remote sensing

MDPI

Article

Fine-Grained Ship Classification by Combining CNN and Swin Transformer

Liang Huang [1], Fengxiang Wang [1,*], Yalun Zhang [2] and Qingxia Xu [3]

[1] College of Electronic Engineering, Naval University of Engineering, Wuhan 430000, China; jwf20@mails.tsinghua.edu.cn
[2] Institute of Noise & Vibration, Naval University of Engineering, Wuhan 430000, China; carterdickson@163.com
[3] College of International Studies, National University of Defense Technology, Wuhan 430000, China; cuihaotian@sjtu.edu.cn
* Correspondence: lvteng@stu.ouc.edu.cn; Tel.: +86-159-7298-8910

Abstract: The mainstream algorithms used for ship classification and detection can be improved based on convolutional neural networks (CNNs). By analyzing the characteristics of ship images, we found that the difficulty in ship image classification lies in distinguishing ships with similar hull structures but different equipment and superstructures. To extract features such as ship superstructures, this paper introduces transformer architecture with self-attention into ship classification and detection, and a CNN and Swin transformer model (CNN-Swin model) is proposed for ship image classification and detection. The main contributions of this study are as follows: (1) The proposed approach pays attention to different scale features in ship image classification and detection, introduces a transformer architecture with self-attention into ship classification and detection for the first time, and uses a parallel network of a CNN and a transformer to extract features of images. (2) To exploit the CNN's performance and avoid overfitting as much as possible, a multi-branch CNN-Block is designed and used to construct a CNN backbone with simplicity and accessibility to extract features. (3) The performance of the CNN-Swin model is validated on the open FGSC-23 dataset and a dataset containing typical military ship categories based on open-source images. The results show that the model achieved accuracies of 90.9% and 91.9% for the FGSC-23 dataset and the military ship dataset, respectively, outperforming the existing nine state-of-the-art approaches. (4) The good extraction effect on the ship features of the CNN-Swin model is validated as the backbone of the three state-of-the-art detection methods on the open datasets HRSC2016 and FAIR1M. The results show the great potential of the CNN-Swin backbone with self-attention in ship detection.

Keywords: image classification; ship detection; remote sensing images; self-attention; transformer; CNN

Citation: Huang, L.; Wang, F.; Zhang, Y.; Xu, Q. Fine-Grained Ship Classification by Combining CNN and Swin Transformer. *Remote Sens.* **2022**, *14*, 3087. https://doi.org/10.3390/rs14133087

Academic Editors: Qi Wang, Xiangtao Zheng and Fulin Luo

Received: 10 May 2022
Accepted: 23 June 2022
Published: 27 June 2022

1. Introduction

Ship target classification and detection are of great significance to a country's maritime rights, interests, and security. In the civilian field, ship target classification and detection have broad application prospects in monitoring maritime traffic, safeguarding maritime rights and interests, and improving the early warning capabilities of maritime defense. They can be used for monitoring and managing activities such as managing water transportation traffic, rescuing ships in distress, and monitoring illegal fishing, illegal smuggling, and illegal dumping of oil in specific maritime areas, bays, and ports. In the military field, ships are key targets for maritime monitoring and wartime strikes, so the ability to identify the intent of naval battlefield ship targets at a tactical level quickly and accurately and to provide support for commanders' decisions is greatly related to the success of a war. Therefore, whether in the military or civilian field, accurate ship classification and detection have a very wide range of application scenarios and technical requirements.

Image recognition technology is essentially a kind of mathematical mapping from pattern space to class space. Early image recognition technology was based on image segmentation and other methods that can deeply analyze and accurately model the characteristics of images. For visible images, traditional image recognition and classification use techniques including pixel-level edge detection, genetic algorithms, and support vector machine (SVM) classification. Considering significant changes in image properties often reflect significant events and changes in properties, the purpose of edge detection is to identify points in a digital image with significant changes in brightness. Genetic algorithms are used to combine and optimize the discretized image templates, thus, transforming the image recognition problem into a series of discrete-point combinatorial optimization problems. SVM converts the pixel values of all pixels of the image into vectors to implicitly map its input to a high-dimensional feature space. However, this approach is highly dependent on a predefined distribution or artificially designed features, which results in low robustness and poor generalization of the algorithm. Use of convolutional neural networks (CNNs) has gradually become the mainstream approach in computer vision after the development of AlexNet [1]. Since then, with deeper and wider networks, CNNs have been performing better than before. To increase the depth of the network, the Visual Geometry Group (VGG) [2] was proposed, and it made 3×3 convolutional kernels mainstream. To widen CNNs, the Google Inception Network (GoogLeNet) series [3–6] proposed an inception block to increase the widths of convolutional neural networks, and it achieved good results. However, with the increase in depth and width, CNNs inevitably face a series of problems, such as vanishing gradients. To alleviate this problem, the Residual Networks (Resnet) [7] model included a residual connection module. For the Dense Convolutional Network (Densenet) [8], a dense connectivity was proposed to effectively alleviate the vanishing gradient problem, strengthening feature propagation and encouraging feature reuse using dense connections. To further reduce the parameters, lightweight deep neural networks were built in the efficient models for mobile and embedded vision applications, which are called the Mobilenet series [9–11], by depth-wise separable convolutions. Efficientnet [12] amplifies the network through the depth, width, and resolution. The optimal set of composite coefficients can be obtained by the neural structure search technique, and, thus, good feature extraction performances have been achieved.

CNNs are widely adopted in ship target classification and detection. Lin et al. [13] integrated a CNN and the k-nearest neighbors (KNN) method to classify ships from dual-polarized data. Jeon et al. [14] combined traditional methods of image processing and target recognition methods based on CNNs, supporting the development of AI-based ship vision systems. Li et al. [15] summarized traditional algorithms that combined image processing and machine learning with target recognition algorithms based on convolutional neural networks. Julianto et al. [16] proposed a method that combined a generative adversarial network and convolutional neural networks for recognizing small ships, significantly improving the correctness and robustness. Chen et al. [17] improved AlexNet for deep feature extraction of ship images. Zhao et al. [18] combined the detection of visual salience and CNNs and proposed a fusion model that effectively improved the detection accuracy of ships. Xu et al. [19] combined image pre-processing, image smoothing, and anti-cloud interference algorithms based on the fusion of visible and infrared dual-spectrum images to achieve the detection of ship targets in complex land and sea backgrounds. Gao et al. [20] proposed a CNN framework with fewer layers and parameters that had good classification results when applied to a ship dataset. Ren et al. [21] proposed to learn discriminative features by self-supervised learning and constructed two small optical ship image datasets to validate the effectiveness. Li et al. [22] proposed an optical remote sensing image ship detection method based on a visual attention enhancement network. Bi et al. [23] combined the CNN-ZFNet architecture and random forests to improve ship target detection accuracy.

In recent years, self-attention has been widely applied in natural language processing (NLP). The transformer proposed by Google [24] makes the most of attention and has achieved better results in NLP. In 2020, the transformer structure was applied to computer

vision (CV) for the first time. Vision Transformer (ViT) [25], the first transformer structure applied to CV, demonstrated the feasibility of transformer applications in CV. The good application of self-attention allowed ViT to have a better global reception than CNNs. After ViT was developed, there were many follow-ups [26–30] that further analyzed and developed ViT structures. The Pooling-Based Vision Transformer (PiT) [31] introduced pooling operations into ViT structures to enrich the criteria for transformer structure design which performed well in image classification and a series of downstream tasks. Class-Attention in Image Transformers (CaiT) [32] proposed a deeper ViT structure devoted to image classification and discussed optimization techniques for improving the performance. The Swin transformer [33] showed the disadvantages of ViT, i.e., large video memory usage and difficulty in applying it to downstream tasks, and solved them by limiting the scope of the attention computation to solve the problem of varying scales of objects in images.

Recently, attention mechanisms have been applied in remote sensing yield. The Local Perception Swin transformer (LPSW) [34] backbone was designed to enhance the local perception of the network and to improve the detection accuracy of small-scale objects in remote sensing images. The Pyramid Information Distillation Attention Block (PIDAB) [35] was proposed to extract the complex spatial distributions and rich details of remote sensing images with a pyramid information distillation (PID) module and a hybrid attention mechanism (HAM) module.

However, there is no model that takes advantage of a transformer for ship image classification and detection. To further enhance the feature extraction capability of CNNs, this paper introduces a transformer structure with self-attention into ship classification and detection for the first time and proposes a combined model named the CNN-Swin model. At the same time, to avoid overfitting due to the model fusion, a multi-branch CNN structure was redesigned and then a Resnet-like CNN backbone was constructed. For the distinct characteristics of ship image features, a CNN-Swin model was proposed for ship image classification and detection. The features of the model are as follows: (1) The design of the CNN backbone combines the simplicity of models, such as VGG, and the accessibility to the training of multi-branch models, such as Resnet. This gives full play to the advantages of the CNN structure on a local receptive field, avoiding overfitting and improving the effectiveness. (2) The transformer structure with self-attention was applied to ship target recognition and classification for the first time, and a structure that parallelized the CNN and transformer for extracting features of images was designed.

To validate the performance of the CNN-Swin model, the model was validated using the open FGSC-23 [36] dataset. In addition, a total of 2973 typical military ship images from seven categories were collected from the internet as extra experimental data to make up for the lack of data in the FGSC-23 dataset. All the results were compared with nine state-of-the-art approaches to demonstrate the superiority of the CNN-Swin model. The CNN-Swin model demonstrated the great potential of transformer structures in ship image classification.

Furthermore, the CNN-Swin model we proposed was adopted as the backbone to extract features with different state-of-the-art ship detection methods on the open ship detection dataset HRSC2016 [37] and part of the FAIR1M dataset [38]. Experiments verified that the CNN-Swin exhibits an excellent extraction effect for the features of ship images compared to different state-of-the-art ship detection methods due to the fusion of a CNN and a transformer.

This paper is structured as follows: Section 1 briefly introduces the research background, the state of the art, the problems faced, and the main innovative aspects of this paper. Section 2 provides a detailed description of the proposed CNN-Swin model. First, the CNN-Swin model is introduced in general for the fusion of CNNs and transformers and then the novel CNN-backbone, the adopted transformer structure, and the multi-layer perceptron neural network employed are introduced. Section 3 presents the experiments on the CNN-Swin model on different datasets. First, the four datasets used by CNN-Swin are introduced. After presenting the model details, a series of comparison experiments

with other state-of-the-art approaches as references for classification and detection tasks are discussed, respectively. In addition, to demonstrate the necessity of designing a two-branch parallel structure to fuse the CNN and the transformer, we designed single-branch ablation experiments and used CAM to demonstrate the feature extraction capability of each branch. In Section 4, we conclude the work of this paper.

2. Models

2.1. Overview

The analysis of the ship image dataset revealed that the differences between the ship images were in the superstructures and various types of ship equipment. Therefore, the ship images had distinct features. A CNN has a good extraction effect on features due to the inductive bias of local perception, and the self-attention of a transformer structure also has a good feature extraction ability for specific features. Therefore, we designed a CNN-Swin model that integrates a CNN and a transformer. The model network structure is shown in Figure 1.

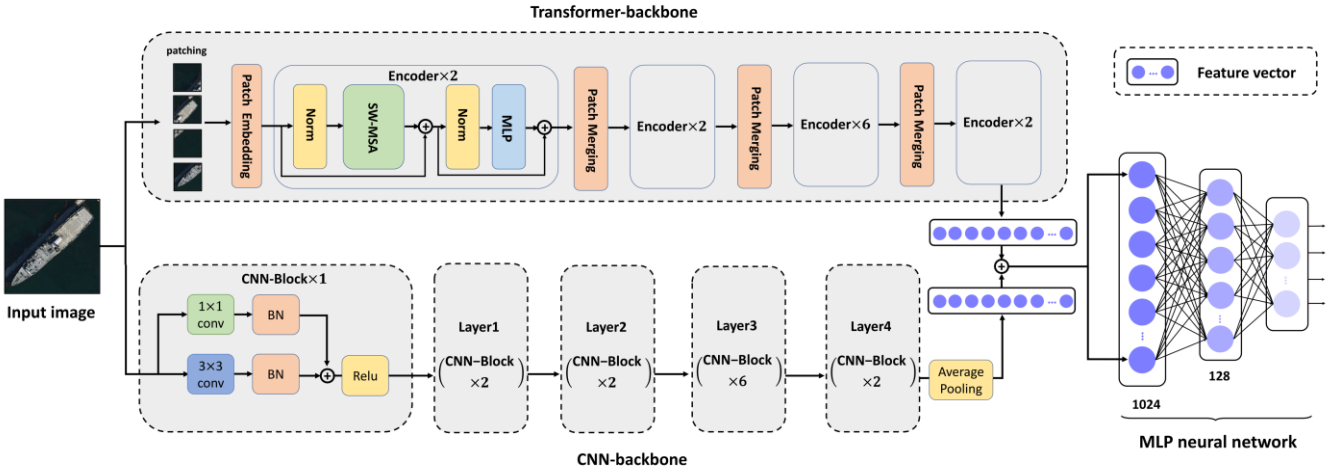


Figure 1. Network architecture of the proposed model.

The model consists of three parts: a transformer backbone, a CNN backbone, and a classification module based on multi-layer perceptron (MLP) neural networks. To exploit the advantages of a CNN structure in extracting features, the model has a CNN backbone to extract features of the image accurately and efficiently. The CNN backbone enhances the semantic information in the bottom layer of the feature map by continuously stacking CNN-Blocks, and, finally, a feature vector is output with abundant information. In the CNN backbone, a CNN-Block is first used to reduce the resolution of the feature map, then, four layers are used to extract information. The numbers of CNN-Blocks in each layer were $\{2, 2, 6, 2\}$, where the first CNN-Block in each layer uses a convolutional kernel with a stride of two pixels to reduce the resolution. Springenberg et al. [39] pointed out that the classification accuracy does not decrease and even slightly improves when using convolutional layers with a stride greater than one pixel instead of pooling layers. Therefore, a stride of two pixels is used to compress the size of the feature map to ensure not only that the spatial dimensions of the network are reduced and redundancy is removed but also that the accuracy is improved.

The transformer backbone based on the Swin transformer [33] first divides the image in each channel into 4×4 patches, with a total number of $3 \times 4 \times 4 = 48$. Each non-overlapping patch that forms the full image is referred to as a "token" [25], which means the smallest processing unit in the model. After dividing and rearranging, the transformer backbone embeds every patch, then stacks encoders to further extract image features, and, finally, outputs a feature vector with local information. Moreover, to further extract features and produce feature maps with high-level semantic information, the transformer backbone

uses patch merging before every encoder to reduce the number of tokens. Each patch merging performs a twofold downsampling of tokens, which means that the features of the neighboring 2×2 tokens are concatenated with each other.

The MLP neural network contains three fully connected layers: a 1024-dimensional input layer, a 128-dimensional hidden layer, and an output layer, the size of which depends on the number of classes after summation. The CNN-Swin model uses an MLP neural network to map the fused feature vectors into ship classification.

2.2. Convolutional Neural Network (CNN) Backbone

To avoid overfitting as much as possible and achieve better results, the CNN backbone of the model should use a concise CNN structure with a minimum number of parameters, such as the VGG model [2]. However, the Top-1 accuracy of VGG-19 on ImageNet is 72%, and it was 84.5% on the military ship dataset used in this paper. Both are lower than the current complex CNN models in terms of classification accuracy. The CNN backbone proposed in this paper is inspired by Resnet [7] and subsequently developed models based on Resnet [40–42]. The multi-branch CNN-Block module is designed based on CNN structures with fewer parameters (such as the VGG model), and it achieves better results.

2.2.1. Layers

The CNN backbone uses cascaded CNN-Blocks to extract the features of the image, and it consists of two parts. The first part consists of separate CNN-Blocks to reduce the image resolution, a 3×3 convolutional layer, a BN (batch normalization) layer, the parallel structure of a 1×1 convolutional layer and a BN layer, and, finally, a rectified linear (ReLU) activation function by which the feature map is output. The second part consists of a cascade of four layers, each containing a different number of Resnet blocks and a global pooling layer, after which the final result is output.

The architecture of Layer is showed in Figure 2. The designs of Layer2, Layer3, and Layer4 are similar to that of Layer1; the difference is that the numbers of convolution channels are 64, 128, 256, and 512. Furthermore, Layer3 is three times the size of Layer1. Multiple CNN-Blocks are concatenated within each layer with similar structures. The difference is that, in the first CNN-Block, the strides of the 1×1 convolutional layer and the 3×3 convolution layer are two pixels. Therefore, compared to the input X, the output feature map $F(X)$ of the resolution decreases, and the number of channels changes. In the rest of the CNN-Blocks, the strides of the 1×1 convolutional layer and the 3×3 convolutional layer are one pixel, so the size and the number of channels of the feature map do not change.

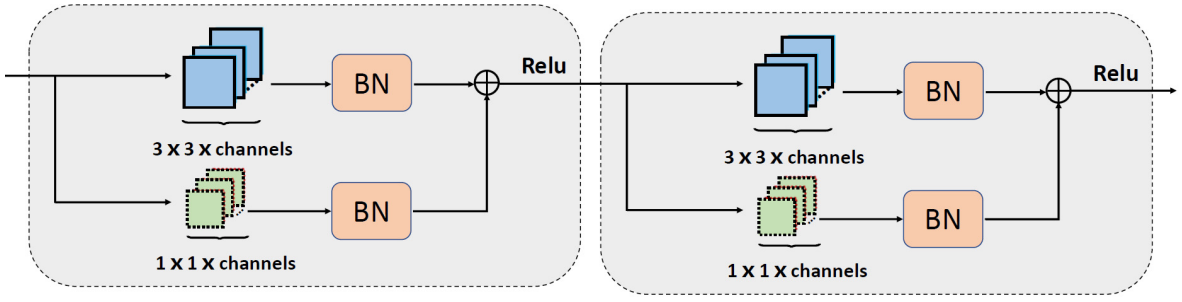


Figure 2. Network architecture of Layer1.

2.2.2. CNN-Block

The design of the CNN-Block is inspired by Resnet. Resnet learns feature information by residual connections with a basic residual module of $Y = \mathrm{ReLU}(F(X) + X)$. $F(X)$ is the residual mapping to be learned by the network , X is the input, and Y is the output. When the size of the feature map $F(X)$ decreases, it is necessary to adjust the size of X through a 1×1 convolutional layer to match $F(X)$. In this case, $Y = \mathrm{ReLU}(F(X) + G(X))$. $G(X)$

is the convolutional shortcut implemented in the 1×1 convolutional layer. The Resnet structure has the advantage that the multi-branch architecture makes the model an implicit ensemble of numerous shallower models [43]. In general, each block in Resnet has two roads of information flow, and, thus, for a Resnet model with n blocks, it can be represented as a collection of 2^n models.

A multi-branched structure facilitates the training of the model [43], so we use a parallel, two-branch structure based on a CNN model with fewer parameters. To control the number of parameters in the CNN backbone, we use a 1×1 convolutional layer as the second branch to construct the new block. The CNN-block of the model is:

$$Y = \mathrm{ReLU}(F(X) + G(X)), \quad (1)$$

where X and Y are the input and output feature maps of the CNN-Block module, $F(X)$ is the residual mapping to be learned by the network, and $G(X)$ is the 1×1 convolutional shortcut implemented in the convolutional layer. As shown in Figure 1, the Resnet block module contains two branches. After entering the CNN-Block, X is input to the 3×3 convolutional layer and the 1×1 convolutional layer for data processing. In the 3×3 convolutional layer, the input feature map X goes through the convolutional layer and a BN layer and becomes the output $F(X)$. In the 1×1 convolutional layer, the input feature map X goes through the convolutional layer and the BN layer successively and becomes the output $G(X)$. When the size of $F(X)$ decreases, the CNN-Block adds up the outputs of the two branches in terms of the pixels at corresponding positions, and, finally, the result is output after the ReLU activation function. The activated feature map Y is the output of this CNN-Block.

If the size of the feature map decreases, the 3×3 convolution kernel in the CNN-Block has a stride of two pixels and a padding of one pixel. When the size does not decrease, the 3×3 convolution kernel has a stride of one pixel and a padding of one pixel. The stride and channels of the 1×1 convolution kernel keep pace with the 3×3 convolution kernel in the same CNN-Block module.

We construct the CNN backbone model by stacking CNN-Blocks. From the same perspective as reported previously [43], for n CNN-Blocks, the CNN backbone can also be represented as a collection of 2^n models. In the CNN backbone, $n = 13$. Compared with Resnet, the CNN backbone implements a multi-branch architecture based on fewer convolutional layers, which enhances the extraction of the features of images. Finally, the output of Layer4 is subjected to a global average pooling operation by average pooling, then, the pooled feature map is flattened to a 512-dimensional feature vector.

2.3. Transformer Backbone

The CNN-Swin model borrows from the Swin transformer [32] model for image feature extraction. The CNN structure has a specific induction bias that makes it locally perceptive and capable of weight sharing, but the strict induction bias also limits its ability to extract image features to some extent. In comparison, the transformer structure has less induction bias, which makes it more similar to the attention of human vision when recognizing images. Therefore, it has better feature extraction capabilities. The Resnet-like hierarchical transformer structure can better extract the features of ship images.

The transformer model used in this work contains two parts: patch embedding and an encoder. After patch embedding, the input ship image passes through a multi-layer cascaded encoder module which extracts the features of the image. The encoder contains a norm operation, shifted, window-based, multi-head, self-attention (SW-MSA), and an MLP, as shown in Figure 1.

2.3.1. Patch Embedding

Assuming that H and W are the height and width of the image and C is the number of channels, the input ship image can be written as $x \in R^{H \times W \times C}$. For the input x, its

channel features are recombined to obtain a sequence of patches (x_p) and $x_p \in R^{N \times (P^2 \times C)}$. The image x is divided into a total of $N = \frac{H \times W}{P^2}$ patches, contained in x_p, where the size of x_p is $P \times P$. Then, x_p is flattened and mapped to a dimension of size D to finally obtain $x'_p \in R^{N \times D}$.

2.3.2. Encoder

The transformer encoder consists of two parts. Each part has two parallel branches, and the feature maps output from the two branches are added up linearly on a pixel-by-pixel basis. In the first part, the first branch has a linearly varying norm in series and shifted, window-based, multi-head, self-attention (SW-MSA). The second branch has a residual connection with the input. In the second part, the difference is that the first branch is a new, linearly varying norm and an MLP neural network.

The core operation of the transformer encoder is the SW-MSA. The SW-MSA is based on multi-head self-attention and builds shifted windows to further reduce the computational effort. In the multi-head self-attention, first, an inner product of the feature vector (x), encoded by patches with the matrix, is performed to obtain query (q) and key (k), i.e., $q = x \times W_Q$, $k = x \times W_K$, where W_Q and W_K are defined as learnable parameters in the encoder. Self-attention is actually a calculation of similarity between the query (q) and the key (k). Each query and key are obtained by the matrix inner product operation to obtain the similarity value, which is further normalized by the softmax function. The normalized similarity values represent the particular query and its corresponding key weights.

Meanwhile, an inner product operation of the feature vector (x) encoded by patches with the number of learnable parameter matrices W_V is performed, and the value (v) is obtained, i.e., $v = x \times W_V$. The similarity value output after the similarity calculation for each query and its corresponding key is used as the weighted value of its corresponding value.

Furthermore, multi-head self-attention performs the self-attention multiple times on the feature vector (x) encoded by patches, and each self-attention has its own W_Q, W_K, and W_V. The outputs are obtained after multiple different self-attention operations are linearly concatenated together, then, the results are obtained by a linear transformation of compressing scales.

Global self-attention has a high computational complexity [32]. To reduce the complexity, the SW-MSA restricts the self-attention to be computed only within the divided windows. The window divides the whole image in a non-overlapping manner, and each window contains $M \times M$ blocks. Additionally, to solve the problem of information non-flow between windows, the SW-MSA shifts the windows and proposes a batch computation approach by cyclic shifting.

When shifting the window partition, a portion of the window no longer has the size $M \times M$. Therefore, the batch computation approach splices the unappropriated sub-window partitions into a single $M \times M$-sized window. Then, the spliced and unspliced windows with sizes of $M \times M$ are subjected to the in-window self-attention computation. In computing the self-attention, the model uses the relative position encoding to compute the similarity [44–47]. The formula for calculating self-attention with the addition of a relative position encoding is as follows:

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{Q \times K^T}{\sqrt{d_k}} + B\right)V, \quad (2)$$

where B is the relative position code $B \in R^{M^2 \times M^2}$. M^2 denotes the number of patches in the window. $Q, K, V \in R^{M^2 \times d}$ is where d is the matrix dimension. $\sqrt{d_k}$ is the scaling factor used to avoid the effect of variance from the dot product, and k is the number of queries.

The overall encoder formulas are as follows:

$$Z'_\ell = SW - MSA(LN(Z_{\ell-1})) + Z_{\ell-1}, \ \ell = 1, 2, \cdots L, \quad (3)$$

$$Z_\ell = MLP(LN(Z'_\ell)) + Z'_\ell, \ \ell = 1, 2, \cdots L, \quad (4)$$

where Z'_{ℓ} and Z_{ℓ} denote the output of the ℓth encoder from the SW-MSA and MLP, respectively. Z'_{ℓ} and Z_{ℓ} together form a transformer encoder with a total of L layers.

2.4. Multi-Layer Perceptron (MLP) Neural Network

The two feature vectors with good local features extracted by the CNN backbone and transformer backbone are $feature\ vector1$ and $feature\ vector2$, respectively. To further fuse the feature information, we perform a pixel-by-pixel summation operation on the feature vectors of both. To accomplish this operation, we use a fully connected layer to make the dimension of $feature\ vector1$ extracted by the CNN backbone consistent with that of $feature\ vector2$, obtaining $feature\ vector1x$. The information fusion operation is as follows:

$$\text{Fused}(feature\ vector)_{1\times q} = feature\ vector1x_{1\times q} + feature\ vector2_{1\times q}, \tag{5}$$

where q represents the dimension of the feature vectors, and q is 1024. The fused feature vectors are processed using an MLP neural network.

The MLP neural network consists of three layers: an input layer, hidden layer, and output layer. The fused feature vectors containing a large amount of information are mapped into seven categories of ship image in the ship dataset by the MLP neural network. Assuming that the lth layer has m^l neurons, the input vector can be represented as:

$$x^{l-1} = \left[x_1^{l-1}, x_2^{l-1}, \ldots, x_{m^l}^{l-1}\right]^T, \tag{6}$$

where x^{l-1} denotes the output of the feature vector from the $(l-1)$th layer. The connection weight matrix between the $(l-1)$th layer neuron and the layer l layer neuron is as follows:

$$W^l = \begin{bmatrix} w_{11}^l & w_{12}^l & \cdots & w_{1m^{l-1}}^l \\ w_{21}^l & w_{22}^l & \cdots & w_{2m^{l-1}}^l \\ \vdots & \vdots & \ddots & \vdots \\ w_{m^l1}^l & w_{m^l2}^l & \cdots & w_{m^lm^{l-1}}^l \end{bmatrix}, \tag{7}$$

where w_{jk}^l denotes the connected weights between the kth neuron in the $(l-1)$th layer and the jth neuron in the lth layer.

The bias vector of the neuron in the lth layer has the following bias vector:

$$b^l = \left[b_1^l, b_2^l, \ldots, b_{m^l}^l\right]^T. \tag{8}$$

Then, forward propagation of the three-layer MLP neural network is performed as follows:

$$x^1 = \text{ReLU}\left(W^{(1)} \bullet x^0 + b^{(1)}\right), \tag{9}$$

$$x^{l-1} = \left[x_1^{l-1}, x_2^{l-1}, \ldots, x_{m^l}^{l-1}\right]^T \tag{10}$$

where ReLU is the activation function used for the neuron in each layer. The MLP neural networks use softmax functions to normalize the output of the three-layer MLP and obtain the final output value z, where z denotes the prediction probability of each class. The value z satisfies the properties of a probability distribution. For i categories, $z_i \in (0,1)$ and $\sum_i z_i = 1$, by selecting the node with the highest prediction probability, the model can obtain the final category of the classification.

3. Experiment

3.1. Datasets

To evaluate our proposed model, we conducted extensive experiments from two perspectives. First, we used an open, high-resolution ship dataset named FGSC-23 [36]

and a military ship dataset for image classification. To further explore the ability of the CNN-Swin model, we conducted extensive experiments on the two most widely used, oriented ship detection datasets, namely, HRSC2016 [37] and FAIR1M [38]. The results showed that the CNN-Swin model has great potential as a backbone network in ship image classification and detection.

3.1.1. FGSC-23 Dataset

FGSC-23 is a high-resolution, optical, remote sensing, ship target, fine-identification image dataset. It contains 23 categories of ship sub-targets and 4052 ship sample slices. Each target is given a category label, an aspect ratio label, and a distribution direction label. We only used the category label for image classification. In contrast to the existing optical, remote sensing, ship-target-recognition image datasets, the FGSC-23 dataset has diverse image scenes, fine category differentiation, and complete labeling. The images of the FGSC-23 dataset are mainly derived from publicly available Google Earth data and Gaofen-2 (GF-2) satellite panchromatic remote sensing images, and some ship image slices were derived from the publicly available HRSC2016 [37] ship target detection dataset with 0.4 to 2 m image resolution.

3.1.2. Military Ship Dataset

The FGSC-23 dataset is not comprehensive and does not contain the military class of ships for image classification. Currently, there is no large-scale, standard, open-source ship dataset that contains military-class ships because internet ship images are limited in quantity, and most of them are low in quality. We collected open-source images and compiled a dataset containing seven categories of military ship image. Some samples from the military ship dataset are shown in Figure 3. The specific categories of the ship images are shown in Table 1.

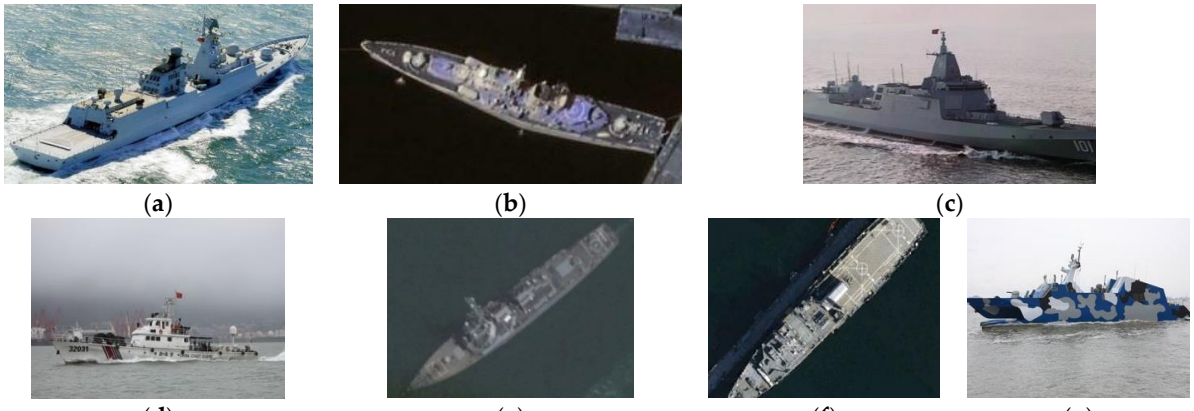


Figure 3. Some samples of the military ship dataset: (**a**) Type-054 frigate, (**b**) Sovremenny class destroyer, (**c**) Type-055 destroyer, (**d**) Surveillance boat, (**e**) Arleigh Burke class destroyer, (**f**) landing vessel, and (**g**) missile boat.

Table 1. Number of categories in military ship dataset.

Category of Ships	Number of Images
Type-054 frigate	404
Type-055 destroyer	409
Sovremenny class destroyer	420
Surveillance boat	397
Arleigh Burke class destroyer	445
Landing vessel	495
Missile boat	403

As shown in Figure 3, the ship images in the military dataset we collected contained different views, including bird's-eye views, side views, and satellite panchromatic remote sensing images of the ships. By further analyzing the samples of the seven categories used for ship image classification, it was discovered that, regardless of the orientation of the ship relative to the camera, the differences between the seven categories were not the hull structures but the ship equipment and superstructures.

Taking the Arleigh Burke class destroyer and Sovremenny class destroyer as examples: the Sovremenny class destroyer has a large, enclosed mainmast at the aft edge of the tall forward superstructure with a unique, large top plate with features such as an air search radar antenna mounted on top. Furthermore, a large single smokestack, square in appearance, is located aft of the middle of the ship. A framed mizzen mast is located behind the smokestack, and a helicopter hangar is located below it. The small flight deck is positioned slightly higher, in front of the stern anti-aircraft missile launchers. In comparison, in the Arleigh Burke class destroyer, the two longitudinal chimneys, the rear irradiating radar, and the MK-15 close-in defense system are mounted in a stepped sequence along an axis running down the center of the ship. The destroyer is also equipped with an aft hangar, with the end of the second stack structure attached to the hangar structure. Although the two are highly similar in hull architecture, there are still differences in the superstructures and equipment. Therefore, the classification of the ship image should focus on fine-grained features, such as superstructures.

3.1.3. HRSC2016

HRSC2016 was extracted from Google Earth, published by NWI in 2016, and annotated using the oriented bounding box (OBB) annotation format. The dataset contains a total of 1061 images. It is divided into groups of 436, 181, and 444 images for the training, validation, and test sets, respectively. There are three major categories and 27 minor categories, with a total of 2976 target annotations in the dataset. The image resolution is in the range of 300 $\times$ 300 to 1500 $\times$ 900.

3.1.4. FAIR1M

The FAIR1M dataset contains more than one million instances and more than 40,000 high-resolution, remote sensing images for fine-grained object recognition. The remote sensing images in the dataset have a resolution of between 0.3 and 0.8 m from different platforms and were obtained from many countries and regions. All objects in the FAIR1M dataset are annotated with respect to five categories and 37 subcategories by oriented bounding boxes.

Compared with existing detection datasets that are dedicated to ship detection, the FAIR1M dataset has four particular characteristics: (1) It is much larger than other existing ship detection datasets, both in terms of the number of instances and the number of images. (2) It provides richer, fine-grained category information for objects in remote sensing images. (3) It contains geographic information, such as latitude, longitude, and resolution attributes. (4) It provides better image quality due to the use of a careful data cleaning procedure.

The dataset subdivides the targets into categories, including 37 fine-grained categories. Of these, ships are subdivided into nine categories: liquid cargo ships, dry cargo ships, fishing ships, cruise ships, tugboats, engineering ships, motorboats, warships, and other categories. We only used the ship data in the FASIR1M dataset to verify the model we proposed.

3.2. Experiments and Results

3.2.1. Experiment Details and Basic Results of CNN-Swin Model

Table 2 shows the details of the architecture under the ship image dataset with a resolution of 224 $\times$ 224. The input images were processed in parallel by the CNN backbone and the transformer backbone to output two feature vectors containing large amounts of information. The model then took the two output feature vectors through a pixel-by-pixel summation process and output them. The output feature vectors were sequentially input

into the three layers of the MLP neural network, i.e., the input layer, the hidden layer with 128 neurons, and then the output layer, where the softmax function finally output the results.

Table 2. Architectural details of model.

Output Size		CNN Backbone	Transformer Backbone	
128×128	CNN-Block	$\begin{pmatrix} 3\times3, 64, \text{ stride} = 2, \text{ padding} = 1 \\ 1\times1, 64, \text{ stride} = 2 \end{pmatrix} \times 1$	LN	Patch embeddings
56×56	Layer1	$\begin{pmatrix} 3\times3, 64, \text{ stride} = 2(\text{or } 1), \text{ padding} = 1 \\ 1\times1, 64, \text{ stride} = 2(\text{or } 1) \end{pmatrix} \times 2$	concat 4×4, 128-d, LN $\begin{pmatrix} \text{window size} = 7\times7 \\ \text{dim} = 1286, \text{ head} = 4 \end{pmatrix} \times 2$	Stage1
28×28	Layer2	$\begin{pmatrix} 3\times3, 128, \text{ stride} = 2(\text{or } 1), \text{ padding} = 1 \\ 1\times1, 128, \text{ stride} = 2(\text{or } 1) \end{pmatrix} \times 2$	concat 2×2, 256-d, LN $\begin{pmatrix} \text{window size} = 7\times7 \\ \text{dim} = 256, \text{ head} = 8 \end{pmatrix} \times 2$	Stage2
14×14	Layer3	$\begin{pmatrix} 3\times3, 256, \text{ stride} = 2(\text{or } 1), \text{ padding} = 1 \\ 1\times1, 256, \text{ stride} = 2(\text{or } 1) \end{pmatrix} \times 8$	concat 2×2, 512-d, LN $\begin{pmatrix} \text{window size} = 7\times7 \\ \text{dim} = 512, \text{ head} = 16 \end{pmatrix} \times 8$	Stage3
7×7	Layer4	$\begin{pmatrix} 3\times3, 512, \text{ stride} = 2(\text{or } 1), \text{ padding} = 1 \\ 1\times1, 512, \text{ stride} = 2(\text{or } 1) \end{pmatrix} \times 2$	concat 2×2, 1024-d, LN $\begin{pmatrix} \text{window size} = 7\times7 \\ \text{dim} = 1024, \text{ head} = 32 \end{pmatrix} \times 2$	Stage4
1×1		Average Pooling		
Fully connected layers		(1024,128), (128, num_class)		

Of the parameters (m × n, d) × k in the convolution layer, d denotes the depth of the convolution layer, m × n denotes the convolution kernel size, and k denotes that the (m × n, d) cascade is stacked k times. Stride = 2 (or 1) denotes that the stride was 2 (or 1) in the first CNN-Block of the layer, and stride = 1 in the other CNN-Blocks of the layer.

Concat n × n denotes a concatenation of n × n neighboring features in a patch. This operation results in the downsampling of the feature map by a rate of n. A window size of 7×7 denotes a multi-head self-attention module with window size of 7×7.

To verify the feature extraction ability, we performed experiments with the CNN-Swin model on the military ship dataset, the number of categories of which is smaller than that of the FGSC-23 dataset. The use of fewer categories provided an intuitive and clear confusion matrix and t-distributed, stochastic-neighbor-embedding (t-sne) clusters [48] for verifying the CNN-Swin model's performance.

The computational platform had an Inter@Core i7-10875H CPU, and the GPU was a single NVIDIA GeForce RTX 2080. The programming language was Python, version 3.8, and the deep learning framework was PyTorch 1.7.0. The training and validation sets were divided with a ratio of 7:3. The number of epochs was 100, and the batch size was 16. To further reduce the computational cost and overfitting issues, the CNN-Swin model used an early stop, and the number of epochs was reduced to 82. The model was trained iteratively using Adam as the optimizer. The learning rate was 0.01, and the weight was set to 0.001 using L2 regularization. After pre-processing, the resolution of the input images was adjusted to 224×224 with the resize function in PyTorch, which was the size of the natural scene images, after which they were input into the model for training.

The confusion matrix was mainly used to represent the classification accuracy and visualize the classification performance of the algorithm. The horizontal axis is the prediction result, and the vertical axis is the true result. The darkest colored areas of the confusion matrix were concentrated along the diagonal line, which showed the success of the classification. The confusion matrix of the CNN-Swin model on the military ship dataset is shown in Figure 4.

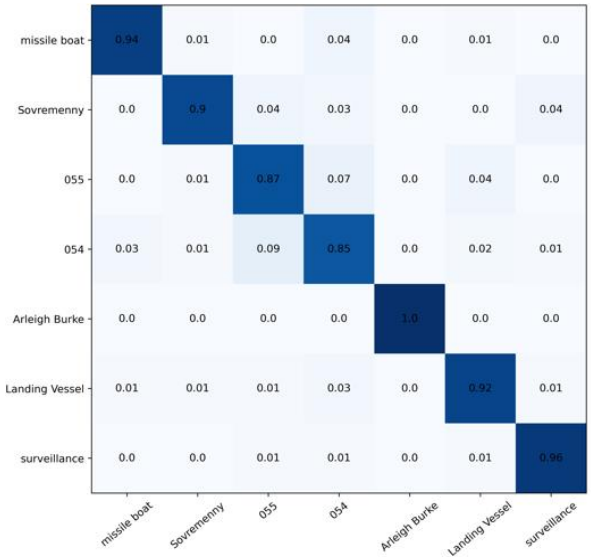


Figure 4. Confusion matrix of ship dataset.

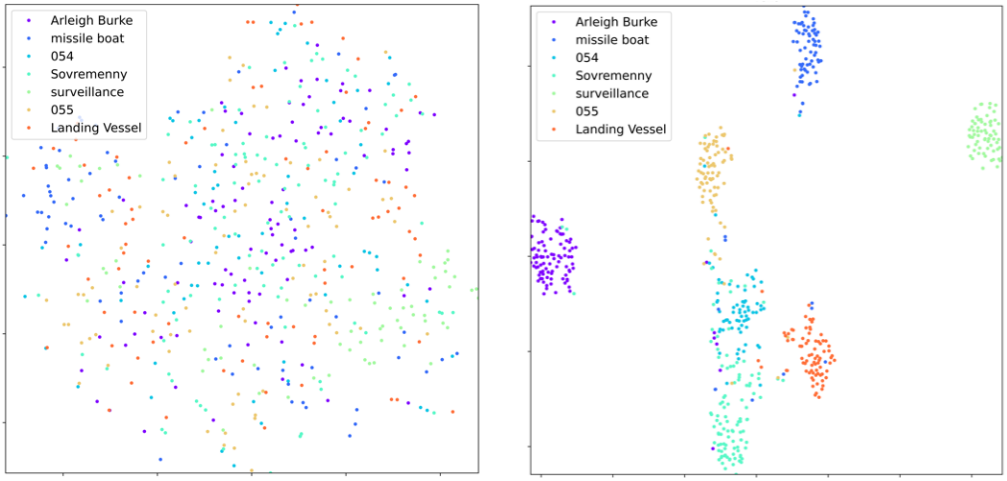
The confusion matrix of ship dataset showed two points:

1. The model was trained well overall. The Arleigh Burke class destroyers had the highest accuracy of 100%. The missile boat, landing vessel, and surveillance boat had accuracies higher than 90%. In general, the model had a high classification accuracy for the seven categories of ship image, demonstrating the reliability and robustness of the model with ship images as the objects;
2. Type-055 destroyers and Type-054 frigates had poor classification results, with accuracy rates of 0.87% and 0.85%, respectively, where 9% of Type-054 frigates were misclassified as Type-055 destroyers by the model, and 7% of Type-055 destroyers were misclassified as Type-054 frigates by the model. The reason was that the two are extremely similar in hull structure and overall superstructure, differing only partially in the superstructures. Specifically, in addition to the main bridge, the Type-055 destroyer has an integrated mast that is nearly the same height as the main bridge; the Type-054 frigate has a lower mast arranged at the rear and a radar with a spherical outer cover on top. In addition, the stern of the Type-054 frigate has a helicopter hangar, while the Type-055 destroyer does not, which results in some subtle differences in the superstructures at the stern. Compared to the more evident, fine-grained features, such as the superstructures in the other categories, the fine-grained feature differences between Type-055 destroyers and Type-054 frigates are not significant enough to give rise to a 7–9% classification error rate.

To further analyze the effectiveness of the model on the ship image dataset, the model was used to perform a t-sne [48] clustering analysis on the valid set. The t-sne clustering analysis chart in Figure 5 shows that the chart of the untrained sample set presents a scattered and disorganized point distribution without a good clustering effect. In comparison, the chart of the CNN-Swin model presents a good point distribution and basically includes seven clusters. The superiority of the model in terms of its classification effectiveness is visualized by Figures 4 and 5.

Specifically, the clustering effects of each category were as follows: The points of the Arleigh Burke class destroyer, missile boat, and surveillance boat were clustered completely. The points of the Type-054 frigate and Type-055 destroyer were mixed with each other, showing an average classification effect. A small portion of the points of the Sovremenny class destroyer was mixed with the points of the Type-054 frigate and the Type-055 destroyer. The t-sne clustering and confusion matrix were generally consistent and demonstrated the model's effectiveness for the analysis of specific samples. The overall analysis of the

classification effect of the model validated the effectiveness of the CNN-Swin model on the ship image dataset.



(**a**) (**b**)

Figure 5. t-sne clustering analysis on the military ship dataset: (**a**) sample of untrained dataset and (**b**) sample of dataset trained by CNN-Swin model.

3.2.2. Performance Comparison for Image Classification

This section focuses on the performance comparison of the model with two kinds of state-of-the-art approach. The first section compares the performance of the model with CNN models, and the second section compares the performance of the model with transformer models. All the approaches were compared for the FGSC-23 dataset and our ship dataset. The results in Table 3 show that the proposed model achieved the best results of all the algorithms.

For the state-of-the-art approaches using CNNs, we compared the classical models from recent years to the newer-model Regnet. For the state-of-the-art approaches using transformers, we compared the ViT structure, an improved model with the ViT structure (CaiT), and the Swin transformer model borrowed by the CNN-Swin model.

The results shown in Table 3 show that the CNN-Swin model constructed for the features of ship images achieved high effectiveness on the FGSC-23 dataset and our military ship dataset. For the same resolution of 224 $\times$ 224, the Top-1 accuracy rates of the CNN-Swin model were 3.8–8.5% and 2.4–4.2% higher than those of the Swin transformer model for our ship dataset and the FGSC-32 dataset, respectively. Furthermore, the model yielded 5.5–15.2% and 4.8–6.9% improvements in the Top-1 accuracy compared to the approaches using CNNs on our ship dataset and FGSC-23 dataset, respectively. With a resolution of 384^2, the model we proposed also outperformed other state-of-the-art approaches.

The comparison with other state-of-the-art approaches highlighted the good feature extraction capability of the CNN-Swin model proposed in this paper on the ship image datasets. The specific design and improvement for feature extraction of ship images are effective and meaningful.

We also used five-fold cross-validation for the CNN-Swin model to validate the model stability. In the five-fold cross-validation, the initial sample was split into five subsamples. One subsample was retained to validate the model, while the other four samples were used for training. The cross-validation was repeated five times, once for each subsample, and the results were averaged over five times or using other combinations to end up with a single estimate.

Table 3. Comparison with state-of-the-art approaches for image classification.

Model	FGSC-32		Our Military Ship Dataset	
	Resolution	Top-1 Accuracy	Resolution	Top-1 Accuracy
Densenet-121 [8]	224^2	84.0%	224^2	76.7%
Efficientnet [12]	224^2	85.2%	224^2	81.1%
GoogLeNet v4 [6]	224^2	84.2%	224^2	81.6%
Resnet-18 [7]	224^2	81.9%	224^2	84.2%
VGG-19 [2]	224^2	81.8%	224^2	84.5%
Regnet [2020] [40]	224^2	86.1%	224^2	86.4%
ViT-tiny [25]	224^2	84.3%	224^2	81.4%
ViT-small [25]	224^2	86.3%	224^2	85.1%
ViT-base [25]	224^2	87.9%	224^2	87.2%
ViT-base [25]	384^2	88.1%	384^2	87.5%
CaiT [2020] [32]	224^2	88.0%	224^2	86.5%
Swin-tiny [2021] [33]	224^2	86.7%	224^2	83.4%
Swin-base [2021] [33]	224^2	88.5%	224^2	88.1%
Swin-base [2021] [33]	384^2	89.1%	384^2	88.4%
CNN-Swin Model	**224^2**	**90.9% ± 0.1%**	**224^2**	**91.9% ± 0.1%**
	384^2	**91.4% ± 0.1%**	**384^2**	**92.1% ± 0.1%**

The variance of the results was controlled to within 0.1%. This also proved that the 91.9% and 90.9% accuracy rates achieved by CNN-Swin model were not a coincidence but had relatively good stability.

Figure 6 shows the confusion matrices for the state-of-the-art models for the military ship dataset. The horizontal axis is the prediction result, and the vertical axis is the true result. The darkest color in the confusion matrices was concentrated along the diagonals, and the lighter colors appeared in the other the locations. The confusion matrix of Desenet-121 had the most disorganized color distribution and was less effective in classification. The confusion matrices of Regnet and Swin-base had more concentrated colors and the darkest colors on the diagonal, which showed good effectiveness in classification. This was consistent with the Top-1 accuracy results obtained from our experiments.

The specific analysis of the classification of each approach for the samples of the seven categories is discussed as follows. For the Alibek-class destroyers, the Top-1 accuracy of the multi-branch structures Resnet, GoogLeNet, and Regnet was in the range of 68–75%, which showed less accuracy and effectiveness. However, the Swin transformer was more effective for feature extraction for this category of ships, and the Swin-base model had an accuracy of 95%. The fusion of the Swin model based on the multi-branch CNN had a good complementary effect on the extraction of features.

The confusion matrix of the CNN-Swin model in Figure 6, with the colors nearly completely concentrated on the diagonal, shows the good classification performances. In comparison, other state-of-the-art approaches had average classification results and some generalizability limitations.

3.2.3. Performance Comparison for Ship Detection

This section focuses on the performance comparisons of different model backbones with three kinds of state-of-the-art method for ship detection. The first section compares the performance of the backbone on the HRSC2016 dataset, and the second section compares the performance of the backbone on the ship data in the FAIR1M dataset.

For the ablation study, we considered three typical detection frameworks: Axis Learning [49], Rotation-Sensitive Regression Detector (RRD) [50], and Sparse Label Assignment (SLA) [51]. Axis Learning is a one-stage, anchor-free method with a new aspect-ratio-aware orientation centerness method for large-aspect-ratio detection. RRD uses two network branches with different designs: a regression branch extracts rotation-sensitive features, and a classification branch extracts rotation-invariant features. SLA is a sparse label-assignment

strategy to select high-quality sparse anchors based on the posterior intersection over union (IoU) of detections. In addition, we used three different backbones for each detection framework: Resnet, VGG, and ViT. For these different backbones, we utilized the same settings as those of the CNN-Swin backbone: the Adam optimizer, an initial learning rate of 0.001, a weight decay of 0.05, and a batch size of 16. For the different detection frameworks, we used the settings mentioned in their respective papers. The comparisons were conducted by changing only the backbones and leaving the other settings unchanged.

The mean average precision (mAP07) was calculated based on the visual object classes (VOC) 07 method which was proposed by the Pascal VOC Challenge [52]. As shown in the Table 4, on the HRSC2016 dataset, the CNN-Swin backbone we proposed surpassed all the other backbones from different state-of-the-art methods, including Axis Learning [49], RRD [50], and SLA [51]. With the CNN-Swin backbone we proposed, Axis Learning, RRD, and SLA obtained 67.23%, 90.67%, and 90.56% mAP values, respectively. It is surprising that by using the CNN-Swin backbone, all the methods outperformed the other backbones, even with the higher resolution. Figure 7 shows some results from the HRSC2016 dataset.

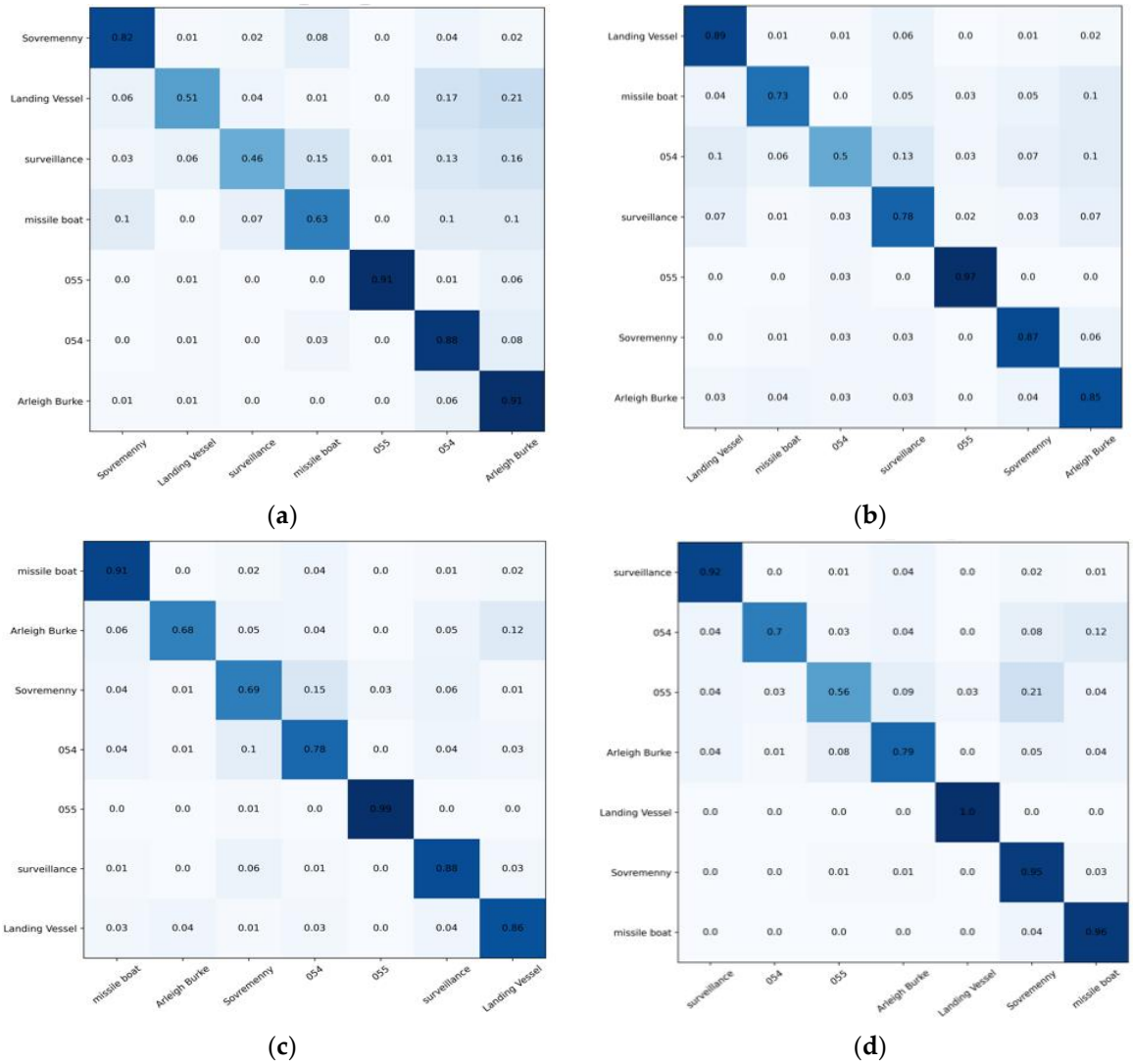


Figure 6. *Cont.*

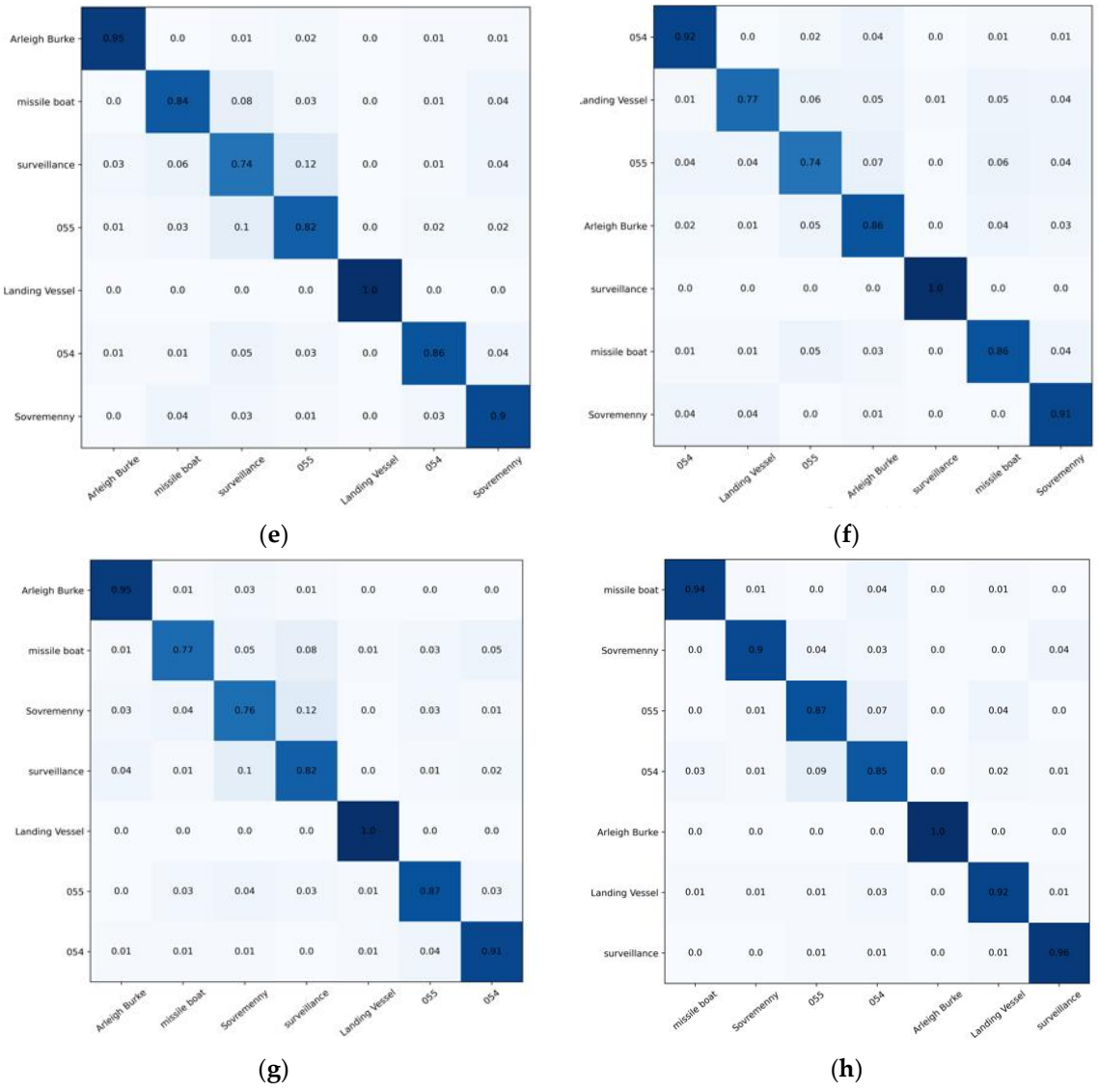


Figure 6. Confusion matrices of state-of-the-art approaches for military ship dataset: (**a**) Densenet-121, (**b**) Efficientnet, (**c**) Resnet-18, (**d**) Regnet, (**e**) ViT-base, (**f**) CaiT, (**g**) Swin-base, and (**h**) CNN-Swin.

Table 4. Performance comparison of different methods with different backbones for ship detection in the HRSC2016 dataset.

Methods	Backbone	Input Size	HRSC2016 mAP(07)
Axis Learning [49]	Resnet101	800 × 800	65.98
	Resnet50	800 × 800	64.04
	ViT-S	384 × 384	66.72
	CNN-Swin	**384 ×384**	**67.23**
RRD [50]	Resnet101	768 × 768	89.51
	VGG16	384 × 384	84.30
	ViT-S	384 × 384	86.79
	CNN-Swin	**768 ×768**	**90.67**
SLA [51]	Resnet101	384 × 384	87.14
	Resnet50	768 × 768	89.51
	ViT-S	384 × 384	88.23
	CNN-Swin	**384×384**	**90.56**

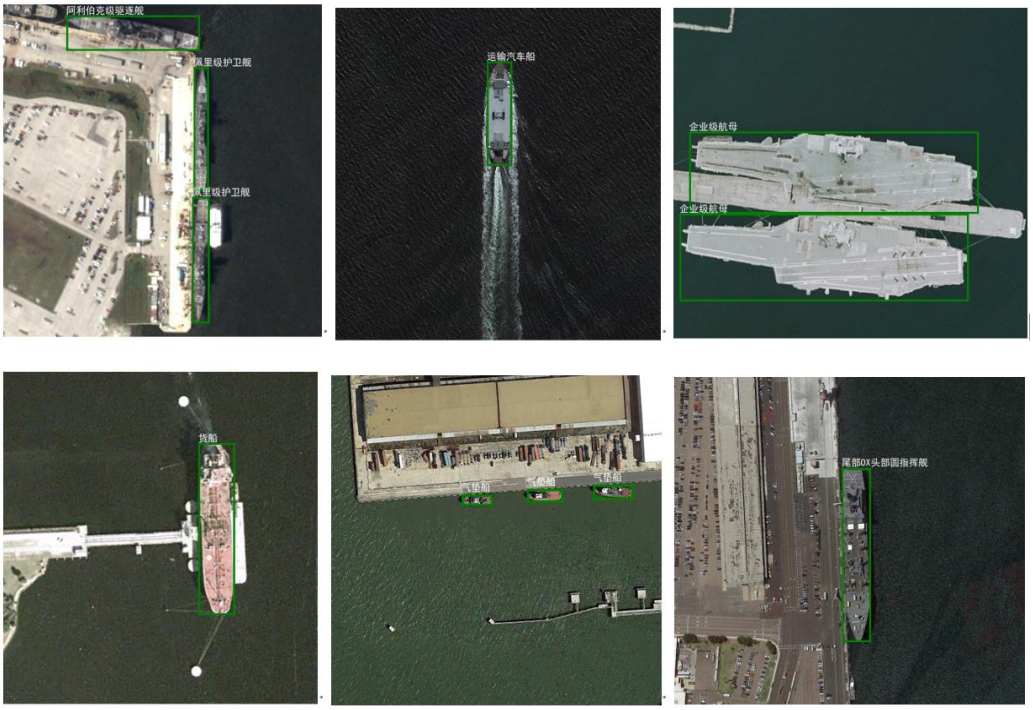


Figure 7. Some of the detection results obtained from the HRSC2016 dataset with our backbone in different state-of-the-art methods.

Table 5 shows the performance with the CNN-Swin as the basic backbone in the three different state-of-the-art methods for the FAIR1M dataset. Due to the high number of categories in the FAIR1M that we did not need, we only used nine categories of ship data from this dataset to verify the performance of the CNN-Swin backbone for feature extraction. The results showed that, with the CNN-Swin backbone, a higher head mAP could be achieved, with values of 38.52%, 39.77%, and 43.16%, respectively. Figure 8 shows some results from the FAIR1M dataset.

Table 5. Performance comparison of different methods with different backbones for ship detection in the ship data of the FAIR1M dataset.

Methods	Backbone	Input Size	Ship Data of FAIR1M mAP(07)
Axis Learning [49]	Resnet101	800 × 800	33.82
	Resnet50	800 × 800	35.72
	ViT-S	384 × 384	36.81
	CNN-Swin	**384 ×384**	**38.52**
RRD [50]	Resnet101	768 × 768	36.53
	VGG16	384 × 384	35.64
	ViT-S	384 × 384	37.75
	CNN-Swin	**768 ×768**	**39.77**
SLA [51]	Resnet101	384 × 384	41.94
	Resnet50	768 × 768	40.31
	ViT-S	384 × 384	42.23
	CNN-Swin	**384×384**	**43.16**

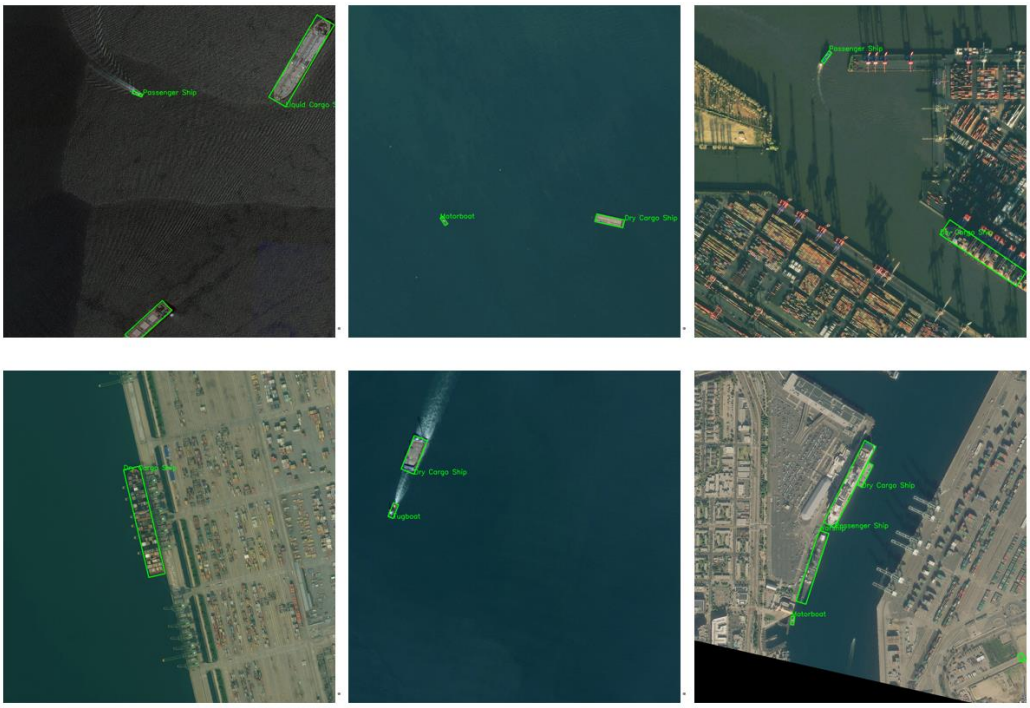


Figure 8. Some of the detection results obtained from part of FAIR1M dataset with our backbone in different state-of-the-art methods.

Figures 7 and 8 show that different methods with the CNN-Swin backbone achieved high detection performances. The CNN-Swin model was effective in ship image detection in different ship remote sensing datasets, and its feature extraction performance also demonstrated great potential as the backbone network for a range of downstream tasks related to ship image detection.

3.2.4. Ablation Studies

We performed an ablation study to highlight the contributions of each neural stream to the final classification result. The results are shown in Table 6, where Acc^{CNN} and Acc^{Tran} denote the accuracy of the CNN and transformer branches, respectively. First, the accuracy of the CNN-Swin model with the combined CNN and transformer increased by 3.9–8.7% compared to that of its individual branches. The CNN-Swin model was also compared with an ensemble model that combined the outputs of the CNN and the transformer. For a fair comparison, the same data augmentation and training epochs were used to train ResNet-101, and it was combined with the ViT-base model to form an ensemble model. The results are reported in Table 6. The accuracies of the CNN branch, the transformer branch, and the CNN-Swin model reached 84.8%, 88.0%, and 91.9%. In contrast, the ensemble model (ViT-base + ResNet-101) achieved 89.7% accuracy, which was 2.2% lower than that of CNN-Swin (91.9%).

Table 6. Performance comparison of single branch.

Model	Acc^{CNN}	Acc^{Tran}	Acc^{All}
ViT-base	—	87.2%	87.2%
Resnet-101	82.1%	—	81.8%
ViT-base + Resnet-101	82.1%	87.2%	89.7%
CNN-Swin	**83.2%**	**88.0%**	**91.9%**

The results of the ablation study proved that each neural stream in the CNN-Swin model contributed well to the final classification. However, the reason that the model

worked better when the two branches, i.e., the CNN and the transformer, were fused was not explained clearly by the experiments in Table 6.

Therefore, the class activation maps of each branch in the CNN-Swin model were visualized to explain why the proposed model was better than the others. To clearly reveal the class activation maps, the CNN branch used the Grad-CAM [53], and the transformer branch used attention rollout [54]. The feature maps of the last layer of the CNN branch and the last layer of the transformer branch were the inputs of the class activation maps.

As shown in Figure 9, the CNN branch concentrated more on the local regions, which is clearly demonstrated in Figure 9a–c. Compared with the CNN branch, the transformer branch of the CNN-Swin model tended to activate larger regions rather than local areas, suggesting enhanced long-distance feature dependencies, which are clearly demonstrated in Figure 9d–f.

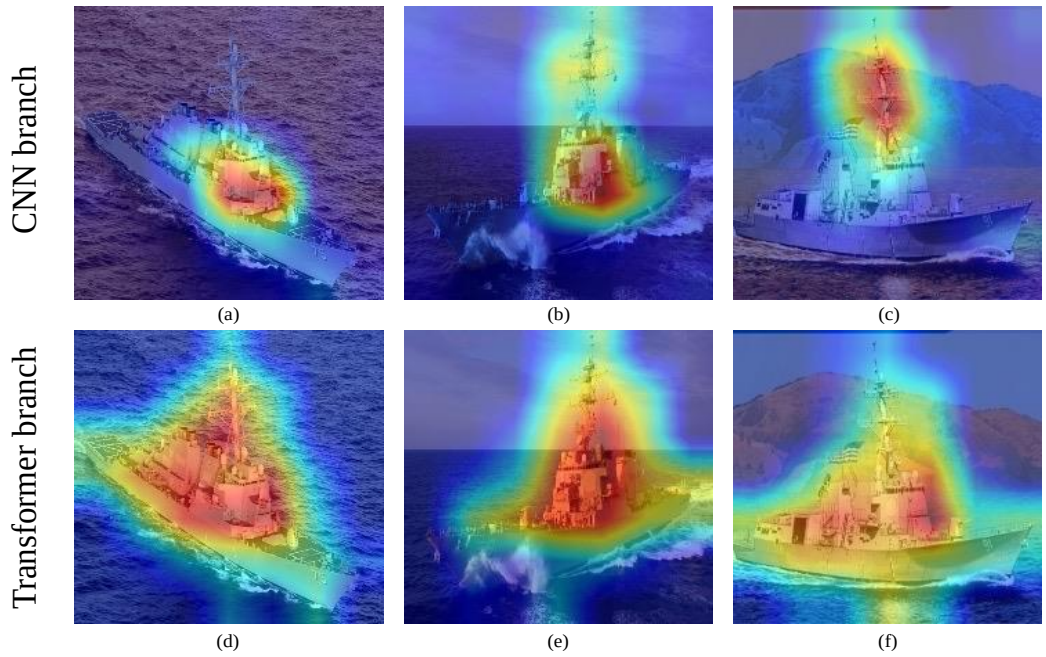


Figure 9. Class activation maps: (**a**) CNN branch in Arleigh Burke-class Destroyer, (**b**) CNN branch in Arleigh Burke-class Destroyer, (**c**) CNN branch in Arleigh Burke-class Destroyer, (**d**) Transformer branch in Arleigh Burke-class Destroyer, (**e**) Transformer branchin Arleigh Burke-class Destroyer, (**f**) Transformer branch in Arleigh Burke-class Destroyer.

Due to the fine-detailed local features progressively provided by the CNN branch, the CNN-Swin retained important, detailed local features. Furthermore, the attention areas in Figure 9d–f are more complete, while the background is significantly suppressed, implying that the transformer branch had a higher discriminative capacity for the global feature representations in the CNN-Swin model. In contrast, the CNN branch activated discriminative local regions. The CNN-Swin model took advantage of global cues from the transformer branch and, thereby, performed better than other single models.

Specifically, Figure 9 reveals that the transformer branch focused more on the overall architectural features of the ship, and the CNN branch concentrated more on the local features, such as the superstructure, of the ship. The restriction of its receptive field limits its performance for extracting features of target ship. Additionally, the CNN mainly concentrates on the local, typical features of ship targets. Similarly, its local receptivity limits its performance. Considering the different active regions of the CNN and the transformer, our CNN-Swin model combined them to extract the different-grained features of ship targets. Finally, the performance of classification and detection demonstrated that our combined method is effective.

When a person judges the type of a ship, the characteristics of both the hull shape and the superstructure of the ship are combined to determine the type of ship. The CNN-Swin model incorporating a CNN and a transformer is in line with human visual common sense for ship images.

To further improve the performance of the CNN-Swin model and alleviate the overfitting issues, we used different data augmentations, such as RandomResizedCrop, RandomHorizontalFlip, RandomRotation, ShiftScaleRotate (random affine change), and HueSaturationValue (random changes of the hue and saturation values of the network). These data augmentation methods have already been used in recent ship detection methods [55,56].

Each technique was implemented in the PyTorch framework, which is widely used in deep learning. Finally, the CNN-Swin model with different data augmentation techniques was implemented, and the results are shown in Table 7.

Table 7. Performance comparison of different data augmentations in CNN-Swin model.

Data Augmentation Methods	Accuracy
RandomRotation	90.37%
RandomBrightnessContrast	90.59%
RandomHorizontalFlip	91.24%
ShiftScaleRotate	91.59%
HueSaturationValue	91.72%
RandomResizedCrop	91.77%
CNN-Swin without data augmentation	91.41%
CNN-Swin+Data Augmentation (Final)	**91.94%**

The experimental results showed that when using ShiftScaleRotate (random affine change), HueSaturationValue (random changes of the hue and saturation values of the network), and RandomResizedCrop, the CNN-Swin model yielded a certain degree of performance improvement. By combining various types of data augmentation technique, we finally selected three optimization methods that had positive effects on the CNN-Swin model for the ship dataset.

The CNN-Swin model achieved a 91.94% accuracy on the ship image dataset using three data augmentation tricks. Compared with the basic model without any data augmentation, the performance of the model after using aggressive data augmentation was improved.

To further explain the computational complexity, we performed experiments to explore the computational complexity of CNN-Swin model, and the results are in Table 8.

Table 8. Computational complexity analysis of in CNN-Swin model.

Model	Batchsize	Computational Time/h	Accuracy
VGG-16	8	2.38	83.9%
VGG-19	8	2.56	84.5%
CNN-Swin	8	2.66	**91.9%**
Swin+VGG-13	4	4.86	89.9%
Swin+Resnet-50	4	3.96	90.4%
CNN-Swin	4	**3.26**	**92.0%**

All models were trained 100 epochs to provide a fair comparison. In addition, for fair contrast, the batch size was not only kept the same between comparators, but was also adjusted by the GPU-limited performance, which was improved to the highest level of the computational platform.

As the results in the upper column of the table show, our CNN-Swin model can be trained equally in nearly the same time as the competitors VGG-19 and VGG-16. Although

the training computational time of CNN-Swin model is longer than others, the accuracy is much higher than VGG-16 and VGG-19. We believe that the high performance of CNN-Swin model makes it worth training for longer than others to a certain extent.

The lower rows of the table show that, with the concise CNN backbone we designed, our model not only trained faster than other CNN branches but also had higher accuracy.

4. Conclusions

By analyzing remote sensing ship images in detail, we found that the differences in ship images were concentrated in the ship equipment and superstructures. Therefore, ship image feature extraction requires more attention to be paid to fine-grained and coarse-grained features. To better extract the features of ship images, we applied a transformer structure based on self-attention for the first time in ship image classification and detection, and we proposed a CNN-Swin model that fused CNN and transformer structures. To exploit the good extraction effect of the CNN on the features of ship images and avoid overfitting due to the model fusion, we proposed the CNN-Block, a CNN feature extraction module with multiple branches in the CNN-Swin model. We also used a Resnet-like structure to construct a CNN backbone based on the CNN-Blocks.

Experimental results showed that the CNN-Swin model achieved a 90.9% accuracy on the FGSC-23 dataset and 91.9% accuracy on a military ship dataset. We also performed experiments with other state-of-the-art approaches. The results showed that, for the military ship dataset, the CNN-Swin model accuracy was 4.9% higher than that of Regnet (2020), the highest-accuracy CNN model, and 3.2% higher than the Swin transformer model (2021), the highest-accuracy transformer. Meanwhile, for the FGSC-23 dataset, the CNN-Swin model accuracy was 4.8% higher than that of Regnet and 2.4% higher than that of the Swin transformer model. In ship image classification and recognition, the CNN-Swin model showed good classification performances.

Moreover, the model applied a transformer structure with self-attention for the first time in this field, and its good feature extraction ability showed its great potential for a range of downstream tasks, such as ship detection. Therefore, the CNN-Swin model was used as the basic backbone to extract features with different state-of-the-art remote sensing detection methods. The Axis Learning, RRD, and SLA methods with the CNN-Swin backbone achieved higher mAP values than the other backbones, Resnet and ViT, on the open dataset HRSC2016 and part of the FAIR1M dataset. Although the CNN-Swin model we proposed achieved good results in ship classification and detection, its computational complexity increased to a certain extent. However, the novel CNN-Block was designed to decrease the computational complexity, and, finally, the CNN-Swin model was trained faster than other confused models. The feature extraction performance of CNN-Swin demonstrates its great potential as a backbone network for downstream tasks such as ship detection.

Author Contributions: Formal analysis, Q.X.; Funding acquisition, L.H.; Investigation, F.W.; Methodology, F.W.; Project administration, F.W.; Validation, F.W.; Visualization, F.W. and Y.Z.; Writing—original draft, F.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

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

References

1. Krizhevsky, A.; Sutskeve, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in neural information processing systems, Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1097–1105.
2. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. In Proceedings of the International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015.

3. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 8–10 June 2015; pp. 1–9.
4. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015.
5. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, S.; Wojna, Z. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016.
6. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A. Inception-v4, inception-resnet and the impact of residual connections on learning. In Proceedings of the AAAI Conference on Artificial Intelligence, Hilton San Francisco, CA, USA, 4–9 February 2017.
7. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
8. Huang, G.; Liu, Z.; Maaten, L.V.D.; Weinberger, K.Q. Densely connected convolutional net- works. In Proceedings of the IEEE conference on computer vision and pattern recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
9. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
10. Sandler, M.; Howard, A.G.; Zhu, M.; Zhmoginov, A.; Chen, L. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018.
11. Howard, A.; Sandler, M.; Chu, G.; Chen, L.; Chen, B.; Tan, M.; Wang, W.; Zhu, Y.; Pang, R.; Vasudevan, V. Searching for mobilenetv3. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019.
12. Tan, M.; Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–16 June 2019.
13. Lin, T.; Dollar, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 936–944.
14. Jeon, H.; Yang, C. Enhancement of Ship Type Classification from a Combination of CNN and KNN. *Electronics* **2021**, *10*, 1169. [CrossRef]
15. Li, J.; Ruan, J.; Xie, Y. Research on the Development of Object Detection Algorithm in the Field of Ship Target Recognition. *Int. Core J. Eng.* **2021**, *7*, 233–241.
16. Julianto, E.; Khumaidi, A.; Priyonggo, P.; Munadhif, I. Object recognition on patrol ship using image processing and convolutional neural network (CNN). *J. Phys. Conf. Ser.* **2020**, *1450*, 012081. [CrossRef]
17. Chen, Z.; Chen, D.; Zhang, Y.; Cheng, X.; Zhang, M.; Wu, C. Deep learning for autonomous ship-oriented small ship detection. *Saf. Sci.* **2020**, *130*, 104812. [CrossRef]
18. Zhao, S.; Xu, Y.; Li, W.; Lang, H. Optical Remote Sensing Ship Image Classification Based on Deep Feature Combined Distance Metric Learning. *J. Coast. Res.* **2020**, *102*, 82–87. [CrossRef]
19. Xu, C.; Yin, C.; Wang, D.; Han, W. Fast ship detection combining visual saliency and a cascade CNN in SAR images. *IET Radar Sonar Navig.* **2020**, *14*, 1879–1887. [CrossRef]
20. Gao, X. Design and Implementation of Marine Automatic Target Recognition System Based on Visible Remote Sensing Images. *J. Coast. Res.* **2020**, *115*, 277–279. [CrossRef]
21. Ren, Y.; Yang, J.; Zhang, Q.; Guo, Z. Multi-Feature Fusion with Convolutional Neural Network for Ship Classification in Optical Images. *Appl. Sci.* **2019**, *20*, 4209. [CrossRef]
22. Li, Z.; Zhao, B.; Tang, L.; Li, Z.; Feng, F. Ship classification based on convolutional neural networks. *J. Eng.* **2019**, *21*, 7343–7346.
23. Bi, F.; Hou, J.; Chen, L.; Yang, Z.; Wang, Y. Ship Detection for Optical Remote Sensing Images Based on Visual Attention Enhanced Network. *Sensors* **2019**, *10*, 2271. [CrossRef] [PubMed]
24. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
25. 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 16×16 words: Transformers for image recognition at scale. In Proceedings of the International Conference on Learning Representations, Vienna, Australia, 3–7 May 2021.
26. Touvron, H.; Cord, M.; Douze, M.; Massa, F.; Sablayrolles, A.; Jegou, H. Training data-efficient image transformers & distillation through attention. *arXiv* **2020**, arXiv:2012.12877.
27. Yuan, L.; Chen, Y.; Wang, T.; Yu, W.; Shi, Y.; Tay, F.E.; Feng, J.; Yan, S. Tokens- to-token vit: Training vision transformers from scratch on imagenet. *arXiv* **2021**, arXiv:2101.11986.
28. Chu, X.; Zhang, B.; Tian, Z.; Wei, X.; Xia, H. Do we really need explicit position encodings for vision transformers? *arXiv* **2021**, arXiv:2102.10882.
29. Han, K.; Xiao, A.; Wu, E.; Guo, J.; Xu, C.; Wang, Y. transformer in transformer. *arXiv* **2021**, arXiv:2103.00112.
30. Wang, W.; Xie, E.; Li, X.; Fan, D.; Song, K.; Liang, D.; Lu, T.; Luo, P.; Shao, L. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. *arXiv* **2021**, arXiv:2102.12122.

31. Heo, B.; Yun, S.; Han, D.; Chun, S.; Choe, J.; Oh, S.J. Rethinking Spatial Dimensions of Vision Transformers. *arXiv* **2021**, arXiv:2103.16302v2.
32. Touvron, H.; Cord, M.; Sablayrolles, A.; Synnaeve, G.; Jégou, H. going deeper with Image Transformers. *arXiv* **2021**, arXiv:2103.17239v2.
33. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. *arXiv* **2021**, arXiv:2103.14030v2.
34. Xu, X.; Feng, Z.; Cao, C.; Li, M.; Wu, J.; Wu, Z.; Shang, Y.; Ye, S. An Improved Swin Transformer-Based Model for Remote Sensing Object Detection and Instance Segmentation. *Remote Sens.* **2021**, *13*, 4779. [CrossRef]
35. Huang, B.; Guo, Z.; Wu, L.; He, B.; Li, X.; Lin, Y. Pyramid Information Distillation Attention Network for Super-Resolution Reconstruction of Remote Sensing Images. *Remote Sens.* **2021**, *13*, 5143. [CrossRef]
36. Yao, L.; Zhang, X.; Lyu, Y.; Sun, W.; Li, M. FGSC-23: A large-scale dataset of high-resolution optical remote sensing image for deep learning-based fine-grained ship recognition. *J. Image Graph.* **2021**, *26*, 2337–2345.
37. Liu, Z.; Wang, H.; Weng, L.; Yang, Y. Ship rotated bounding box space for ship extraction from high-resolution optical satellite images with complex back- grounds. *IEEE Geosci. Remote Sens. Lett.* **2016**, *13*, 1074–1078. [CrossRef]
38. Sun, X.; Wang, P.; Yan, Z.; Xu, F.; Wang, R.; Diao, W.; Chen, J.; Li, J.; Feng, Y.; Xu, T.; et al. FAIR1M:A Benchmark Dataset for Fine-grained Object Recognition in High-Resolution Remote Sensing Imagery. *arXiv* **2021**, arXiv:2103.05569.
39. Springenberg, J.T.; Dosovitskiy, A.; Riedmiller, M.A. Striving for Simplicity: The All Convolutional Net. *arXiv* **2014**, arXiv:1412.6806.
40. Han, D.; Yun, S.; Heo, B.; Yoo, Y. Rethinking channel dimensions for efficient model design. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 732–741.
41. Radosavovic, I.; Kosaraju, R.P.; Girshick, R.; He, K.; Dollar, P. Designing network design spaces. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 10428–10436.
42. Ding, X.; Zhang, X.; Ma, N.; Han, J.; Ding, G.; Sun, J. RepVGG: Making VGG-style convnets great again. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021.
43. Veit, A.; Wilber, M.J.; Belongie, S. Residual networks behave like ensembles of relatively shallow networks. In *Advances in Neural Information Processing Systems, Proceeding of the 30th International Conference on Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016*; Curran Associates Inc.: Red Hook, NY, USA, 2016; pp. 550–558.
44. Hu, H.; Zhang, Z.; Xie, Z.; Lin, S. Local relation networks for image recognition. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27–28 October 2019; pp. 3464–3473.
45. Hu, H.; Gu, J.; Zhang, Z.; Dai, J.; Wei, Y. Relation networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 3588–3597.
46. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* **2020**, *21*, 1–67.
47. Bao, H.; Dong, L.; Wei, F.; Wang, W.; Yang, N.; Liu, X.; Wang, Y.; Gao, J.; Piao, S.; Zhou, M.; et al. Unilmv2: Pseudo-masked language models for unified language model pre-training. In Proceedings of the International Conference on Machine Learning, Vienna, Austria, 12–18 July 2020; pp. 642–652.
48. Maaten, L. Accelerating t-SNE using tree-based algorithms. *J. Mach. Learn. Res.* **2014**, *15*, 3221–3245.
49. Xiao, Z.; Qian, L.; Shao, W.; Tan, X.; Wang, K. Axis learning for orientated objects detection in aerial images. *Remote Sens.* **2020**, *12*, 908. [CrossRef]
50. Zhong, B.; Ao, K. Single-stage rotation-decoupled detector for oriented object. *Remote Sens.* **2020**, *12*, 3262. [CrossRef]
51. Ming, Q.; Miao, L.; Zhou, Z.; Song, J.; Yang, X. Sparse Label Assignment for Oriented Object Detection in Aerial Images. *Remote Sens.* **2021**, *13*, 2664. [CrossRef]
52. Everingham, M.; Gool, L.; Williams, C.K.; Winn, J.; Zisserman, A. The Pascal Visual Object Classes (VOC) Challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [CrossRef]
53. Selvaraju, R.R.; Das, A.; Vedantam, R.; Cogswell, M.; Parikh, D.; Batra, D. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. *Int. J. Comput. Vis.* **2019**, *128*, 336–359. [CrossRef]
54. Abnar, S.; Zuidema, W. Quantifying Attention Flow in Transformers. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; pp. 4190–4197.
55. Zhu, M.; Hu, G.; Zhou, H.; Wang, S.; Feng, Z.; Yue, S. A Ship Detection Method via Redesigned FCOS in Large-Scale SAR Images. *Remote Sens.* **2022**, *14*, 1153. [CrossRef]
56. Li, L.; Jiang, L.; Zhang, J.; Wang, S.; Chen, F. A Complete YOLO-Based Ship Detection Method for Thermal Infrared Remote Sensing Images under Complex Backgrounds. *Remote Sens.* **2022**, *14*, 1534. [CrossRef]