drones

MDPI

Article

Self-Supervised Representation Learning for Quasi-Simultaneous Arrival Signal Identification Based on Reconnaissance Drones

Linqing Guo, Mingyang Du, Jingwei Xiong, Zilong Wu and Jifei Pan *

College of Electronic Engineering, National University of Defense Technology, Hefei 230031, China; guolq@nudt.edu.cn (L.G.)
* Correspondence: panjifei17@nudt.edu.cn

Abstract: Reconnaissance unmanned aerial vehicles are specifically designed to estimate parameters and process intercepted signals for the purpose of identifying and locating radars. However, distinguishing quasi-simultaneous arrival signals (QSAS) has become increasingly challenging in complex electromagnetic environments. In order to address the problem, a framework for self-supervised deep representation learning is proposed. The framework consists of two phases: (1) pre-train an autoencoder. For learning the unlabeled QSAS representation, the ConvNeXt V2 is trained to extract features from masked time–frequency images and reconstruct the corresponding signal in both time and frequency domains; (2) transfer the learned knowledge. For downstream tasks, encoder layers are frozen, the linear layer is fine-tuned to classify QSAS under few-shot conditions. Experimental results demonstrate that the proposed algorithm can achieve an average recognition accuracy of over 81% with the signal-to-noise ratio in the range of −16∼16 dB. Compared to existing CNN-based and Transformer-based neural networks, the proposed algorithm shortens the time of testing by about 11× and improves accuracy by up to 21.95%.

Keywords: unmanned aerial vehicles; quasi-simultaneous arrival signal; self-supervised representation learning; transfer learning

Citation: Guo, L.; Du, M.; Xiong, J.; Wu, Z.; Pan J. Self-Supervised Representation Learning for Quasi-Simultaneous Arrival Signal Identification Based on Reconnaissance Drones. *Drones* **2023**, 7, 475. https://doi.org/10.3390/drones7070475

Academic Editor: Diego González-Aguilera

Received: 18 June 2023
Revised: 16 July 2023
Accepted: 17 July 2023
Published: 19 July 2023

1. Introduction

Due to the flexible usage of drones or unmanned aerial vehicles (UAV), they are generally not restricted by terrain or climate conditions, making them ideal for carrying high-performance equipment for reconnaissance purposes [1–3]. Reconnaissance UAV intercept signals transmitted by radars [4]. These signals are then amplified and processed to identify their features. However, radars transmit signals with different modulations within a short period of time, causing the UAV to intercept signals of pulse fronts to not arrive exactly simultaneously. Identifying quasi-simultaneous arrival signsal (QSAS) poses major challenges.

Most existing technologies in radar signal identification focus on intra-pulse modulation recognition. Currently, there are two main approaches for the identification of intra-pulse modulation types: one is based on manual feature extraction combined with machine learning (ML) methods, and the other is based on deep learning (DL) methods. The first approach involves extracting handcrafted features from the raw radar data and using ML, such as support vector machines (SVM), k-nearest neighbors (KNN) [5–7]. However, this approach heavily relies on the domain expertise and feature engineering, which can be time-consuming and may not capture all the relevant information in the data. The second approach automatically learns hierarchical representations from the raw radar data by using DL, such as deep residual network (ResNet) and visual geometry group (VGG) [8–10]. DL can effectively capture the complex patterns and variations in radar signals, leading to improved performance compared to traditional ML.

However, these method treat QSAS as a single pulse for signal sorting and recognition, resulting in the loss of valuable multi-pulse information. Furthermore, the presence of multiple labels for QSAS further increases the difficulty of the labeling task, and obtaining data labels is often difficult [11], making it challenging to obtain large amounts of labeled data for supervised training [12]. Therefore, there is growing interest in using only unlabeled signals for self-supervised learning (SSL) of QSAS.

SSL pre-training can overcome the limitations of supervised learning (SL) by enabling the learning of generic features representation from unlabeled data, which can be transferred to downstream tasks with few-shot learning. Various SSL methods such as self-distillation [13], contrast learning [14], and masked autoencoders (MAE) [15] have even outperformed SL [16]. For instance, unlike Vision Transformers (ViT) [17], the powerful MAE [18] does not necessarily require a good recipe with strong regularization. The new SSL called ConvNeXt V2 [19] is proposed in 2023, which has benefited from better initialization. To leverage the advantages of convolutional neural networks (CNN) fully, we extend the ConvNeXt V2 scheme to QSAS identification.

ConvNeXt V2 focuses on the neural network architecture and the training method, but the quality of training data also plays a significant role in overall model performance [20–24]. While ConvNeXt V2 can achieve good results through tuning parameters, it's important to consider the impact of the dataset. In this case, we aim to apply a standard ConvNeXt V2 to QSAS identification with minimal modifications. To construct the dataset, we use a label powerset [25–29] and short-time Fourier transform (STFT) [30–34] on multi-label radar signals. However, QSAS is not simply a combination of different modulations and contains a lot of noise. Therefore, it is crucial to explore how to train the network and perform transfer learning (TL) [35,36] to adapt to different signal-to-noise ratios (SNR) [37,38] and improve noise adaptation.

Figure 1 shows the conceptual representation of the SSL of QSAS identification, where the first phase (the upper part of the figure) is SSL for pre-training and the second phase (the lower part of the figure) is the few-shot of SL for fine-tuning based on the features learned from pre-training. Since observing QSAS in the time domain is difficult, STFT is applied to observe the existence of signal overlap in the time-frequency more clearly (the left part of the figure). In order to learn features in both time and frequency domains, instead of just predicting the amplitudes at different times, we randomly mask patches in both time-frequency domains and then use a sparse convolution-based [39] ConvNeXt V2 to extract features from the unmasked patches in SSL. Then, we reconstruct the masked signal using only a simple ConvNeXt V2 block that learns inter-contextual correlations in time-frequency domains. The encoder of the pre-trained model is transferred for the QSAS identification task. We add the linear layer after the encoder network and use a few-shot learning to fine-tune the model for QSAS identification.

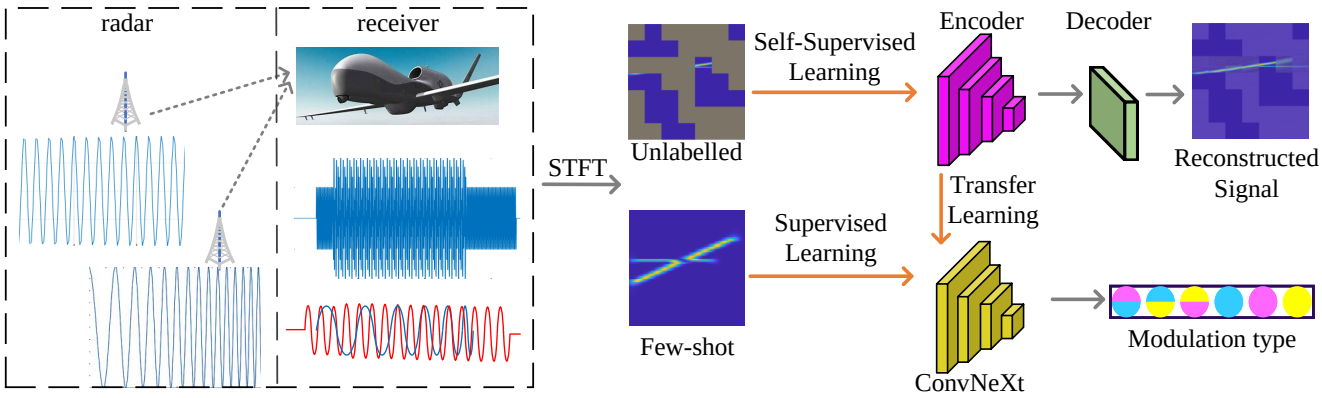

Figure 1. A conceptual representation of the SSL of QSAS identification.

Our main contributions can be summarized as follows:

1. We adopt an autoencoder called ConvNeXt V2 to identify QSAS in two-dimensional time-frequency images. The overlapping QSAS in time domain can be distinguished effectively by STFT which significantly improves the accuracy of the model. This model consists of a complex encoder and a simple decoder. To improve its performance, we pre-train it in an unsupervised manner as a generative model on massive time-frequency images with a 60% mask ratio. Then, we fine-tune the encoder for few-shot QSAS identification performance.
2. We conduct thorough experiments on our self-built QSAS dataset, which includes six kinds of combinations of continuous wave (CW), linear frequency modulation (LFM) and binary phase shift keying (BPSK). These combinations obey the rules that the arriving time delay is within one pulse width (PW), and the frequency is close. We perform a very detailed analysis of various parameters and conduct ablation experiments on different aspects such as spatial masking strategy, decoder setting, relationship between patchsize and mask ratio and the number of pre-training epochs. The simulation results demonstrate that our method achieves high identification accuracy and exhibits more robustness in a low SNR environment.

The structure of this paper is as follows: In Section 2, we analyze the model and formula representation of the QSAS dataset. In Section 3, we introduce the architecture of the SSL model. In Section 4, we analyze the results of the experiments. In Section 5, we conclude our work.

2. QSAS Dataset

2.1. Signal Model

We assume that the raw radar signal received by the reconnaissance receiver on UAV consists of two parts: signal and noise [40], and the formula is expressed as

$$x(t) = s(t) + n(t), \tag{1}$$

where $s(t)$ is the received effective signal, commonly uses non-modulation of $CW(t)$, or $LFM(t)$ [41], or $BPSK(t)$ [42], $n(t)$ is multi-environmental noise [43].

To be specific, the CW form is relatively simple, using a single carrier frequency without intra-pulse modulation. The signal model is as follows:

$$CW(t) = A\sin(2\pi f_c t), \tag{2}$$

where A is signal amplitude, f_c is intermediate frequency (IF), $t = 0 : \frac{1}{f_s} : PW - \frac{1}{f_s}$, where f_s is sampling frequency.

The frequency of the LFM signal changes linearly with time, and the signal model is

$$LFM(t) = A\sin\left[2\pi\left(f_c - \frac{B}{2}\right)t + k\pi t^2\right], \tag{3}$$

where $k = \frac{B}{PW}$ is the frequency modulation slope of LFM, and B is the signal bandwidth, namely the frequency variation range.

The phase of $BPSK$ changes with the reversal of the encoding mode, and the signal model is

$$BPSK(t) = A\sin(2\pi f_c t + \phi_m(t)), \quad \phi_m(t) \in \{0, \pi\}, \tag{4}$$

where $\phi_m(t)$ is the 13-bit Barker code that determines the phase value to be either 0 or π.

We assume that the white noise signal and colored noise can be found in various electromagnetic environments and conform to Gaussian distribution of $N(\mu, \sigma^2)$. The power spectral density (PSD) of white noise is constant across an infinite frequency range of $-\infty\sim+\infty$ [44], so the probability density function is

$$p(t) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{(t-\mu)^2}{2\sigma^2}\right]. \tag{5}$$

The amplitude of colored noise is correlated at each moment [45], so the PSD is the Fourier transform of the autocorrelation function $R(p)$:

$$P(\omega) = \sum_{p \in \mathbb{Z}} R(p) \exp(-j\omega p). \tag{6}$$

The SNR is the ratio of the power of the signal σ_s^2 to the power of the noise σ_n^2:

$$SNR = 10 \cdot \lg\left(\sigma_s^2 / \sigma_n^2\right). \tag{7}$$

UAVs usually regard the raw signal as a single modulation for subsequent processing, e.g., $s(t) = \{\mathrm{CW}(t), \mathrm{LFM}(t), \mathrm{BPSK}(t)\}$, but the received signal often has the condition of various modulations, e.g., $s(t) = \{\mathrm{CW}(t) + \mathrm{LFM}(t+\tau), \mathrm{CW}(t) + \mathrm{BPSK}(t+\tau), \mathrm{LFM}(t) + \mathrm{BPSK}(t+\tau)\}$, where τ represents the delay of signal arrival time.

Definition of QSAS: In the electromagnetic environment, numerous radars transmit massive signals. In the same PW, there are various signals that arrival with a delay of $\tau (\tau \leq PW)$, and have similar frequency. We call such signals with overlapping information in time-frequency domains QSAS. The presence of QSAS makes it difficult to carry out accurate signal sorting, signal recognition and other subsequent work.

It is difficult for raw signals to directly observe the aliasing state in data. In such cases, time-frequency transform can be employed to estimate the joint distribution of signal in time-frequency domains. This representation helps to reveal the relationship between the time domain and frequency domain [46]. One commonly used time-frequency transform is STFT. STFT adds a short-time sliding window to the signal processing process to perform Fourier transform, which allows for the reflection of time-frequency information of signals. Moreover, the time-frequency visualization can be easily realized by the spectrogram function in Matlab. Therefore, the STFT is used to calculate the time-frequency spectrum:

$$STFT(t, f) = \int_{-\infty}^{+\infty} s(\tau) h^*(\tau - t) \exp(-j2\pi f \tau) d\tau, \tag{8}$$

where $s(\tau)$ represents the one-dimensional time domain signal, t and f represent the time and the frequency, respectively. $h^*(\tau - t)$ represents the window function.

2.2. Label Powerset

The construction of the QSAS dataset involves three main steps: original signal simulation, data augmentation and data annotation. The original signal is simulated according to the signal model described in the previous subsection, and we use simple augmentation methods such as image scaling. Data annotation is the key to the difference between QSAS and intra-pulse signals of a single modulation.

One of the challenges with QSAS is that it contains various modulations, and their labels are parallel without any connection between them. To address this, the multi-label classification (MLC) problem is transformed into a single label classification (SLC) problem using the Label Powerset method to facilitate the use of the subsequent model. This method treats each label combination as a unique category and converts a multi-label dataset into a single-label multi-category dataset.

As shown in Table 1, if we have six samples named CWBPSK, CWLFM, dBPSK, dCW, dLFM and LFMBPSK, and each sample has three labels. The Label Powerset method assigns new labels to each sample according to the original multi-label value of the sample. This form a new label set {0, 1, 2, 3, 4, 5}. Then, it only needs to train a SLC to predict the label combination of each instance. Moreover, this method is extensible and can handle any number of labels.

Table 1. Multi-label problem transformation method.

Samples	Attribute	Multi-Label Set			New Label
		CW	*LFM*	*BPSK*	
1	CWBPSK	1	0	1	0
2	CWLFM	1	1	0	1
3	dBPSK	0	0	1	2
4	dCW	1	0	0	3
5	dLFM	0	1	0	4
6	LFMBPSK	0	1	1	5

3. The Method of QSAS Identification

We begin with an overview of the network structure during the pre-training phase. We explain how the network enables SSL and the advantages it provides. The details required to use the model in downstream tasks are then detailed. These include the network structure and loss function.

3.1. Pre-Training

Definition of Self-Supervised Representation Learning: Automatically learn features with a recognition ability applicable to multiple downstream tasks under the absence of signal labels.

In the pre-training phase, the state of the art MAE is used to learn the time-frequency information representation of QSAS as shown in Figure 2. MAE covers patches of the input image at random and reconstructs missing pixels for learning a low-dimensional representation of data. After the STFT, we send time-frequency images to the autoencoder. The design rules of the autoencoder are as follows:

1. Asymmetrical encoder–decoder network architecture is used. The encoder only operates on visible patches, and the potential feature representation of visible patches is obtained through convolution operation, namely the blue block in the figure. The lightweight decoder can refactor signals in time-frequency domains based on underlying features and mask tokens (orange blocks) by combining time-frequency information.
2. High mask ratio increases the difficulty and improves the generalization ability of model learning. When the mask ratio is high, the model needs to learn from limited information and reconstruct the original data, which can force the model to learn more robust feature representation, thus improving the generalization ability of the model. In addition, a high mask ratio can reduce the model's dependence on local details, thus improving the model's ability to understand the overall structure. However, a too high mask ratio will also lead to information loss and underfitting of the model, so it is necessary to make reasonable adjustment in practice.

The details of the autoencoder network are shown in the Figure 3, where k represents the size of the convolution kernel, s represents the step length, namely the sampling interval, and p represents the padding. The small yellow block represents a ConvNeXt V2 block, as detailed on the right. The small blue block represents the downsample, as detailed on the right. The data conversion process is as follows: firstly, the whole time-frequency graph is downsampled and divided into multiple patches, and the input image is compressed into a low-dimensional representation. Then, we add a mask to randomly patches of the image area, and a dense tensor is converted into a sparse tensor. After feature extraction by multiple sparse convolution layers of the encoder, the sparse tensor is converted into a dense tensor. Then we restore the masked area by the decoder.

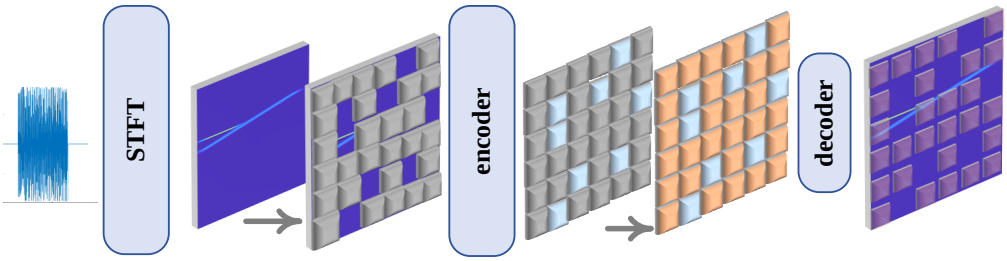

Figure 2. Simple illustration of autoencoder network model.

Wherein, the generating process of a mask is as follows: Firstly, a mask tensor is defined. Its shape is the same as the input image. Its initial value is all 1. Then, we generate a random set of rectangles representing the areas to be masked. Then, we set the value of the corresponding rectangular region in the mask tensor to 0. In order to accommodate the convolution operation, we upsample the input mask, enlarge to the size of the last layer, and add a dimension to mask by the unsqueeze function, so its type is the same as the input data. Then, the mask tensor is used as a multiplication factor to multiply the input patches to obtain the masked image.

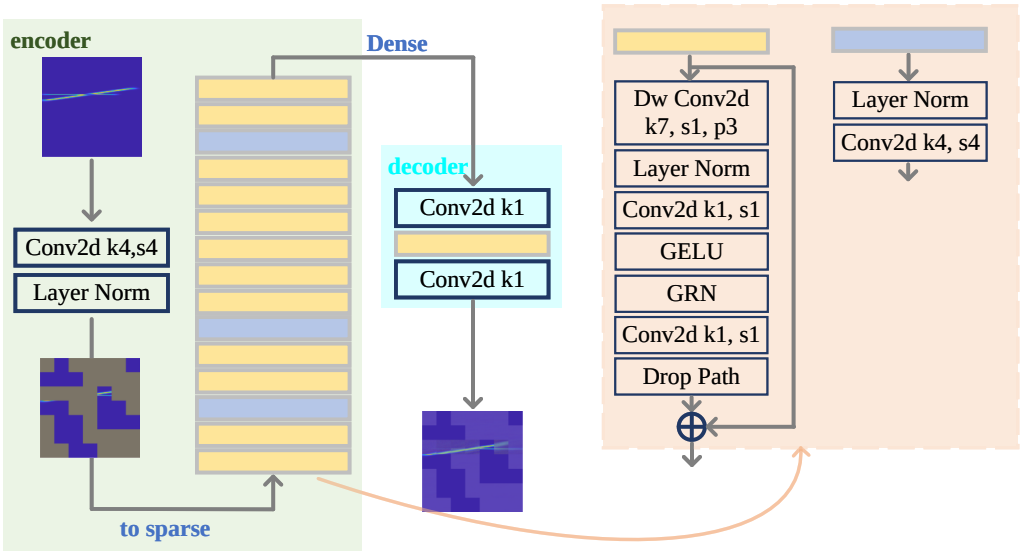

Figure 3. Autoencoder network structure diagram.

3.1.1. Encoder

The encoder, shown in the light green background in the Figure 3, compose data downsampling, to sparse, complex CNN layers and dense. Data downsampling is to use the convolution layer with a convolution kernel size of 4 and step length of 4 to segment the image into patches. The details of the complex network layer are in the next subsection. The uniqueness of the encoder lies in the sparse and dense. We treat the mask input as a set of sparse patches, namely a two-dimensional sparse pixel array, and use sparse convolution to process the visible part.

Because of the disorder of sparse data, voxelizing the points and applying convolution to three-dimensional grids is a natural solution [47]. Sparse convolution is a practical substitution for vanilla 3D convolution, and skips the non-active regions that only operate when the center of the convolutional kernel covers active voxels. Active voxels are stored as sparse tensors for fixed convolution operations. Sparse input data are transformed into a sparse matrix, which only stores non-zero positions (effective sites) and corresponding weight values. Submanifold sparse convolution (SSC) is used to reduce the influence of sparsity. SSC is calculated only when the center of the convolution kernel slides through the activate sites of the sparse matrix. The size and shape of the convolution kernel can be

adjusted adaptively, and the same sparsity degree can be maintained in the whole network, which is suitable for deep CNN.

The traditional convolution operation is to convolve every site of the input tensor, which leads to a lot of redundant computation. Sparse convolution only convolves non-zero sites in the input tensor, so a large number of multiplication operations in the convolution operation can be turned to 0. The SSC reduces the amount of computation and improves the efficiency of training and testing under the condition of making full use of the input information.

3.1.2. Decoder

The decoder is a single ConvNext V2 block, shown in the blue background in the Figure 3, that uses a full CNN to generate mask tokens. The decoder restores the low-dimensional representation of the time-frequency image obtained by the encoder to the original image. With a lightweight decoder, we reconstruct the original image based on underlying features and mask tokens. Specifically, the decoder receives underlying features output by the encoder and mask tokens, restores the underlying feature representation to the original image through deconvolution operation, and fills the mask in the original image according to the mask tokens.

Mean squad error (MSE) is the reconstructed loss function, and the error was calculated only on the mask part. The goal of the network is to minimize the reconstruction error. We try to reconstruct the sample as close to the original sample as possible, so that the network can learn the effective representation of the original image.

$$MSE = \frac{\sum(pred - target)^2}{3p^2} \text{ only if mask } = 1, \tag{9}$$

where $pred$ represents the predicted value, and $target$ represents the true value. For each sample, we calculate $(pred - target)^2$ and sum up the squared differences for all samples. Then, we divide this sum by $3p^2$, where 3 represents the number of channels per pixel, and p represents the resolution of the patch, namely patch size, and p^2 represents the number of pixels per channel.

3.2. Fine-Tuning

Transfer learning: Pre-train the existing large-scale dataset, then transfer the learned knowledge to the classification task of few-shot, and fine-tune parts of layers to improve the classification accuracy.

After pre-training the encoder map, input the data to the low-dimensional representation, which can be used as the feature representation of the data for the subsequent fine-tuning. During the fine-tuning, the weight is converted to the standard form, and the dense layer does not need special treatment. As shown in the Figure 4, we freeze the first few layers, that is, transfer the corresponding structural parameters of the encoder to the fine-tuning network, then add layer normalization and linear layer and train them.

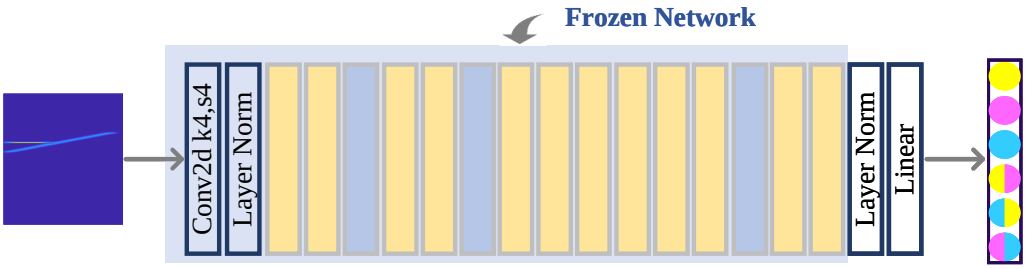

Figure 4. Fine-tuning phase network structure diagram.

Inspired by the ConvNeXt V2 neural network structure, we make full use of the advantages of the local connection and weight sharing of CNN to obtain excellent performance

of QSAS identification. ConvNeXt [14] trains the ResNet-50 in the same way as the Transformer [48] training method. On this basis, ConvNeXt experiments with Macro design, ResNeXt, Inverted bottleneck, Large kernel size, and Various layer-wise Micro. ConvNeXt V2 modifies the supervised ConvNeXt network into a SSL network by referring to MAE, but all the networks are proposed and tested in ImageNet and other datasets [49,50]. The radar signal is very different from the open dataset. Therefore, we combine the characteristics of the radar signal to further follow up the network parameters. The specific structure of the ConvNeXt V2 block (small yellow part) and downsampling block (small blue part) in the network is shown in Figure 3. The specific design process and important details of the whole network are as follows.

First, the network downsamples the image through a convolution layer with a convolution kernel size of 4×4 and a step distance of 4, and the height and width of the image are reduced to $\frac{1}{4}$ of the original. Then, it successively passes Stage 1, Stage 2, Stage 3 and Stage 4. Each stage is composed of a series of ConvNeXt V2 blocks. We will talk about the structure of the ConvNeXt V2 block. If the height, width and channel of the input feature map are h, w and dim, through the depthwise convolution with kernel size of 7×7, step size of 1, padding of 3 and through a LayerNorm, then the output size is still $h \times w \times dim$. Then, through a convolution layer, which has a kernel size of 1×1 and activation function of Gaussian error linear units (GELU). Then, the height and width remain the same, and dim increases by 4 times. Then, through a convolution layer, which has a kernel size of 1×1 and the DropPath layer, the output size reduces to $h \times w \times dim$. Then, add the input as output. Linear, as a fully connected layer, maps input features to different class labels.

The network follows the structure of VGG [51], which divides the backbone network into four different stages. The number of blocks in each stage is $(2, 2, 6, 2)$, and the input channel is $(20, 20, 40, 80)$. So the stage ratio is 1:1:3:1, consistent with Swin-transformer [52].

Depthwise convolution is a special case of group convolution in ResNeXt. The convolution kernel's channel is 1 and only convolves with a single channel of the input. The number of convolution kernels is the same as the channel of the input and the channel of the output, as shown in Figure 5a. So the channel of the feature matrix does not change, and a better balance between FLOPs and accuracy can be achieved.

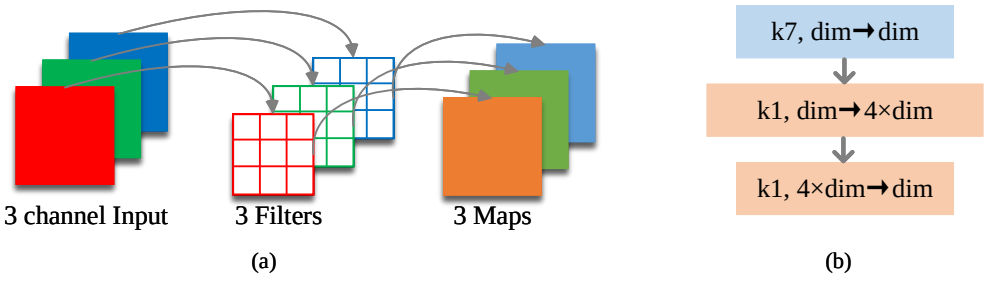

Figure 5. The depthwise convolution and the inverse bottleneck layer. (**a**) Depthwise convolution; (**b**) Inverse bottleneck.

The inverse bottleneck layer with two large, middle and small ends can effectively avoid information loss, as shown in the Figure 5b. In addition, the depthwise convolution is moved up and the convolution kernel size is changed to 7×7, which is consistent with the window size in Swin-transformer.

We use three regularization techniques: LayerNorm (LN), Global Response Normalization (GRN) and DropPath. The details are as follows. We only add one GELU activation function between two 1×1 convolutions.

1. LN is added only before the first 1×1 convolution. LN normalizes each dimension of each sample, turns the mean to 0, and turns the variance to 1. It is helpful to solve the problem of gradient vanishing and gradient explosion in neural networks and improve the generalization performance of the model.

2. GRN normalizes the features of different channels, makes features comparable, and then enhances the feature competition among channels. GRN can avoid overfitting and improve the generalization performance of the model.

$$X_i = \gamma \times X_i \times \frac{\|X_i\|}{\sum_{j=1,\ldots,c}\|X_j\|} + \beta + X_i, \tag{10}$$

where, X_i denotes the input data, $\|X_i\|$ denotes the L2 norm for the ith channel, C denotes the number of channels, and γ and β are two learnable parameters.

3. DropPath randomly selects some network layers for each training sample in the process of forward propagation and sets their output to 0. Some paths of the network are deleted randomly, so different paths will be deleted for each training sample. Another feature of DropPath is that it operates on a network hierarchy. Specifically, DropPath works on the deep structure of the network. The deep structure is often the bottleneck of the network and tends to lead to overfitting. Therefore, DropPath can help regularize the network on the deep structure, thus reducing overfitting.

The residual connection structure can effectively avoid the problem of gradient disappearing and gradient explosion, so that the network can learn features in deeper layers.

Using a separate downsample layer, a 2×2 convolution with step size 2 is inserted between different stages, and a LN is added before and after downsampling.

The loss function selects the cross entropy loss function.

4. Experiments

In this section, we assess the performance of the proposed approach using a specific simulation and computation environment. The details of this environment are provided in Table 2.

Table 2. The simulation and computation environment setting.

Configuration Type	Configuration Instruction
CPU	12th Gen Intel(R) Core(TM) i9-12900K 3.20 GHz
RAM	128 GB
GPU	NVIDIA GeForce RTX3090
OS	Ubuntu 20.04
programming language	Python
model framework	Pytorch

4.1. Datasets and Settings

To verify the performance of the proposed method, we construct a dataset of QSAS. We use three typical radars modulations: CW, LFM and BPSK. For each modulation, we set corresponding parameters, as shown in the Table 3:

Table 3. The parameter of the dataset.

Data Generation Parameter	Parameter Size Setting
f_s (MHz)	80
noise	white/colored
SNR (dB)	−16 dB∼16 dB
A	0.5/0.75/1
f_c (MHz)	25/30/35
PW (μs)	10/15/20/25
B (MHz)	3.75–6.25
time delay (μs)	0/2.5/5
time-frequency graph resolution	224

Each modulation generates 108 samples according to the above parameter range, and a total of 34,668 samples are combined in pairs. Overall, 5% of the samples are labeled for few-shot training and 20% are used for testing. To facilitate subsequent data reading, data are stored in a folder by label.

The super parameter settings of the pre-training and fine-tuning phases is shown in Table 4.

Table 4. The parameter settings of pre-training and fine-tuning.

Parameter	Pre-Training	Fine-Tuning
model	ConvNeXt V2 Atto	ConvNeXt V2 Atto
weight init	trunc. normal (0.2)	-
optimizer	AdamW	AdamW
base learning rate	1.5×10^{-4}	2×10^{-4}
weight decay	0.05	0.3
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.95$	$\beta_1, \beta_2 = 0.9, 0.999$
batch size	196	32
learning rate schedule	cosine decay	cosine decay
warmup epochs	40	0
warmup schedule	linear	-
training epochs	800	250
augmentation	RandomResizedCrop	RandAug (9, 0.5)
drop path	-	0.1
head init	-	0.001

4.2. Network Evaluation

During the pre-training phase, we visualize six types of data processing processes. In Figure 6, the left side is the original image, the middle is the input mask image, and the right side is the output reconstructed image. It can be observed that the model is capable of reconstructing the signal even in the presence of significant lost information in the time-frequency domain. This is achieved by utilizing patches from the preceding and succeeding positions. Furthermore, in certain cases where only a single signal is visible, the model can predict the presence of multiple signals.

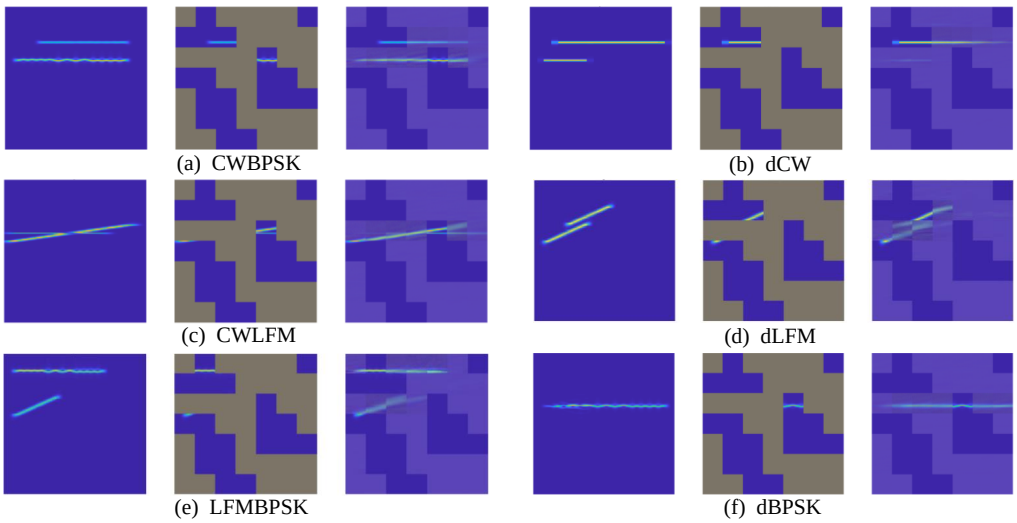

Figure 6. (**a**–**f**) Visualization of data changes in the pre-training phase.

In the fine-tuning phase, we use the recognition accuracy to demonstrate the effectiveness of the model during the test phase:

$$Accuracy = \frac{True}{True + False}, \tag{11}$$

where *True* indicates all results that are predicted correctly, and *False* indicates all results that are missed or misreported.

Figure 7 displays the classification results of non-noise signals. It is notable that the results become stable after 80 epochs. The model's accuracy steadily improves after several epochs, and there is no occurrence of overfitting.

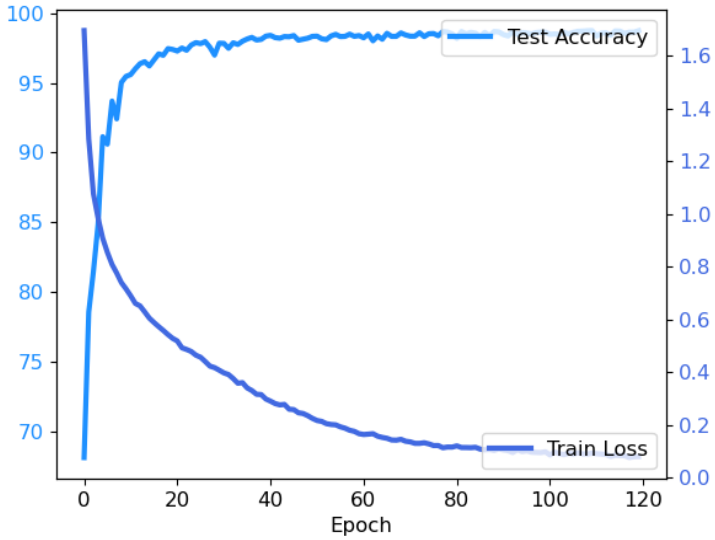

Figure 7. Non-noise signal train loss and test accuracy change with epochs.

On the QSAS dataset, we conduct experiments using two settings, namely in-domain pre-training (i.e., pre-training and fine-tuning on the same dataset) and cross-domain pre-training (i.e., pre-training and fine-tuning on different dataset). Table 5 shows that for the test with SNR of 0 dB, the effect of using the non-noise pre-training model for fine-tuning is a little better than that of using 0 dB signal for the pre-training model, which also verifies what the author mentioned in BERT [53], that using pure data for pre-training has better performance to fine-tune data with noise. Therefore, a large number of non-noise signals are used for pre-training, and −16 dB∼16 dB, 4 dB as the step signals are used for fine-tuning training, and signals with different SNR are tested.

Table 5. Pre-training dataset selection experiment. *Train* ↓ shows the selected train data; *Test* → shows the selected test data.

Train ↓ \ ***Test*** →	**Non-Noise**	***SNR* = 0 dB**
Non-noise	98.12	87.92
***SNR* = 0 dB**	96.39	82.52

Table 6 shows the average recognition accuracy of the *SNR* from −16 dB∼16 dB with different noise, Acc1 shows the testing accuracy in the presence of white noise, while Acc2 shows the testing accuracy in the presence of colored noise. Comparing the results under different noise conditions, we find that the accuracy decreases when the noise is enhanced. However, there is no significant difference in the model's recognition performance between the two types of noise, indicating that the model has a strong generalization ability. Subsequent experiments will be conducted based on white noise. From Acc1, we can see that the average recognition accuracy of the network is above 81%. When it is above 0 dB, the recognition accuracy can reach more than 90%, and when it is above −16 dB, the recognition accuracy can reach more than 57%. It can be seen that our model is suitable for low *SNR* QSAS identification.

Table 6. Average accuracy with different SNR.

SNR (dB)	Acc1	Acc2	*SNR* (dB)	Acc1	Acc2
16	97.92	96.77	−4	74.30	70.09
12	97.85	96.56	−8	64.21	60.03
8	97.44	94.98	−12	60.81	56.85
4	94.75	90.81	−16	57.02	55.24
0	87.92	82.51	**mean**	81.36	80.10

In the form of confusion matrix, we summarize and visualize the dataset labels according to two criteria, the real label and the prediction label recorded in the test experiment results, where the rows in the matrix represent the real value and the columns in the matrix represent the predicted value as shown in Figure 8. We observe that the scheme can accurately separate the six signals under the condition of high SNR. Under the condition of low SNR, it can also achieve good results for aliasing related to CW and LFM. However, for samples involving BPSK, there are some misjudgments, which may be due to the single frequency of BPSK. Therefore, it is easy to be recognized as CW, and there are strong frequency components around the IF of the BPSK. When the noise is too strong, it is easy to be confused as LFM with a low frequency modulation slope.

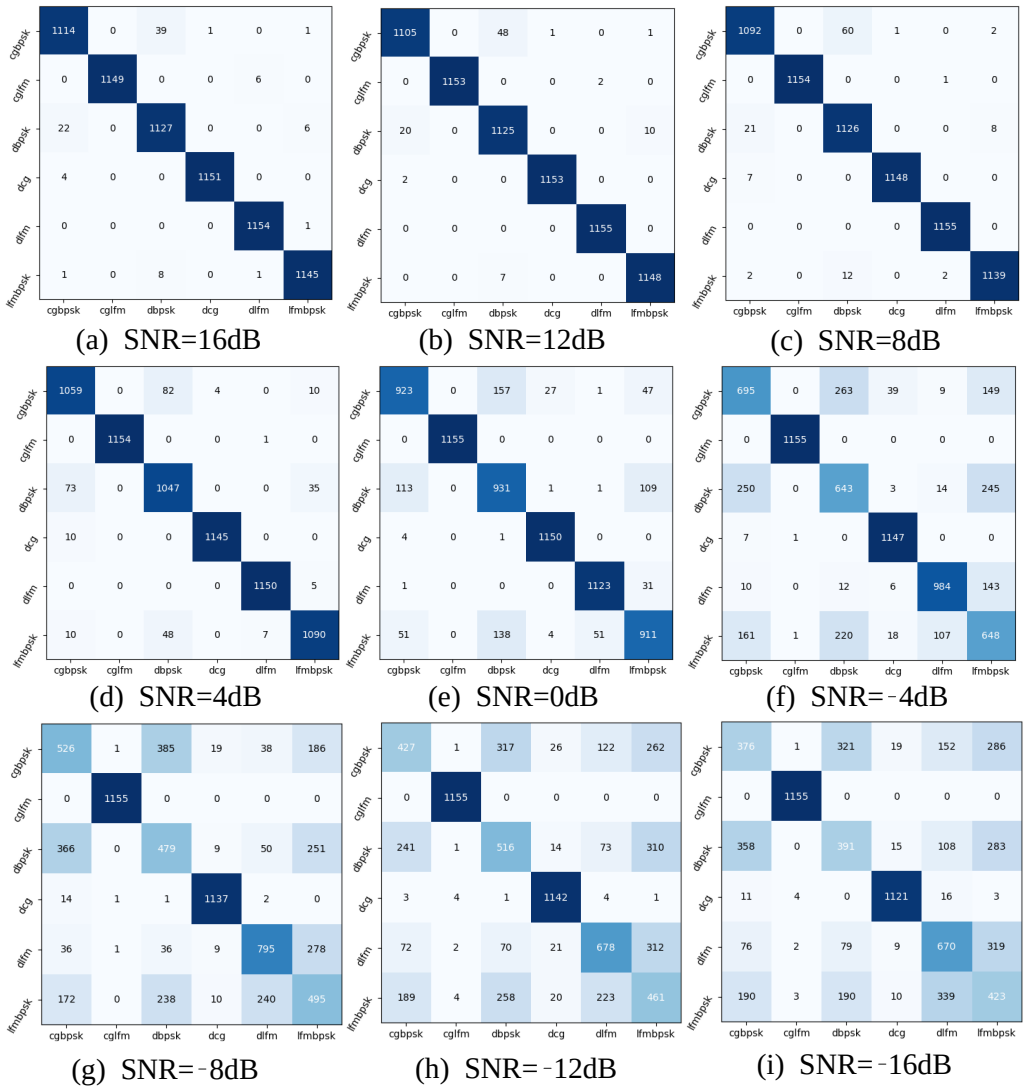

Figure 8. (**a**–**i**) Confusion matrix under different SNR. The darker the color, the greater the value.

4.3. Ablation Study

Hyperparameters refer to the parameters that need to be set manually during model training. In order to prove the rationality of each module of our method, we conduct ablation experiments on some key parameters.

4.3.1. Spatial Masking Strategy

We study different mask strategies and mask ratio ablation. According to MAE's proposal, we study a larger mask ratio, starting from a mask ratio of 50%, gradually increasing the masking ratio and adopting the random mask strategy proposed by MAE. The model performance is best when 60% of the area is masked, as shown in Figure 9.

Although the horizontal and vertical input images, respectively, represent the time-frequency domain information of the signal, VideoMAE [54] has studied that the effect of masking for different domains is not ideal in autoencoder pre-training.

Using separate time-frequency domain masks will increase the computational complexity of the algorithm. Separate time-frequency domain masks can also cause unique identity problems. Since the time-frequency domain masks are processed separately, it is possible for the encoder to limit the global features by combining different time domain or frequency domain masks, thus reducing the generalization of the algorithm. The full mask method can consider features of time domain and frequency domain at the same time, so as to retain the information of the RE signal better. The full mask will dynamically adjust the radar signal according to the time-frequency domain features, so as to better retain the information of the signal. Therefore, full mask is a more effective radar signal processing method.

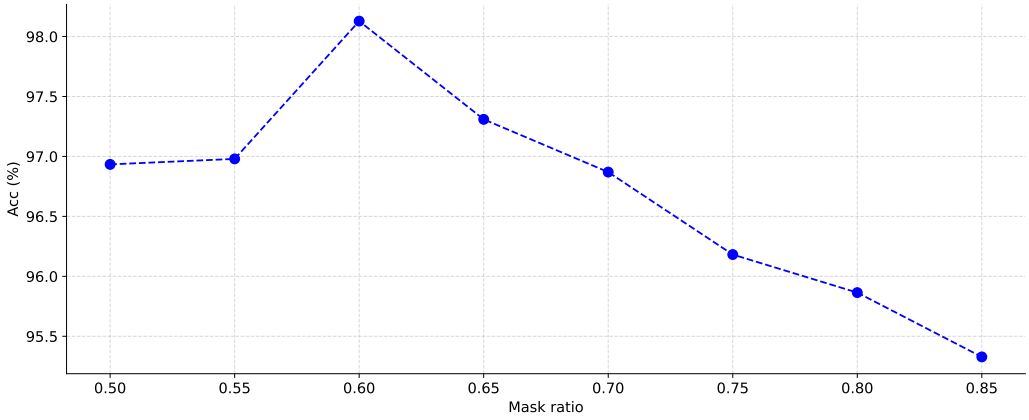

Figure 9. Mask strategy ablation experiment.

4.3.2. Decoder Setting

We conducted ablation experiments that constitute different decoder blocks with different numbers and dimensions, as shown in the Table 7. Experimental results show that using a simple block has the best performance. Increasing the number of decoders or increasing the depth introduces more computing costs, but does not result in better improvements.

Table 7. Decoder ablation experiment.

Depth	Dim	Acc
4	128	97.03
1	512	97.21
1	256	97.04
1	128	**98.12**

The above two ablation experiments also demonstrate the validity of high mask ratio and asymmetric encoder decoder designs.

4.3.3. Relationship Between Patch Size and Mask Ratio

Table 8 shows ablation experiments with different patch sizes. When the patch size is 32, we divide the input picture with a resolution of 224 × 224 into 7 × 7 patches; when the patch size is 16, we divide the input picture with a resolution of 224 × 224 into 14 × 14 patches. We observe that under the condition of 60% mask ratio, the patch size of 32 has better recognition accuracy. We consider that it might be difficult for the model to predict fine-grained signal details. However, when the mask ratio was increased, the recognition accuracy on smaller patches is quite good, but this increased the training time of the model. We chose the patch size of 32.

Table 8. Patch size ablation experiment.

Patch Size	Mask Ratio	Acc	Time (s)
16	0.6	96.76	11
32	0.6	**98.12**	3
16	0.75	96.90	11

4.3.4. Number of Pre-Training Epochs

Table 9 shows ablation experiments on the influence of the number of pre-training epochs on the model results. When the autoencoder model is continuously iterated, the reduction of reconstruction loss was no longer obvious after reaching a certain number of epochs. However, we find that the internal features could still be learned after multiple epochs, thus improving the fine-tuning accuracy.

Table 9. Epochs ablation experiment.

Epoch	Acc
450	96.38
800	**98.12**

4.4. Comparison

In addition, we compare the results with ConvNeXt, MAE, ViT, ResNet, VGG, SVM and KNN. The relationship and difference between ConvNeXt V2 and the previous three models are listed in Table 10. ResNet and VGG are widely recognized as universal DL models in the field of signal identification, while SVM and KNN are traditional methods.

Table 10. The relationship and difference between ConvNeXt V2 and ConvNeXt, MAE, ViT.

Study	Self-Supervised	Supervised
Conv	ConvNeXt V2	ConvNeXt
Transformer	MAE	ViT

MAE pre-trains the data for 800 epochs and then fine-tunes it for 250 epochs. ConvNeXt and ViT train for 250 epochs directly on 5% of the data. The recognition accuracy of the four models under different SNR conditions is shown in Figure 10. The red bars represent ViT recognition results, orange bars represent MAE recognition results, green bars represent ConvNeXt recognition results, and blue bars represent ConvNeXt V2 recognition results. It can be observed that low SNR has a significant influence on model accuracy. However, the accuracy of ConvNeXt V2 models described in this paper is higher than other models, which confirms the applicability of ConvNeXt V2 for QSAS identification.

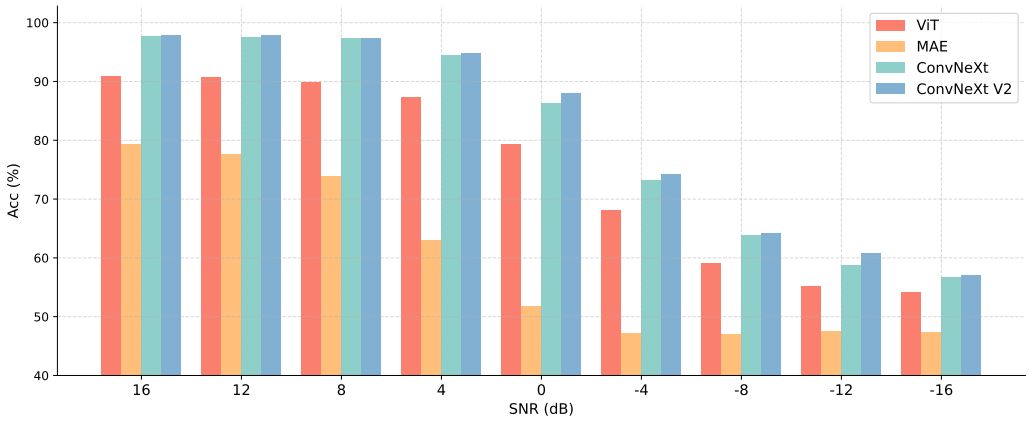

Figure 10. Average accuracy of contrast algorithm under different SNR.

The input for SVM and KNN is a 1 × 50,176 array obtained by flattening a 224 × 224 spectrogram. ResNet and VGG train for 250 epochs on 5% of the data. The average recognition accuracy and test time of the eight models are shown in Table 11. The average recognition accuracy refers to the accuracy averaged over the SNR range of −16 dB∼16 dB with a step size of 4 dB. The test time refers to the time taken to predict the results of 6930 time-frequency graphs using only one NVIDIA GeForce RTX3090. It can be seen that the traditional ML methods of SVM and KNN have the lowest accuracy, and the prediction speed has no comparative significance. Our proposed method achieves the highest recognition accuracy and the lowest time consumption. In terms of recognition accuracy, the effect of convolution is much greater than that of Transformer, indicating that different network models should be selected for different datasets. However, the time consumption of SSL convolution is much lower than that of SL convolution. A single graphics card can complete the recognition of 10,000-level pulses within 5 s. Moreover, the CNN based on the ConvNeXt has higher performance than ResNet and VGG.

Table 11. The average accuracy and test time of the compared algorithms.

Method	Acc	Time (s)	Method	Acc	Time (s)
SVM	58.56	170	ViT	74.95	13
KNN	35.94	260	MAE	59.41	9
ResNet	75.56	40	ConvNeXt	80.67	33
VGG	74.51	56	ConvNeXt V2	**81.36**	**3**

In general, the proposed methods demonstrate excellent performance in different modulations of radars, and can be applied to practical radar signal processing.

5. Conclusions

In this paper, ConvNeXt V2 was adopted to identify QSAS in two-dimensional time-frequency images. The motivation was to address the challenge raised by the fine-grained identification of QSAS, which was ubiquitous in the field of signal recognition. Moreover, the intercepted signal by UAV was simulated by overlapping information in time-frequency domains. The training dataset for the model was designed by multiple parameters in different noise environments, and the label powerset was used for the data annotation task. Most importantly, the ablation study evaluated the impact of each parameter on the performance for identifying QSAS. Compared with ConvNeXt, MAE, ViT, ResNet, VGG, SVM and KNN, the classification accuracy of few-shot can be significantly improved with the usage of the proposed pre-train and fine-tune phases, and the time consumption was lower via SSL. The generalization was strong in different noise. Experimental results showed that the proposed algorithm was superior to all other models in QSAS identification.

Future work can study the interpretability of neural networks for the essential features of radars.

Author Contributions: Conceptualization, L.G. and J.P.; methodology, L.G. and Z.W.; software, L.G. and M.D.; validation, L.G. and J.X.; formal analysis, L.G.; investigation, L.G. and J.X.; resources, J.P.; data curation, L.G.; writing—original draft preparation, L.G.; writing—review and editing, L.G., M.D., J.X., Z.W. and J.P.; visualization, L.G.; supervision, J.P.; project administration, J.P.; funding acquisition, J.P. and L.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China under Grant 62071476, and Postgraduate Scientific Research Innovation Project of Hunan Province under Grant CX20220031.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: We generated the data via a specific radar emitter, and the radar signal is private, so it is not convenient to disclose it.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zhao, X.; Yang, R.; Zhang, Y.; Yan, M.; Yue, L. Deep reinforcement learning for intelligent dual-UAV reconnaissance mission planning. *Electronics* **2022**, *11*, 2031. [CrossRef]
2. Hu, L.; Xi, B.; Yi, G.; Zhao, H.; Zhong, J. A multiple heterogeneous UAVs reconnaissance mission planning and re-planning algorithm. *J. Syst. Eng. Electron.* **2022**, *33*, 1190–1207.
3. Guo, Y.; Tang, H.; Qin, R. A Low Complexity Persistent Reconnaissance Algorithm for FANET. *Sensors* **2022**, *22*, 9526. [CrossRef] [PubMed]
4. Wan, P.; Hao, B.; Li, Z.; Ma, X.; Zhao, Y. Accurate estimation the scanning cycle of the reconnaissance radar based on a single unmanned aerial vehicle. *IEEE Access* **2017**, *5*, 22871–22879. [CrossRef]
5. Fang, F.; Zhang, G.; Cheng, Y. Power line identification of millimeter wave radar based on PCA-GS-SVM. *IOP Conf. Ser. Mater. Sci. Eng.* **2017**, *274*, 012139. [CrossRef]
6. Cheng, Y.; Guo, M.; Guo, L. Radar signal recognition exploiting information geometry and support vector machine. *IET Signal Process.* **2023**, *17*, e12167. [CrossRef]
7. Ma, Y.; Chen, T.; Wang, H. Application of Complex Network in Intra-pulse Feature Extraction of Radar Emitter Signals. In Proceedings of the 2022 IEEE 22nd International Conference on Communication Technology (ICCT), Nanjing, China, 11–14 November 2022; pp. 1847–1851.
8. Gupta, A.; Rai, A.B. Feature extraction of intra-pulse modulated LPI waveforms using STFT. In Proceedings of the 2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT), Bangalore, India, 17–18 May 2019; pp. 742–746.
9. Qu, Z.; Hou, C.; Hou, C.; Wang, W. Radar signal intra-pulse modulation recognition based on convolutional neural network and deep Q-learning network. *IEEE Access* **2020**, *8*, 49125–49136. [CrossRef]
10. Wei, S.; Qu, Q.; Su, H.; Wang, M.; Shi, J.; Hao, X. Intra-pulse modulation radar signal recognition based on CLDN network. *IET Radar Sonar Navig.* **2020**, *14*, 803–810. [CrossRef]
11. Sui, J.; Liu, Z.; Liu, L.; Peng, B.; Liu, T.; Li, X. Online non-cooperative radar emitter classification from evolving and imbalanced pulse streams. *IEEE Sens. J.* **2020**, *20*, 7721–7730. [CrossRef]
12. Pan, J.; Guo, L.; Chen, Q.; Zhang, S.; Xiong, J. Specific Radar Emitter Identification Using 1D-CBAM-ResNet. In Proceedings of the 2022 14th International Conference on Wireless Communications and Signal Processing (WCSP), Nanjing, China, 1–3 November 2022; pp. 483–488.
13. Wang, M.; Yu, D.; He, W.; Yue, P.; Liang, Z. Domain-incremental learning for fire detection in space-air-ground integrated observation network. *Int. J. Appl. Earth Obs. Geoinf.* **2023**, *118*, 103279. [CrossRef]
14. Liu, Y. Aircraft Detection in Aerial Remote Sensing Images Based on Contrast Self-supervised Learning. In *International Conference on Machine Learning and Intelligent Communications*; Springer: Cham, Switzerland, 2022; pp. 284–296.
15. Fang, Z.; Xi, Z.; Xu, M.; Fan, X. A ViT-based lightweight method for the UAV platform object detection tasks. In Proceedings of the 4th International Conference on Information Science, Electrical, and Automation Engineering (ISEAE 2022), Hangzhou, China, 25–27 March 2022; Volume 12257, pp. 318–324.
16. Cai, Z.; Ghosh, S.; Stefanov, K.; Dhall, A.; Cai, J.; Rezatofighi, H.; Haffari, R.; Hayat, M. MARLIN: Masked Autoencoder for facial video Representation LearnINg. *arXiv* **2022**, arXiv:2211.06627.
17. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.

18. He, K.; Chen, X.; Xie, S.; Li, Y.; Dollár, P.; Girshick, R. Masked autoencoders are scalable vision learners. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–24 July; pp. 16000–16009.
19. Woo, S.; Debnath, S.; Hu, R.; Chen, X.; Liu, Z.; Kweon, I.S.; Xie, S. ConvNeXt V2: Co-designing and Scaling ConvNets with Masked Autoencoders. *arXiv* **2023**, arXiv:2301.00808.
20. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 11–18 December 2015; pp. 1026–1034.
21. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
22. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
23. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp, 580–587.
24. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
25. Read, J.; Puurula, A.; Bifet, A. Multi-label classification with meta-labels. In Proceedings of the 2014 IEEE International Conference on Data Mining, Shenzhen, China, 14–17 December 2014; pp. 941–946.
26. Tenenboim-Chekina, L.; Rokach, L.; Shapira, B. Identification of label dependencies for multi-label classification. In *Working Notes of the Second International Workshop on Learning from Multi-Label Data*; Citeseer: Princeton, NJ, USA, 2010; pp. 53–60.
27. Pushpa, M.; Karpagavalli, S. Multi-label classification: Problem transformation methods in Tamil phoneme classification. *Procedia Comput. Sci.* **2017**, *115*, 572–579. [CrossRef]
28. Cherman, E.A.; Monard, M.C.; Metz, J. Multi-label problem transformation methods: A case study. *CLEI Electron. J.* **2011**, *14*, 4. [CrossRef]
29. Yap, X.H.; Raymer, M. Multi-label classification and label dependence in in silico toxicity prediction. *Toxicol. In Vitro* **2021**, *74*, 105157. [CrossRef]
30. Griffin, D.; Lim, J. Signal estimation from modified short-time Fourier transform. *IEEE Trans. Acoust. Speech Signal Process.* **1984**, *32*, 236–243. [CrossRef]
31. Li, J.; Fu, S.; Xie, X.; Xiang, M.; Dai, Y.; Yin, F.; Qin, Y. Low-latency short-time Fourier Transform of microwave photonics processing. *J. Light. Technol.* **2023**. [CrossRef]
32. Durak, L.; Arikan, O. Short-time Fourier transform: Two fundamental properties and an optimal implementation. *IEEE Trans. Signal Process.* **2003**, *51*, 1231–1242. [CrossRef]
33. Li, M.; Liu, Y.; Zhi, S.; Wang, T.; Chu, F. Short-time Fourier transform using odd symmetric window function. *J. Dyn. Monit. Diagn.* **2022**, *1*, 37–45. [CrossRef]
34. Kaneko, T.; Tanaka, K.; Kameoka, H.; Seki, S. iSTFTNet: Fast and lightweight mel-spectrogram vocoder incorporating inverse short-time Fourier transform. In Proceedings of the ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Singapore, 23–27 May 2022; pp. 6207–6211.
35. Weiss, K.; Khoshgoftaar, T.M.; Wang, D. A survey of transfer learning. *J. Big Data* **2016**, *3*, 9. [CrossRef]
36. Li, X.; Feng, S.; Guo, Y.; Li, H.; Zhou, Y. Denoising Method for Microseismic Signals with Convolutional Neural Network Based on Transfer Learning. *Int. J. Comput. Intell. Syst.* **2023**, *16*, 91. [CrossRef]
37. Lin, A.; Ma, Z.; Huang, Z.; Xia, Y.; Yu, W. Unknown radar waveform recognition based on transferred deep learning. *IEEE Access* **2020**, *8*, 184793–184807. [CrossRef]
38. Johnson, D.H. Signal-to-noise ratio. *Scholarpedia* **2006**, *1*, 2088. [CrossRef]
39. Graham, B.; Van der Maaten, L. Submanifold sparse convolutional networks. *arXiv* **2017**, arXiv:1706.01307.
40. Parrish, T.B.; Gitelman, D.R.; LaBar, K.S.; Mesulam, M.M. Impact of signal-to-noise on functional MRI. *Magn. Reson. Med.* **2000**, *44*, 925–932. [CrossRef]
41. Nowak, M.; Wicks, M.; Zhang, Z.; Wu, Z. Co-designed radar-communication using linear frequency modulation waveform. *IEEE Aerosp. Electron. Syst. Mag.* **2016**, *31*, 28–35. [CrossRef]
42. Levanon, N. Multifrequency complementary phase-coded radar signal. *IEE Proc.-Radar Sonar Navig.* **2000**, *147*, 276–284. [CrossRef]
43. Zhang, S.; Pan, J.; Han, Z.; Guo, L. Recognition of noisy radar emitter signals using a one-dimensional deep residual shrinkage network. *Sensors* **2021**, *21*, 7973. [CrossRef]
44. Thangaraj, A.; Kramer, G.; Böcherer, G. Capacity bounds for discrete-time, amplitude-constrained, additive white Gaussian noise channels. *IEEE Trans. Inf. Theory* **2017**, *63*, 4172–4182. [CrossRef]
45. Wen, F.; Xiong, X.; Su, J.; Zhang, Z. Angle estimation for bistatic MIMO radar in the presence of spatial colored noise. *Signal Process.* **2017**, *134*, 261–267. [CrossRef]
46. Li, T.; He, Q.; Peng, Z. Parameterized Resampling Time-Frequency Transform. *IEEE Trans. Signal Process.* **2022**, *70*, 5791–5805. [CrossRef]
47. Xu, X.; Wang, Z.; Zhou, J.; Lu, J. Binarizing sparse convolutional networks for efficient point cloud analysis. *arXiv* **2023**, arXiv:2303.15493.

48. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 6000–6010.
49. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]
50. Zhou, B.; Lapedriza, A.; Xiao, J.; Torralba, A.; Oliva, A. Learning deep features for scene recognition using places database. *Adv. Neural Inf. Process. Syst.* **2014**, *27*, 487–495.
51. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
52. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In Proceedings of the Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021; pp. 10012–10022.
53. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
54. Huang, P.Y.; Xu, H.; Li, J.; Baevski, A.; Auli, M.; Galuba, W.; Metze, F.; Feichtenhofer, C. Masked autoencoders that listen. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 28708–28720.