and underestimate SNR in the $[6, 30]$ dB range. In addition, it can be seen that across the entire SNR range, the estimation error of the IQ-ResNet-C algorithm is consistently smaller than that of the two traditional SNR estimation algorithms from Fig. 6(b) and Fig. 6(c). Therefore, we can conclude that the estimation performance of our proposed IQ-ResNet-C algorithm is superior to that of traditional SNR estimation algorithms.

C. COMPARISON OF DIFFERENT NETWORKS

In order to validate the efficacy of the network employed in this paper, we compare the performance of utilizing IQ input across various networks: CNN as detailed in [6], CNN-LSTM as discussed in [11], Transformer as outlined in [32], and the ResNet architecture utilized in our research. As depicted in Figures 7(b) and 7(c), it is evident that the performance of IQ-CNN and IQ-ResNet-C methods align closely, as does the performance of IQ-CNN-LSTM and IQ-Transformer methods. Notably, within the SNR ranges of $[-20, -18]$ dB and $[-12, 26]$ dB, the estimation error of the IQ-ResNet-C method is lower than that of the IQ-CNN method, whereas in other SNR ranges, the estimation error of the IQ-ResNet-C method is marginally higher. Overall, both the IQ-Transformer and IQ-CNN-LSTM methods exhibit larger estimation errors within the $[-18, -12]$ dB and $[20, 30]$ dB ranges compared to the other two methods. As indicated in Table 8, our proposed IQ-ResNet demonstrates lower computational complexity compared to IQ-CNN while maintaining commendable estimation performance. Although the computational complexity of IQ-ResNet exceeds that of IQ-CNN-LSTM, it contributes to enhanced estimation accuracy in high SNR regions. Moreover, the IQ-ResNet method outperforms IQ-Transformer in terms of estimation performance, despite both methods having similar computational complexities. This experimental observation underscores the effectiveness of the ResNet network we adopted, which succeeds in maintaining a reasonable network complexity while preserving estimation performance.

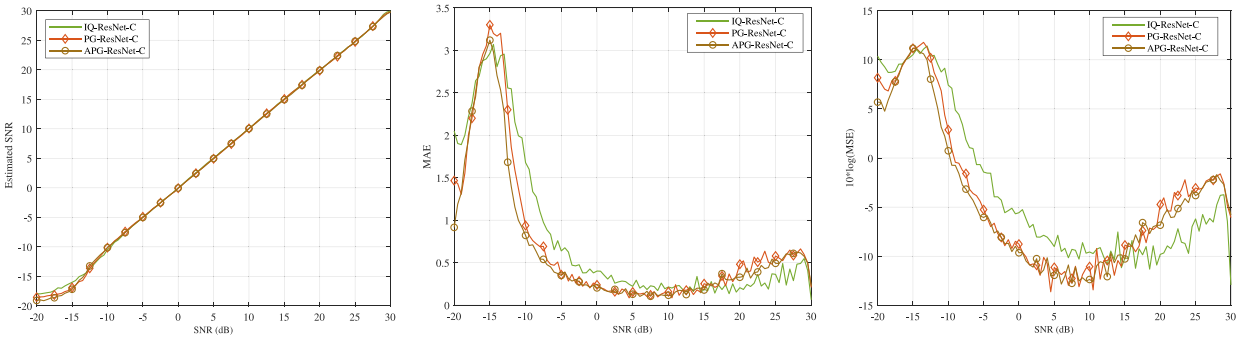

FIGURE 8. Performance of different input formats. (a) Mean SNR, (b) MAE, and (c) MSE.

TABLE 8. FLOPs and params of different methods.

Input \ Complexity	FLOPs	Params
IQ-ResNet-C	3.51664×10^7	$\mathbf{1.22213 \times 10^5}$
PG-ResNet-C	1.75899×10^7	$\mathbf{1.22213 \times 10^5}$
APG-ResNet-C	$\mathbf{4.28197 \times 10^6}$	$\mathbf{1.22213 \times 10^5}$
IQ-CNN-LSTM [11]	1.25015×10^7	2.49219×10^6
IQ-Transformer [32]	3.40811×10^7	5.41669×10^5
IQ-CNN [6]	1.95184×10^8	4.25701×10^5
CDG-GoogleNet [12]	1.58182×10^9	6.07708×10^6

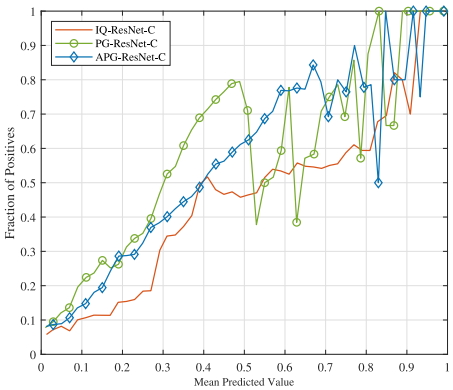

FIGURE 9. Calibration curves for different inputs.

D. COMPARISON OF DIFFERENT INPUT FORMATS

We compare the performance of our proposed DL-based SNR estimation methods with three input formats, i.e., IQ, PG and APG. We choose $L = 4$ in the computation of APG. For simplicity, the methods based on classification with IQ input, PG input and APG input are denoted as IQ-ResNet-C, PG-ResNet-C and APG-ResNet-C respectively in the rest of the paper. In this experiment, QPSK signals and AWGN are used. The results are shown in Fig. 8. As for the average SNR shown in Fig. 8(a), we can see that the estimated average SNR with the three input formats in the extremely low SNR region $[-20, -13]$ dB is slightly higher than the actual SNR. PG-ResNet-C and APG-ResNet-C perform better than IQ-ResNet-C in this SNR region. In the rest of the SNR range, the estimated average SNR is very close to the true SNR, which verifies the superiority of the proposed methods. As for MAE and MSE shown in Fig. 8(b) and Fig. 8(c), we can observe that the MAE and MSE of PG-ResNet-C and APG-ResNet-C are lower than those of IQ-ResNet-C in the low-to-medium SNR range of $[-20, 15]$ dB and higher in the high SNR range of $[15, 30]$ dB. The input dimension of PG and APG is significantly smaller than that of IQ, so in most cases, the latter two methods, especially the APG-ResNet-C, can be selected to decrease the computational complexity which will discussed later.

We also provide calibration curves for the three proposed methods in Fig. 9, showcasing the relationship between the model probability outputs and the actual observed categories. We can observe that their overall trends closely approximate the 45-degree diagonal line, indicating excellent reliability in the models' outputs.

E. PERFORMANCE UNDER DIFFERENT NOISE DISTRIBUTIONS

1) PERFORMANCE IN WHITE NOISES

We now test the performance of the methods in two white noise distributions, i.e., AWGN and AGGN. Existing deep learning-based SNR estimation methods are used for comparison, including CDG-GoogleNet [12] and IQ-CNN-LSTM [11]. In addition to testing the performance of the model against the same noise, we also test the model's adaptability to different types of noise. Specifically, the model trained by the training set data under AWGN is used to test the data with the AGGN noise, which is denoted AWGN-AGGN, and vice versa, which is denoted as AGGN-AWGN. The methods using the same noise distribution in the training and testing are provided for comparison which are denoted as AWGN-AWGN and AGGN-AGGN. QPSK signals and classification-based methods are used. The results are shown in Fig. 10 and Fig. 11.

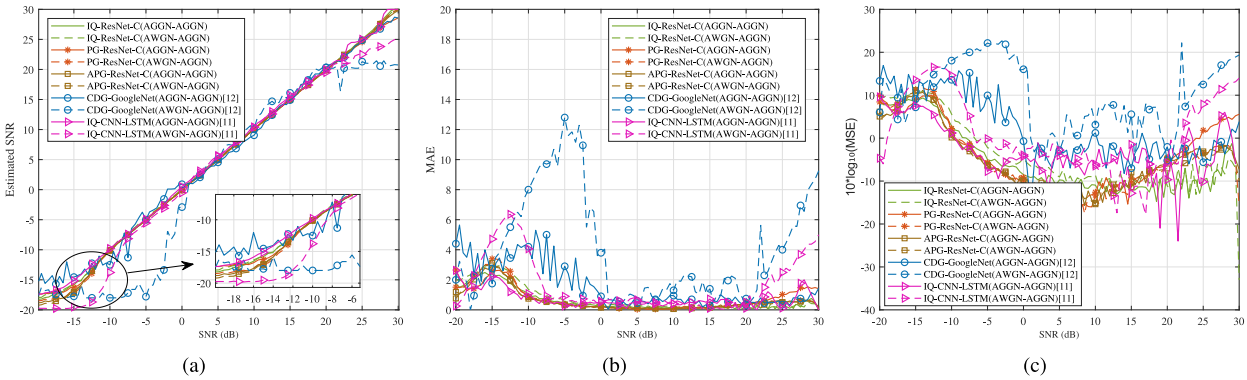

FIGURE 10. Performance under AGGN noise with AWGN/AGGN-trained models. (a) Mean SNR, (b) MAE, and (c) MSE.

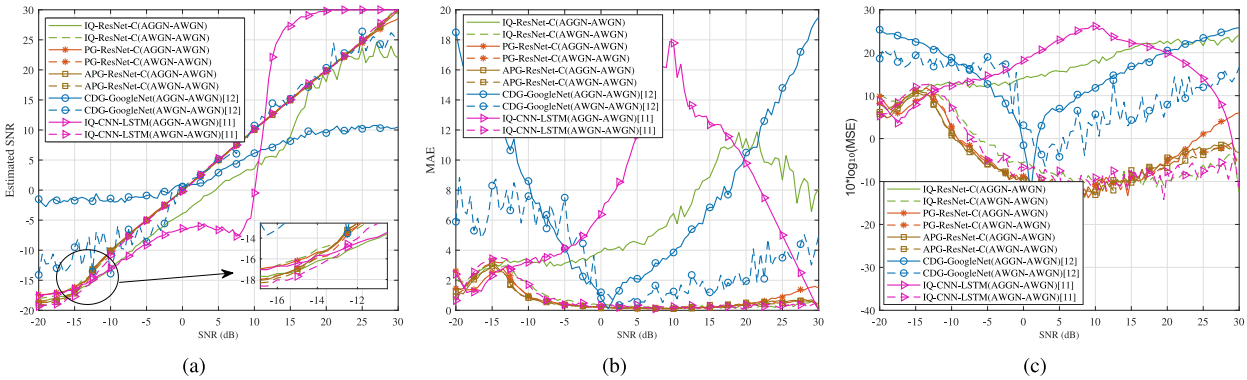

FIGURE 11. Performance under AWGN noise with AWGN/AGGN-trained models. (a) Mean SNR, (b) MAE, and (c) MSE.

Obviously, we can see from Fig. 10 that for our proposed classification-based method, the model trained under the AWGN has good generalization, and the performance of this model used in the prediction of SNR under the AGGN is close to that of retrained AGGN-AGGN no matter what input format is used. However, things are quite different for existing methods CDG-GoogleNet and IQ-CNN-LSTM. The estimation error of IQ-CNN-LSTM increases in the range of $[-20, -10]$ dB and $[20, 30]$ dB, and the estimation error of CDG-GoogleNet becomes larger in the range of $[-20, 0]$ dB and $[20, 30]$ dB. This indicates that our proposed methods perform far better than existing methods for transferring models trained under AWGN to AGGN.

For AGGN trained models, the situation is not the same. As shown from Fig. 11, we can clearly find that the AGGN-trained model using IQ input has a high error when used for prediction under the AWGN noise even though it works well in AGGN. PG and APG overcome this defect, and their performance is close to that of the retrained models. For IQ-CNN-LSTM and CDG-GoogleNet, they both have poor estimation performance when transferring from AGGN to AWGN in the whole SNR range. Furthermore, even when using AGGN trained models to test SNR estimation under AGGN noise, the performance of the CDG-GoogleNet method is still poor. It can infer that our proposed PG-ResNet-C and APG-ResNet-C are superior to existing methods in this case.

2) PERFORMANCE IN COLORED NOISE

The ideal white noise possesses infinite bandwidth, resulting in infinite energy, a condition unattainable in the real world. In this experiment, we utilize pink noise, a type of colored noise characterized by a PSD inversely proportional to frequency. Fig. 12 illustrates the estimation performance of these SNR estimation methods based on deep learning under pink noise. In Fig. 12(a), we analyze the experimental results of the average SNR. The SNR estimation methods proposed in this paper with IQ input and PG input demonstrate superior performance in estimating signals under pink noise. In the high SNR region of $[15, 30]$ dB, the estimation performance of IQ-CNN-LSTM and APG-ResNet-C is compromised. IQ-CNN-LSTM tends to overestimate the SNR of the signal, while APG-ResNet-C tends to underestimate it. In the low SNR region of $[-20, 5]$ dB, CDG-GoogleNet fails to accurately estimate the SNR of the signal. This indicates that IQ-ResNet-C and PG-ResNet-C methods exhibit better performance under colored noise.

F. PERFORMANCE UNDER MULTIPATH CHANNELS

In this experiment, we assess the performance of both our proposed methods and existing methods in multipath channels, encompassing Rician and Rayleigh channels. The results are presented in Fig. 13 and Fig. 14. Based on the experimental results of the estimated average SNR, as illustrated in Fig. 13(a) and Fig. 14(a), it is evident that

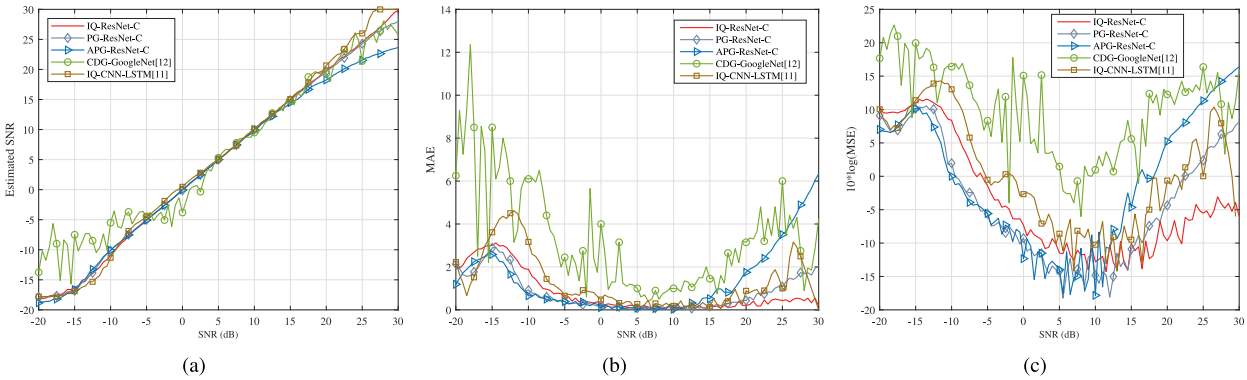

FIGURE 12. Performance under pink noise. (a) Mean SNR, (b) MAE, and (c) MSE.

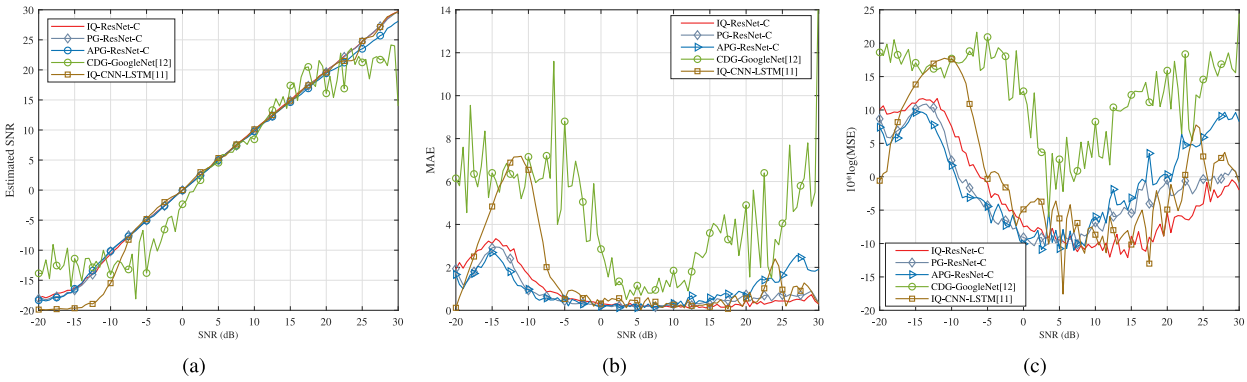

FIGURE 13. Performance comparison of different methods in Rician channel. (a) Mean SNR, (b) MAE, and (c) MSE.

in both the Rician channel and the Rayleigh channel, the estimation performance of CDG-GoogleNet is worst among these methods in both the low SNR region of $[-20, 5]$ dB and the high SNR region of $[15, 30]$ dB. Furthermore, in the Rician channel, the estimation performance of IQ-CNN-LSTM is notably reduced within the range of $[-20, -7]$ dB. In the Rayleigh channel, the estimation performance of IQ-CNN-LSTM deteriorates in the high SNR region of $[15, 30]$ dB. The three SNR estimation methods introduced in this paper, namely IQ-ResNet-C, PG-ResNet-C, and APG-ResNet-C, exhibit robust performance in both Rayleigh and Rician channels. From Fig. 13(b), Fig. 13(c), Fig. 14(b), and Fig. 14(c), it can be observed that, except for a slight increase in the error of APG-ResNet-C within the SNR range of $[20, 30]$ dB, the estimation errors of our methods are significantly lower than those of existing methods. This further highlights the superiority of our proposed approaches.

G. ADAPTABILITY TO NEW SIGNALS

In order to verify the generalization of the method, we carry out experiments where we use the model trained with QPSK signals to estimate the SNR of BPSK signals. The retraining method which uses the BPSK signals to train the model and use the trained model to estimate the SNR of BPSK signals is adopted for comparison. The results are shown in Fig. 15, from which we can find that the proposed method has surprisingly good adaptability to new signals even without retraining. The estimation error of the model under the new modulation is small and the SNR can be predicted well. Note that we also use the trained model with QPSK signals to estimate the SNR of 16QAM and 4PAM signals and similar performance has been observed. We do not repeat these results here for the sake of simplicity.

H. PERFORMANCE OF REGRESSION-BASED METHOD

SNR estimation is essentially a continuous spatial parameter estimation problem. In this paper, we propose both classification and regressed based deep learning methods to solve the problem. We now compare these two solutions considering both on-grid (the tested SNR is the same with the training SNR) and off-grid (the tested SNR is not in the range of the training SNRs) scenarios. For simplicity, the methods based on regression with IQ input, PG input and APG input are denoted as IQ-ResNet-R, PG-ResNet-R and APG-ResNet-R respectively in the rest of the paper. The experimental results with on-grid scenario is shown in Fig. 16. QPSK and AWGN are used. The SNR interval is 0.5 dB. From Fig. 16, the MSE of the regression-based method is significantly smaller than that of the classification-based method in the low SNR region of $[-18, -12]$ dB. Close performance of the two methods is observed in the medium and high SNR region.

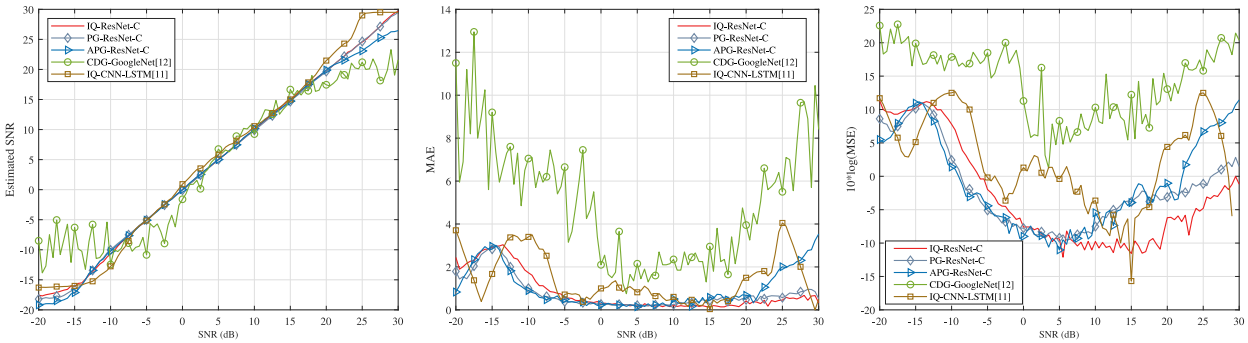

FIGURE 14. Performance comparison of different methods in Rayleigh channel. (a) Mean SNR, (b) MAE, and (c) MSE.

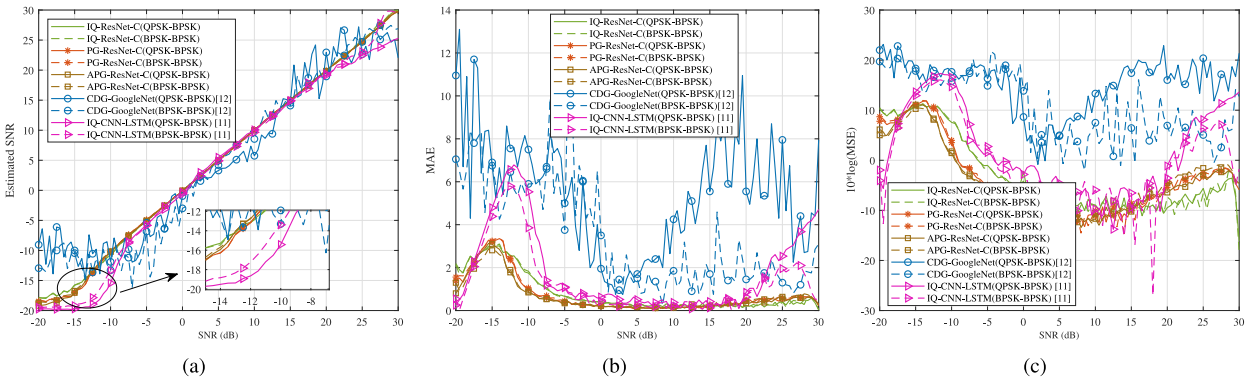

FIGURE 15. Performance in estimating SNR of new signals. QPSK-BPSK means using QPSK-trained models to estimate SNR of BPSK signal. BPSK-BPSK means both the training and testing signals are BPSK. (a) Mean SNR, (b) MAE, and (c) MSE.

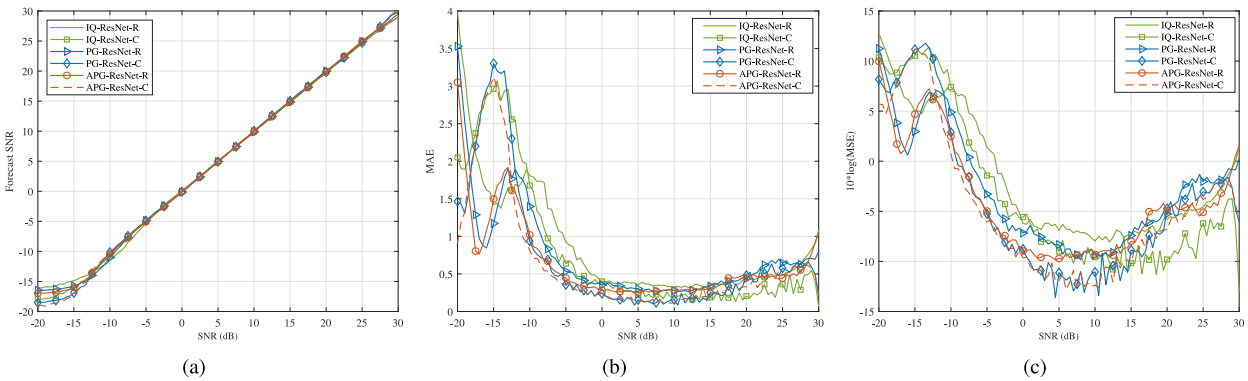

FIGURE 16. Performance of both classification and regression-based methods under on-grid scenario. (a) Mean SNR, (b) MAE, and (c) MSE.

We next consider the off-grid scenario where the training SNR is in the range of $[-20, 30]$ dB with interval 2 dB while the testing SNR is in the same range but with interval 0.5 dB. In this case, there are a lot of SNRs to be estimated that are not within the training set. Results are shown in Fig. 17 and Fig. 18. It is found that when the training SNR interval becomes larger, continuing to use the classification-based method will lead to large prediction error due to the insufficient coverage of the training SNR. We can see in Fig. 17(b) and Fig. 17(c) that the prediction curve is very "sharp" for the off-grid SNRs. This problem can be effectively solved by using regression-based method. Comparing Fig. 18(b) with Fig. 17(b) and Fig. 18(c) with Fig. 17(c), it is obvious that the error curves of the regression-based method are smooth, indicating that it still has a good prediction effect on the SNRs that have "never seen".

I. PERFORMANCE IN FEW-SHOT SCENARIO

In this experiment, we assess the performance of the methods in a few-shot scenario, meaning that we utilize a small number of samples for training. Initially, there are 500 samples for each SNR. During the experiment, we randomly select 2 percent, 10 percent, and 20 percent of

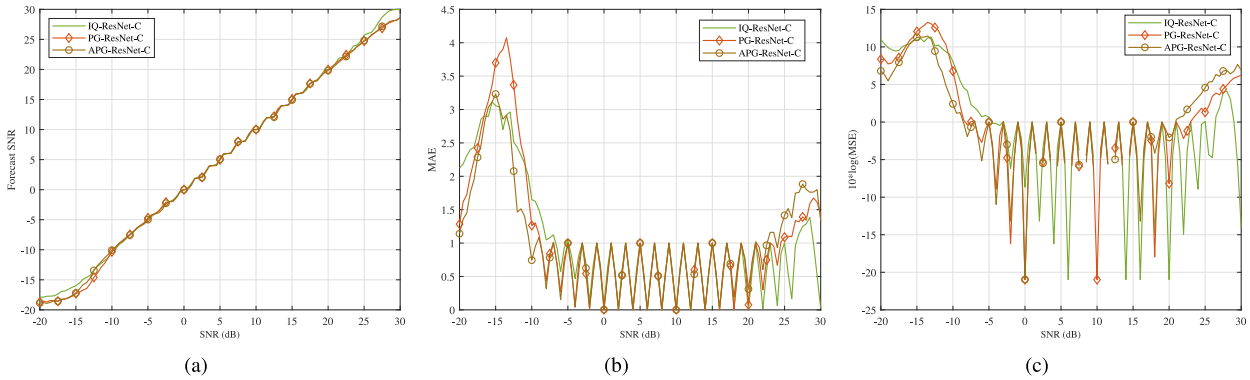

FIGURE 17. Performance of classification-based method under off-grid scenario. (a) Mean SNR, (b) MAE, and (c) MSE.

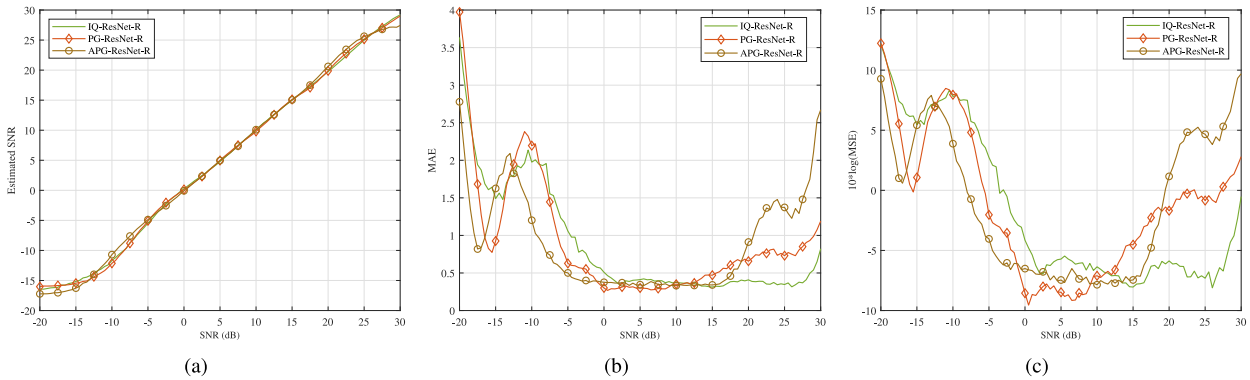

FIGURE 18. Performance of regression-based method under off-grid scenario. (a) Mean SNR, (b) MAE, and (c) MSE.

these samples, equivalent to 10 samples, 50 samples, and 100 samples for each SNR. The experimental results are depicted in Fig. 19.

The five subgraphs illustrate the average SNR estimated by five deep learning-based methods. Upon examination of the individual curves within each subgraph, it becomes apparent that the estimation performance when using IQ data as the network input is significantly influenced by the number of samples. For instance, from Fig. 19(a) and Fig. 19(d), it is evident that both IQ-ResNet-C and IQ-CNN-LSTM exhibit poor estimation performance when trained with only 10 samples for each SNR. The experimental results in Fig. 19(e) reveal that, in the few-shot scenario, even though the performance of the CDG-GoogleNet method is not affected by small samples, it is significantly worse compared to other deep learning-based SNR estimation methods. It fails to accurately estimate SNR within the ranges of $[-20, -5]$ dB and $[15, 30]$ dB.

The observation from Fig. 19(b) and Fig. 19(c) indicates that PG-ResNet-C and APG-ResNet-C exhibit commendable estimation performance in the context of small-sample learning. The errors are significant only within the range of $[-20, -5]$ dB, while accurate SNR estimation is achieved in the remaining range. This validates the outstanding performance of the methods with power spectrum input proposed in this paper.

J. COMPLEXITY ANALYSIS

In this section, we conduct a comprehensive analysis of the complexity of the methods, taking into consideration different input formats. The analysis includes an assessment of both time complexity and space complexity. FLOPs are related to time complexity and the number of Params are related to space complexity. The inputs for PG and APG in this paper incorporate the fast Fourier transform (FFT). In comparison to the discrete Fourier transform, the FFT significantly reduces computational complexity. The time complexity of FFT, as stated in [33], is as follows:

$$\mathcal{D}_{\text{FFT}} \sim O\big(N \times \log_2 N\big), \tag{39}$$

where N is the length of the received signal. When using FFT, the number of complex multiplications is $(N/2) \times \log_2 N$, and the number of complex addition is $N \times \log_2 N$. When using the CDG method to convert the signal into a constellation diagram, its time complexity is

$$\mathcal{D}_{\text{CDG}} \sim O(N). \tag{40}$$

As for CNNs, the FLOPs calculation of the convolution layer of the network [34] is as follows:

$$\text{Conv}_{\text{FLOPs}} = 2 \cdot size_{out}(C_{in} \cdot size_{kernel} + 1)C_{out}, \tag{41}$$

where $size_{out}$ is the size of the output feature map of each convolution kernel, $size_{kernel}$ is the size of each

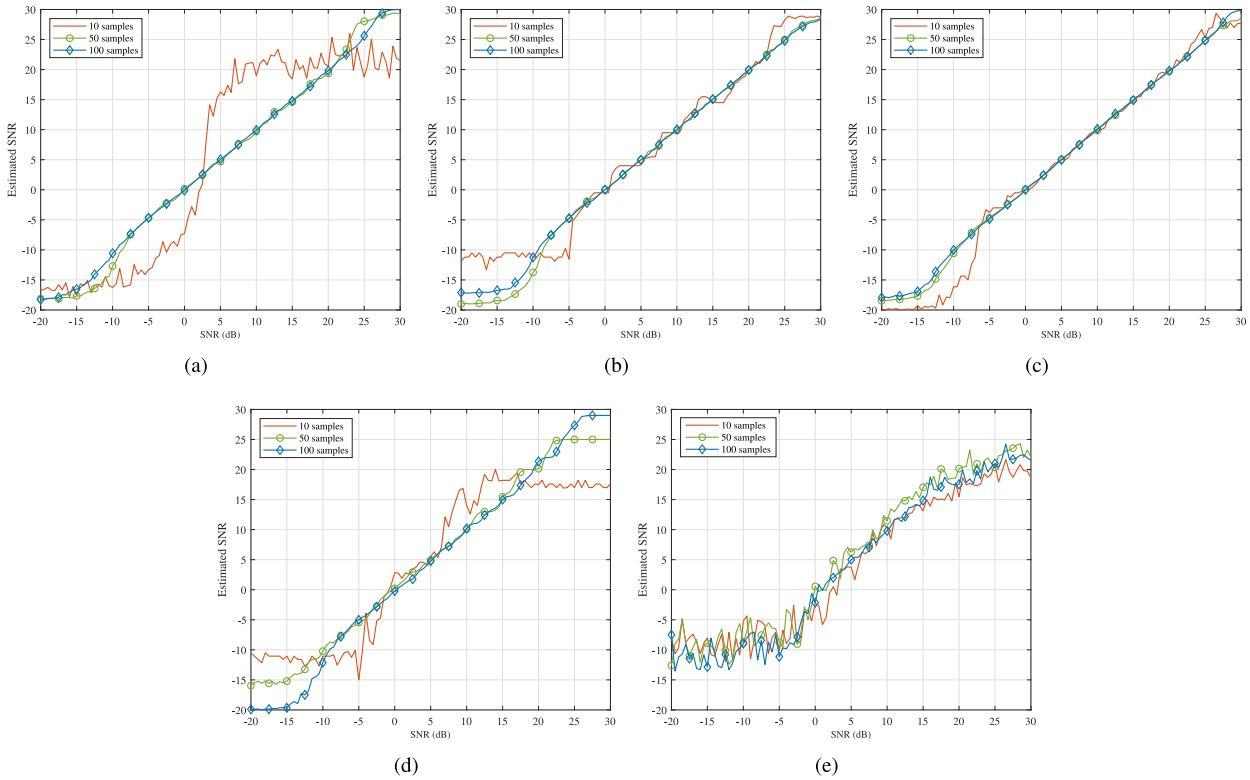

FIGURE 19. Performance in Few-shot learning. (a) IQ-ResNet-C, (b) PG-ResNet-C, (c) APG-ResNet-C, (d) IQ-CNN-LSTM and (e) CDG-GoogleNet.

convolution kernel, C_{in} is the number of channels of the input, C_{out} is the number of convolution kernels, that is, the number of output channels. The computational complexity of the batch normalization layer and the ReLU layer are both [10]

$$\mathcal{D}_{\text{ReLU}} \sim O(size_{\text{in}}), \tag{42}$$

where $size_{\text{in}}$ represents the size of the input feature map. The computational complexity of the pooling layer is [10]

$$\mathcal{D}_{\text{Pooling}} \sim O\big(size_{\text{in}} \times F_l / D_l\big), \tag{43}$$

where F_l is the size of the pooling filter and D_l is the down-sampling factor.

The FLOPs [34] calculation of the fully connected layer is as follows:

$$\text{FC}_{\text{FLOPs}} = (2T - 1)H, \tag{44}$$

where T is the number of neurons in the input layer, and H is the number of neurons in the output layer.

We can obtain the FLOPs which includes the number of pre-processing calculations and Params values of our proposed method as shown in Table 8. The complexity of IQ-CNN-LSTM [11], CDG-GoogleNet [12], IQ-CNN [6] and IQ-Transformer [32] is also provided for comparison. The results indicate that the computational complexity of PG-ResNet-C is marginally lower than that of IQ-ResNet-C. The APG-ResNet-C reduces the computational complexity to nearly a quarter of that of IQ-ResNet-C as we choose $L = 4$ to compute the APG-ResNet-C, demonstrating its superiority. Comparing IQ-CNN, IQ-Transformer, IQ-CNN-LSTM and IQ-ResNet-C, we can find that the FLOPs of IQ-ResNet-C is near that of IQ-Transformer. Furthermore, among these four methods, we can observe that the complexity of the IQ-CNN method is the highest, being one order of magnitude higher than the other three methods. As for Params, the models with PG-ResNet-C and APG-ResNet-C input have the same amount of parameters, both slightly smaller than the number of parameters of the model IQ-ResNet-C. The number of parameters of IQ-CNN-LSTM is nearly twenty times of our proposed methods. The number of parameters of both the IQ-Transformer and IQ-CNN methods is higher than our proposed methods. Both the computational complexity and the number of parameters of CDG-GoogleNet are far more than the other six methods. These results further demonstrate the superiority of our proposed methods in terms of complexity.

VII. CONCLUSION

We have proposed an SNR estimation framework based on deep learning classification. Power spectrum inputs, i.e., PG and APG, have been introduced to reduce the computational complexity. We have also proposed an SNR estimation method based on deep learning regression to overcome

the inevitable estimation error problem of classification-based methods in dealing with signals with SNR not within the training label set. We have conducted a large number of simulation experiments considering various scenarios. The experimental results demonstrate that the proposed classification-based methods outperform in terms of average SNR and MSE compared to two existing deep learning-based SNR estimation methods, i.e., CDG-GoogleNet and IQ-CNN-LSTM. In addition, the proposed regression-based method has better estimation performance for untrained SNR compared to the classification-based method. Furthermore, in the cases of transferring to another noise distribution and few-shot scenario, the proposed methods with PG and APG input have better adaptability than the method based on IQ input. Finally, the proposed methods only need to be trained under one modulation signals to adapt to SNR estimation of other modulation signals, with superior transfer performance. Complexity analysis has shown that both time complexity and space complexity of our proposed method with APG input are much smaller than those of CDG-GoogleNet and IQ-CNN. In summary, our proposed methods provide more accurate, robust, adaptable, and efficient solutions than the existing representative SNR estimation methods. As part of our future research endeavors, we plan to explore hardware implementation and conduct over-the-air experiments in order to thoroughly assess the performance of the proposed methods. We will also analyze the security of the proposed method under adversarial attacks to improve the reliability of the algorithms in non-cooperative environments.

REFERENCES

[1] V. Meghdadi. "BER calculation." Jan. 2008. [Online]. Available: http://perso.ensil.unilim.fr/meghdadi/notes/ber_awgn.pdf

[2] H. Ye, G. Y. Li, and B.-H. Juang, "Power of deep learning for channel estimation and signal detection in OFDM systems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 1, pp. 114–117, Feb. 2018.

[3] T. Gruber, S. Cammerer, J. Hoydis, and S. T. Brink, "On deep learning-based channel decoding," in *Proc. 51st Annu. Conf. Inf. Sci. Syst. (CISS)*, 2017, pp. 1–6.

[4] H. Abeida, T. Y. Al-Nafouri, and S. Al-Ghadhban, "Data-aided SNR estimation in time-variant Rayleigh fading channels," in *Proc. IEEE 11th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, 2010, pp. 1–5.

[5] R. Matzner and F. Englberger, "An SNR estimation algorithm using fourth-order moments," in *Proc. IEEE Int. Symp. Inf. Theory*, 1994, p. 119.

[6] K. Yang, Z. Huang, X. Wang, and F. Wang, "An SNR estimation technique based on deep learning," *Electronics*, vol. 8, no. 10, p. 1139, 2019.

[7] A. Jagannath, J. Jagannath, and T. Melodia, "Redefining wireless communication for 6G: Signal processing meets deep learning with deep unfolding," *IEEE Trans. Artif. Intell.*, vol. 2, no. 6, pp. 528–536, Dec. 2021.

[8] S. Zheng, S. Chen, P. Qi, H. Zhou, and X. Yang, "Spectrum sensing based on deep learning classification for cognitive radios," *China Commun.*, vol. 17, no. 2, pp. 138–148, Feb. 2020.

[9] T. Chen et al., "EMD and VMD empowered deep learning for radio modulation recognition," *IEEE Trans. Cogn. Commun. Netw.*, vol. 9, no. 1, pp. 43–57, Feb. 2023.

[10] S. Zheng, S. Chen, and X. Yang, "DeepReceiver: A deep learning-based intelligent receiver for wireless communications in the physical layer," *IEEE Trans. Cogn. Commun. Netw.*, vol. 7, no. 1, pp. 5–20, Mar. 2021.

[11] T. Ngo, B. Kelley, and P. Rad, "Deep learning based prediction of signal-to-noise ratio (SNR) for LTE and 5G systems," in *Proc. 8th Int. Conf. Wireless Netw. Mobile Commun. (WINCOM)*, 2020, pp. 1–6.

[12] X. Xie, S. Peng, and X. Yang, "Deep learning-based signal-to-noise ratio estimation using constellation diagrams," *Mobile Inf. Syst.*, vol. 2020, pp. 1–9, Nov. 2020.

[13] D. Guo, Y. Wu, S. S. Shitz, and S. Verdú, "Estimation in Gaussian noise: Properties of the minimum mean-square error," *IEEE Trans. Inf. Theory*, vol. 57, no. 4, pp. 2371–2385, Apr. 2011.

[14] S. K. Tiwari and P. K. Upadhyay, "Maximum likelihood estimation of SNR for diffusion-based molecular communication," *IEEE Wireless Commun. Lett.*, vol. 5, no. 3, pp. 320–323, Jun. 2016.

[15] D. Athanasios and G. Kalivas, "SNR estimation for low bit rate OFDM systems in AWGN channel," in *Proc. Int. Conf. Netw., Int. Conf. Syst. Int. Conf. Mobile Commun. Learn. Technol.*, 2006, pp. 198–198.

[16] R. Scholtz, "Review of 'detection, estimation, and modulation theory, part I' (van trees, H.; 1968)," *IEEE Trans. Inf. Theory*, vol. 14, no. 4, pp. 612–613, Jul. 1968.

[17] R. Gagliardi and C. Thomas, "PCM data reliability monitoring through estimation of signal-to-noise ratio," *IEEE Trans. Commun. Technol.*, vol. 16, no. 3, pp. 479–486, Jun. 1968.

[18] D. Pauluzzi and N. Beaulieu, "A comparison of SNR estimation techniques for the AWGN channel," *IEEE Trans. Commun.*, vol. 48, no. 10, pp. 1681–1691, Oct. 2000.

[19] B. Shah and S. Hinedi, "The split symbol moments SNR estimator in narrow-band channels," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 26, no. 5, pp. 737–747, Sep. 1990.

[20] T. Salman, A. Badawy, T. M. Elfouly, T. Khattab, and A. Mohamed, "Non-data-aided SNR estimation for QPSK modulation in AWGN channel," in *Proc. IEEE 10th Int. Conf. Wireless Mobile Comput., Netw. Commun. (WiMob)*, 2014, pp. 611–616.

[21] T. Benedict and T. Soong, "The joint estimation of signal and noise from the sum envelope," *IEEE Trans. Inf. Theory*, vol. 13, no. 3, pp. 447–454, Jul. 1967.

[22] R. Matzner, "An SNR estimation algorithm for complex baseband signals using higher-order statistics," *Facta Univ.*, vol. 6, no. 1, pp. 41–52, 1993.

[23] M. Simon and A. Mileant, "SNR estimation for the baseband assembly," Telecommun. Data Acquis., NASA, Washington, DC, USA, Rep. 42-85, 1986.

[24] A. Brandao, L. Lopes, and D. McLemon, "In-service monitoring of multipath delay and cochannel interference for indoor mobile communication systems," in *Proc. Int. Conf. Commun.*, vol. 3, 1994, pp. 1458–1462.

[25] L. Huang, S. Bi, and Y.-J. A. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," *IEEE Trans. Mobile Comput.*, vol. 19, no. 11, pp. 2581–2593, Nov. 2020.

[26] J. Sijbers, A. den Dekker, P. Scheunders, and D. Van Dyck, "Maximum-likelihood estimation of Rician distribution parameters," *IEEE Trans. Med. Imag.*, vol. 17, no. 3, pp. 357–361, Jun. 1998.

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, 2012, pp. 1–9.

[28] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[29] C. Szegedy et al., "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1–9.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 770–778.

[31] S. Ruder, "An overview of gradient descent optimization algorithms," 2016, *arXiv:1609.04747*.

[32] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.

[33] P. Welch, "The use of fast fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms," *IEEE Trans. Audio Electroacoust.*, vol. 15, no. 2, pp. 70–73, Jun. 1967.

[34] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," 2016, *arXiv:1611.06440*.

SHILIAN ZHENG received the B.S. degree in telecommunication engineering and the M.S. degree in signal and information processing from Hangzhou Dianzi University, Hangzhou, China, in 2005 and 2008, respectively, and the Ph.D. degree in communication and information system from Xidian University, Xi'an, China, in 2014. He is currently a Researcher with the National Key Laboratory of Electromagentic Space Security, Jiaxing, China. His research interests include cognitive radio, deep learning-based radio signal processing, and electromagnetic space security.

SHURUN CHEN was born in Wenzhou, Zhejiang, China, in 1998. He received the B.S. degree from the Zhejiang University of Science and Technology in 2021, and the M.S. degree from Hangzhou Dianzi University, Hangzhou, China, in 2024. His research interests include signal-to-noise ratio estimation and deep learning.

TAO CHEN received the B.S. degree from Quzhou University, Quzhou, China, in 2020, the M.S. degree from the Zhejiang University of Technology, Hangzhou, China, in 2023, where he is currently pursuing the Ph.D. degree. His current research interests include radio signal recognition and deep learning.

ZHUANG YANG was born in Heze, Shandong, China, in 1999. He received the B.S. degree from the North China University of Science and Technology, Langfang, China, in 2020, and the master's degree from Hangzhou Dianzi University in 2024. His current research interests include direction-of arrival estimation and deep learning.

ZHIJIN ZHAO received the M.S. and Ph.D. degrees from Xidian University, Xi'an, China, in 1984 and 2009, respectively. In 1993 and 2003, as a Visiting Scholar, she studied adaptive signal processing and blind signal processing with the Darmstadta University of Technology and also with the University of Erlangen-Nuremberg, respectively. She is currently a Professor with the School of Communication engineering, Hangzhou Dianzi University, Hangzhou, China. Her research interests include communication signal processing, cognitive radio technology, intelligent signal processing, and other aspects of research. She once served as the President of the College of Communication Engineering, Hangzhou Dianzi University, as well as a Senior Member of China Electronics Society, a member of National Signal Processing Society, a Secretary General of Zhejiang Signal Processing Society, and the Director of Hangzhou Electronics Society.

XIAONIU YANG is currently a Chief Scientist with the National Key Laboratory of Electromagentic Space Security, Jiaxing, China. He published the first software radio book *Software Radio Principles and Applications* [Publishing House of Electronics Industry, 2001 (in Chinese)] (X. Yang, C. Lou, and J. Xu) in China. He holds more than 40 patents. His current research interests are software-defined satellite, big data for radio signals, and deep learning based signal processing. He is also an Academician of the Chinese Academy of Engineering and a Fellow of the Chinese Institute of Electronics.