applied sciences

Article

Classification and Object Detection of 360° Omnidirectional Images Based on Continuity-Distortion Processing and Attention Mechanism

Xin Zhang [1], Degang Yang [1,2], Tingting Song [1,*], Yichen Ye [3], Jie Zhou [1] and Yingze Song [1]

[1] College of Computer and Information Science, Chongqing Normal University, Chongqing 401331, China
[2] Chongqing Engineering Research Center, Educational Big Data Intelligent Perception and Application, Chongqing 401331, China
[3] College of Electronic and Information Engineering, Southwest University, Chongqing 400715, China
* Correspondence: ttsong@cqnu.edu.cn

Abstract: The use of 360° omnidirectional images has occurred widely in areas where comprehensive visual information is required due to their large visual field coverage. However, many extant convolutional neural networks based on 360° omnidirectional images have not performed well in computer vision tasks. This occurs because 360° omnidirectional images are processed into plane images by equirectangular projection, which generates discontinuities at the edges and can result in serious distortion. At present, most methods to alleviate these problems are based on multi-projection and resampling, which can result in huge computational overhead. Therefore, a novel edge continuity distortion-aware block (ECDAB) for 360° omnidirectional images is proposed here, which prevents the discontinuity of edges and distortion by recombining and segmenting features. To further improve the performance of the network, a novel convolutional row-column attention block (CRCAB) is also proposed. CRCAB captures row-to-row and column-to-column dependencies to aggregate global information, enabling stronger representation of the extracted features. Moreover, to reduce the memory overhead of CRCAB, we propose an improved convolutional row-column attention block (ICRCAB), which can adjust the number of vectors in the row-column direction. Finally, to verify the effectiveness of the proposed networks, we conducted experiments on both traditional images and 360° omnidirectional image datasets. The experimental results demonstrated that better performance than for the baseline model was obtained by the network using ECDAB or CRCAB.

Keywords: computer vision; object detection; 360° omnidirectional images; row-column attention mechanism

check for updates

Citation: Zhang, X.; Yang, D.; Song, T.; Ye, Y.; Zhou, J.; Song, Y. Classification and Object Detection of 360° Omnidirectional Images Based on Continuity-Distortion Processing and Attention Mechanism. *Appl. Sci.* **2022**, *12*, 12398. https://doi.org/10.3390/app122312398

Academic Editor: Hui Yuan

Received: 16 November 2022
Accepted: 1 December 2022
Published: 4 December 2022

1. Introduction

The use of 360° omnidirectional images has become more and more popular because they can capture more scene information than traditional images and can meet the increasing demand for broad vision both in industry and daily life. Briefly, 360° omnidirectional images have been widely used in various fields, such as automatic driving [1], 3D scene reconstruction [2], virtual reality [3], and virtual navigation [4]. The most commonly used method to represent 360° omnidirectional images is equirectangular projection (ERP), but the projected images can lose continuity at the left and right edges and produce severe distortion. As shown in Figure 1, the left and right edges of the image are discontinuous and the upper and lower parts are severely distorted.

Furthermore, convolutional neural networks (CNN) have been widely used in various fields of computer vision and have achieved excellent performance. For instance, Pradhan et al. [5] proposed a new method to diagnose cancer using CNN. Mishra et al. [6] used CNN to intelligently identify weed density in soybean crop fields. Mittal et al. [7]

estimated traffic density based on CNN to facilitate traffic management. LeCun et al. [8] applied the LeNet-5 to the task of recognizing handwritten fonts and achieved remarkable results. Krizhevsky et al. [9] applied the AlexNet to achieve strong performance on the ImageNet dataset. With in-depth study of convolutional neural networks, more and more advanced network models have been proposed, including NiN [10], VGG [11], GoogleNet [12], ResNet [13], and other excellent network models [14–18]. At present, most follow-up research being conducted uses these advanced networks as the backbone network to extract better features.

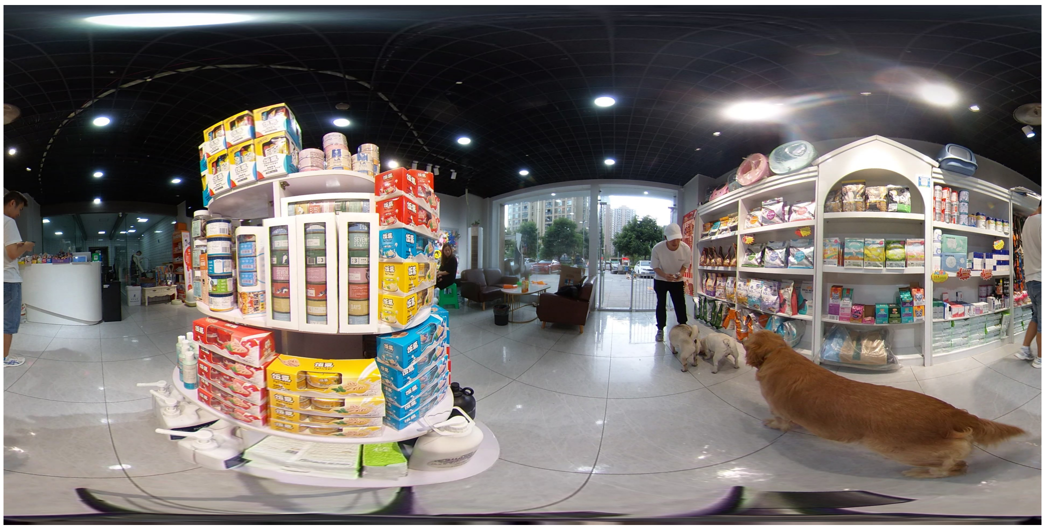
Figure 1. The image captured by the 360° fisheye camera is transformed by equirectangular projection.

Currently, there are many convolutional neural networks obtained using traditional images after training. Since these models do not deal with distortion and discontinuity, the performance obtained by these networks when processing 360° omnidirectional images is not satisfactory. For 360° omnidirectional images, some existing models [19–21] cannot be combined with the pre-trained weights of advanced models since these models are built based on new convolution, which causes difficulty in training of the model. Moreover, many methods [22–27] use reprojection approaches when dealing with discontinuity and distortion, which result in huge computational overhead that can affect the training and inference speed of the model.

To solve these problems, new methods are proposed here. In general, the contributions of the paper can be summarized as follows:

- A novel edge continuity distortion-aware block (ECDAB) and a convolutional row-column attention block (CRCAB) are proposed. ECDAB extracts continuous features through recombination features and group convolution is performed through segmentation features using different convolution kernels between different blocks to alleviate distortion. CRCAB enables interaction of spatial information in a unique way to enhance the receptive field and capacity for feature expression.
- To reduce the memory overhead of CRCAB for very high and wide images or features, we propose an improved convolutional row-column attention block (ICRCAB).
- Based on experimental investigation of classification and object detection using 360° omnidirectional images, it is demonstrated that better performance than for the baseline model is obtained by the network using ECDAB or CRCAB. Moreover, it is shown that CRCAB also performs well when using traditional images.

The rest of the paper is organized as follows: Section 2 details related studies that consider 360° omnidirectional images. Section 3 describes the details of the proposed ECDAB, CRCAB and ICRCAB. Section 4 describes the classification and object detection experiments undertaken. Finally, Section 5 concludes the paper and discusses potential future work.

2. Related Studies

Many state-of-the-art models have been proposed for the use of 360° omnidirectional images in classification [28], object detection [29], and depth estimation tasks [30]. Yang et al. [31] transformed EPR images into spherical images which were projected onto four sub-planes by perspective projection. Each plane image was then fed into YOLOV2 [32] for subsequent processing. Finally, the information learned by each network was aggregated. This multi-projection processing method reduced the influence of distortion, but resulted in greater computational overhead. Monroy et al. [33] and Ruder et al. [34] applied another method that could mitigate the effects of distortion, involving the projection of a 360° omnidirectional image onto the six sub-faces of a cube by perspective projection. Although the distortion was further reduced, the images were not continuous at the boundaries of each cube face and required subsequent processing.

Coors et al. [20] proposed SphereNet, which resampled ERP images (360° omnidirectional images after ERP-processing) to address distortion and left-right edge discontinuity. The method used converted the ERP image into a spherical image. The spherical image was projected onto the tangent plane via a gnomonic projection [35]. Then the convolution kernel sampling position was determined on the tangent plane. Finally, the location of the convolutional kernel sampling in the ERP image was obtained by back projection. Su et al. [36] pointed out that SphereNet introduced an interpolation assumption that could affect the performance of the network as it deepened.

Su et al. [37] suggested that the distortion of ERP images was mainly related to the latitude of 360° omnidirectional images. These authors then trained a convolutional kernel for each row of images. This method involved two networks: the first network enabled the 360° omnidirectional image to be first projected onto multiple tangent planes. Then the trained source network Faster-RCNN [38] was used to extract features. The second network was used to directly extract features from ERP images. Su et al. used features extracted from each layer in the two networks to use the mean square error for back propagation to adjust the weights of the convolution kernels; the learned network SPHCONV [37] was able to handle distortion. Because the convolution kernel of each row in SPHCONV is not shared and needs to be learned, it generates a huge computational overhead. To solve this problem, Su et al. proposed a new network KTN [36], where several rows share a convolution kernel. Whether using SPHCONV or KTN, the source network needed to be used for training before it could be transferred to the target tasks, which is time-consuming.

Lee et al. [21] proposed a representation method based on an icosahedron in which a 360° omnidirectional image was projected onto the icosahedron. As the subdivision of the icosahedron became finer, the effect of distortion was smaller and smaller. However, the computational overhead increased as the degree of subdivision became finer and finer because the convolution operation was not suitable for images represented by an icosahedron. Lee et al. proposed a new method that involved new convolution and pooling operations. Although this projection method reduced the effect of distortion, it entailed huge computational costs.

Li et al. [24] performed depth estimation on 360° omnidirectional images by projecting the 360° omnidirectional images to multiple planes through gnomonic projection, using Transformer [39] to perform efficient feature extraction and, finally, completing the depth estimation task by fusing the output using OmniFusion [24]. Semantic segmentation has became an important component of perception in autonomous driving and 360° omnidirectional images can provide information on a broader field of view. However, Yang et al. [40] found that applying existing semantic segmentation models to 360° omnidirectional images resulted in a sharp degradation in performance. To address this problem, Yang et al. proposed an efficient concurrent attention network that captured long-range dependencies to obtain information about the global context and, in this way, the network achieved better performance. The distribution of 360° omnidirectional images is non-uniform and is distorted differently at different latitudes. The super resolution method for traditional images is not applicable to 360° omnidirectional images, so Deng et al. [41] proposed a new method to apply different upscaling factors to the pixels at different latitudes. In addition, they proposed an automatic selection strategy to automatically select the optimal upscaling factors.

Table 1 summarizes research related to 360° omnidirectional images. At present, many methods use an advanced model infrastructure as the backbone network to extract features. To achieve improved performance and make training easier, many methods involve loading the pre-training weights of the backbone network. However, a number of studies relating to 360° omnidirectional images have proposed new convolution methods. The methods proposed cannot be combined with the pre-training weights of trained backbone networks. It is necessary to retrain the models on a public dataset to obtain the pre-training weights of the network and the multi-projection and resampling methods entailed can generate huge computational costs.

Table 1. A brief summary of studies undertaken in recent years for 360° omnidirectional images and their approach to dealing with distortion and discontinuity.

Works	Year	Processing Methods
Su et al. [37]	2017	Transformation of the network obtained in the plane into ERP images
Yang et al. [31]	2018	Sub-area perspective projection to obtain multiple sub-planes
Monroy et al. [33]	2018	Using the cube mapping method
Coors et al. [20]	2018	Gnomonic projection to obtain a new convolutional kernel picking position
Su et al. [36]	2019	Learn to convert kernel functions for efficient conversion
Lee et al. [21]	2019	Icosahedral projection, new convolution, and pooling methods
Deng et al. [41]	2021	Various upscaling factors for different latitudes, automatic selection strategy
Yang et al. [40]	2021	Attention network that captures global contextual information
Li et al. [24]	2022	Transformer feature extractor, sub-area gnomonic projection

3. Research Methodology

The red dashed box in Figure 2 represents the process of image-processing by the current convolutional neural network. However, for an input 360° omnidirectional image, the distortion and edge discontinuity will make the features extracted in the blue dashed box deviate, which will eventually affect the output of the network. Figure 2 depicts the overall network model structure used in this paper. To improve the performance of the model, it is necessary to make adjustments to mitigate the effects of distortion and edge discontinuities—so distortion and discontinuity processing and feature enhancement are required in the feature extraction section. In this paper, we propose ECDAB and CRCAB models. ECDAB extracts continuous features through recombination features and group convolution is performed through segmentation features using distinct convolution kernels between different blocks to alleviate distortion. CRCAB achieves the interaction of spatial information in a unique way to enhance the receptive field and capacity for feature expression. We describe our proposed methods in detail below.

3.1. Edge Continuity Distortion-Aware Block (ECDAB)

It is beneficial for classification to extract more complete target information; however, in ERP images, the left and right targets are discontinuous and the standard convolution is the same parameter for global spatial features, which is unfavorable for distorted features. More complete target information can be extracted by feature recombination and segmentation. Group convolution is performed to achieve different convolution kernels between different blocks to alleviate distortion.

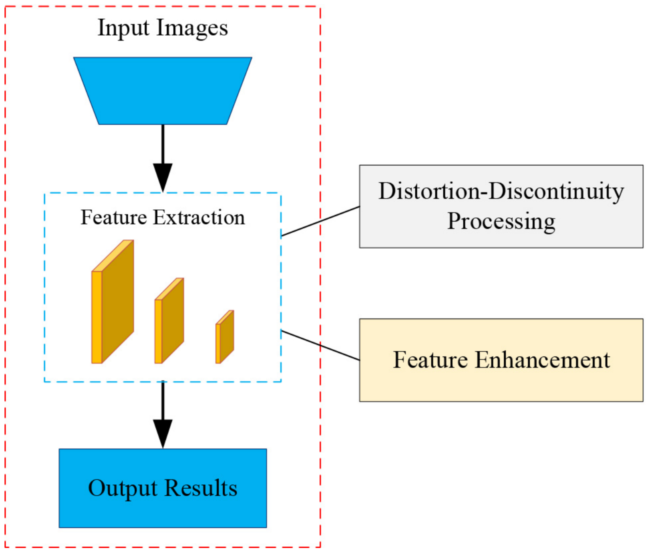

Figure 2. Diagram of the overall network structure giving a brief description of the method in this paper, where distortion and discontinuity processing and feature enhancement are performed in the feature-extraction stage.

In Figure 1, it can be seen that the left and right edges of the image are discontinuous. The ERP image disrupts the continuity of the original 360° omnidirectional images and introduces severe distortion, which will affect the performance of the network. To tackle this situation, we propose ECDAB to process ERP images. Firstly, the continuous edge features of ERP images are extracted through the edge continuity aware block (ECAB). The processed results are then sent to the distortion-aware block (DAB) to alleviate the influence of image distortion. The detailed structure of the ECAB is shown in Figure 3.

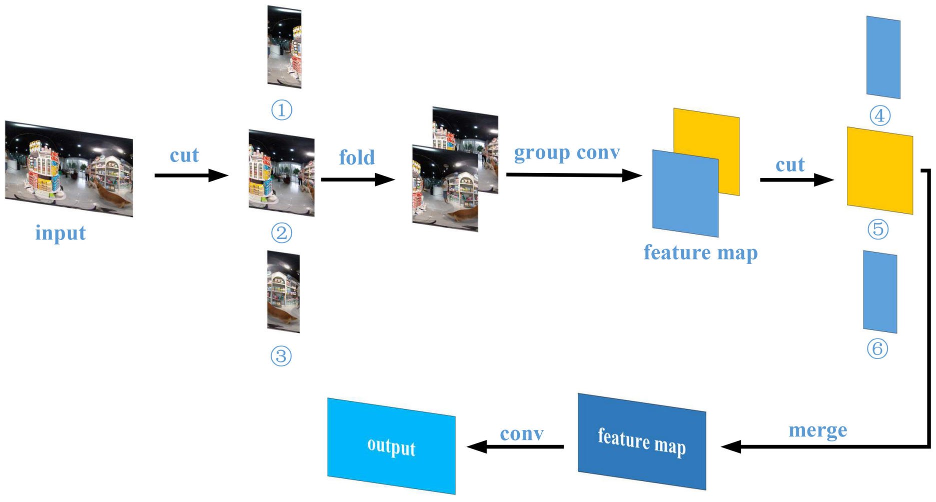

Figure 3. Detailed construction of ECAB.

The image in Figure 1 is taken as an example to extract the continuous features of the left and right edges. Firstly, the image is divided into three pieces ①, ② and ③, then pieces ① and ③ are flipped horizontally and the flipped results are stitched (enabling the left and right edges of the image to be continuous). This process can be understood as a folding operation on a picture, namely, the left and right sides of the image are folded inwards. In ECAB, the shape of the image obtained after stitching ① and ③ needs to be the same as ②, which requires that the width of the input image or feature can be divided by four. Then the results after stitching ① and ③ are stacked with ② in the channel dimension. The next step is to extract features with continuous information using grouping convolution.

The results of grouping convolution are also segmented and stitched, namely, the features obtained by grouping convolution of the original ① and ③ are segmented into two pieces ④ and ⑥. Then they are flipped horizontally—the goal is to ensure that images or features remain in the original order. Then ④, ⑤ and ⑥ are stitched together to form features as the input shape. Finally, a convolution operation is performed to mitigate the negative effects of folding. So the left and right edges of features extracted by ECAB contain continuous information. To further alleviate the influence of distortion, features extracted by EDAB are sent to the distortion-aware block (DAB) for further processing.

Su et al. [37] pointed out that the distortion of ERP images is related to the latitude of the 360° omnidirectional images, while the distortion of the 360° omnidirectional images after ERP is related to the height. So, in the DAB, the ERP image is divided into N blocks along the row direction, and then these blocks are stacked on the channel dimension (the shape of features changes from B C H W to B C×N H//N W). Here, B, C, H, and W represent batch-size, channels, height and width for input images or features, respectively, and the symbol // indicates an integer division symbol. Then the segmented images are grouped for the convolution operation. The purpose of the above operation is to extract features with different convolution kernels for different blocks. Note: the number of blocks needs to be set when implementing a DAB; simply speaking, the segmentation here is equivalent to a reshape operation in the deep learning framework. Finally, the results obtained by grouping convolution are readjusted to the original shape (B C×N H//N W to B C H W) and then a convolution operation is carried out to compensate for the influence of segmentation. DAB's detailed structure is depicted in Figure 4. ECAB and DAB are connected in series to form the edge continuity distortion-aware block (ECDAB).

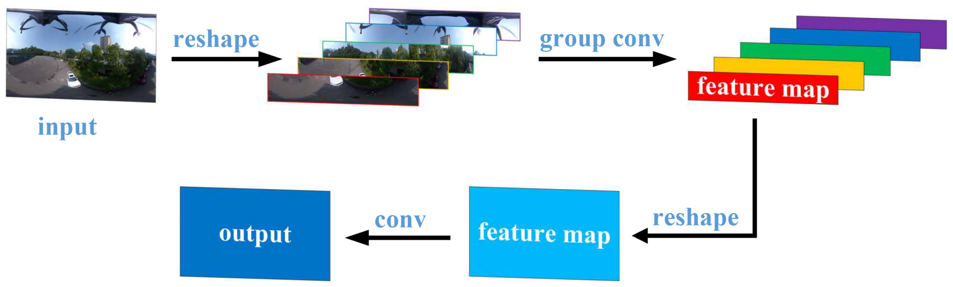

Figure 4. Detailed construction of DAB.

3.2. *Convolutional Row-Column Attention Block (CRCAB)*

Recently, some studies [39,42,43] have shown that the performance of the network can be improved by spatial information interaction, which can capture global information by self-attention, but which causes a huge memory overhead for high resolution images. In this investigation, we implement spatial information interaction in a unique way, improving the memory overhead during information interaction. In the following, we describe our proposed method in detail.

CRCAB uses convolution to obtain row-to-row and column-to-column dependencies and, through this dependence, it aggregates information to build features with stronger representation ability. First, CRCAB takes the maximum and mean values in the row and column directions of the features and adds them to obtain the compression features. The compressed features are then sent to the convolutional operation for learning to obtain weights. To obtain more comprehensive information, a row-column self-attention mechanism similar to [39] and a finer row-column attention mechanism are implemented in CRCAB. Finally, the results obtained by the two row-column attention mechanisms are stacked on the channel dimension and then convolution is implemented to fuse them to build new features. To describe the structure of our CRCAB more clearly, we explain the two operations separately (see Figures 5 and 6). The two parts of the compressed feature are shared during implementation.

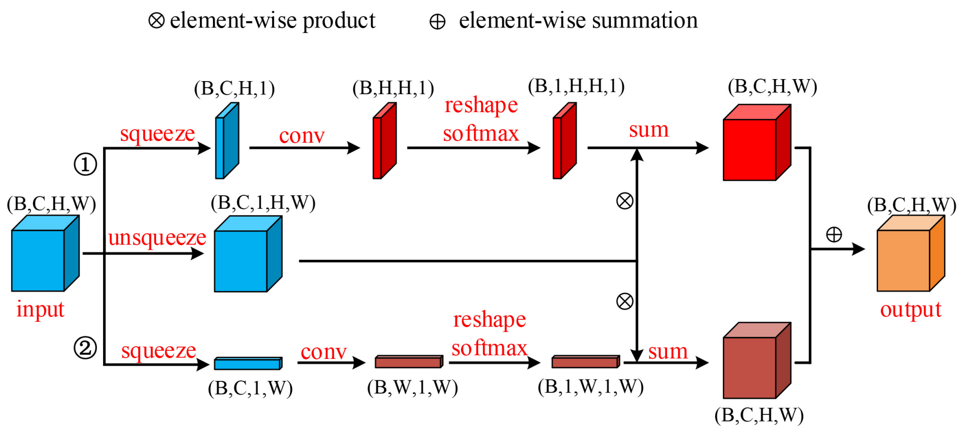

Figure 5. Detailed structure of the row-column self-attention mechanism.

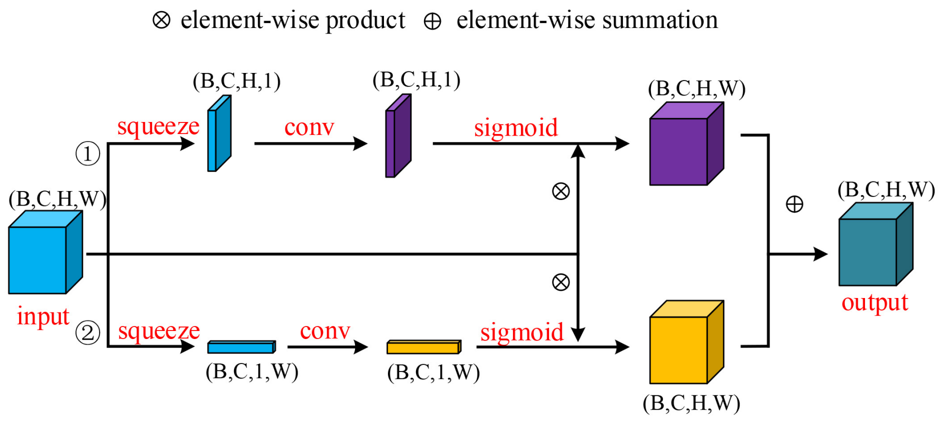

Figure 6. More detailed structure of the row-column attention mechanism.

The detailed structure of the convolutional row-column self-attention mechanism is shown in Figure 5; we implement a self-attention mechanism similar to that in [39] on both row and column directions. The row direction ① is taken as an example and the column direction ② is the same. Firstly, a squeeze operation is performed in the row direction (the maximum and mean values are added) to obtain the compressed features (the shape is B C H W; B C H W here still represents the batch-size, channels, height, and width of features, and the value of W is 1). Next, the convolution operation is performed to obtain weights (the shape is B H H 1). To realize the self-attention mechanism, it is necessary to adjust the learned weight to five dimensions (namely, the shape is B 1 H H 1). Then, a softmax operation is performed in the row direction. Because the weight is five dimensions, it is necessary to expand the input features into five dimensions (the shape is B C 1 H W), namely, the unsqueezed process shown in Figure 5. Then the expanded features are multiplied with the learned weights. Finally, a sum operation is performed in the row direction to reconstruct the features. To implement the self-attention mechanism in the row-column direction to construct features, it can be simply represented by Equations (1) and (2):

$$X_i = \partial_{i1}X_1 + \partial_{i2}X_2 + \ldots + \partial_{iH}X_H \quad (\partial_{i1} + \partial_{i2} + \ldots + \partial_{iH} = 1)\ (i = 1, 2, 3, \ldots, H) \tag{1}$$

$$X_j = \partial_{j1}X_1 + \partial_{j2}X_2 + \ldots + \partial_{jW}X_W \quad (\partial_{j1} + \partial_{j2} + \ldots + \partial_{jW} = 1)\ (j = 1, 2, 3, \ldots, W) \tag{2}$$

In CRCAB, along the row and column directions, the features are divided into H and W feature vectors. These feature vectors are represented in Equations (1) and (2). X_i represents the vector of the i-th row of new features, X_j represents the vector of the j-th column of new features, X_1, X_2, ..., X_H and X_W represent the row and column vectors of the original features, and H, W represent the height and width of the original features.

In Equations (1) and (2), ∂ represents the weights learned after the convolution operation (for an image, if the weights learned for the row direction are adjusted and the final shape is 1 1 H H 1, it can be considered as a H×H matrix. If the weights are obtained for the column learning, the adjusted shape is 1 1 W 1 W—this can also be regarded as a matrix of W×W). When implementing row-column self-attention, the row and column vectors of each channel have the same weights and are shared for each channel. Finally, the features constructed in the row and column directions are added and fused to obtain the output. There are some details that need to be noted here: the features are finally obtained shape-by-sum on the column direction B C W H, so a transposition operation is required, namely, B C W H to B C H W. For the sake of simplicity, the transposition operation is not shown in Figure 5.

Figure 6 implements a finer row-column attention mechanism. The row direction ① is taken as an example, and the column direction ② is the same. Firstly, the convolution operation is performed on the compressed features and then a sigmoid operation is performed to obtain the weights. The obtained weights are not shared for each channel and each row corresponds to a weight. The obtained weights are then multiplied by the input features to build new features. Finally, the features obtained in the row and column directions are added to fuse. We use CRCAB to stack the output features obtained in Figures 5 and 6 in the channel dimension and, finally, perform convolution fusion to obtain the final output. The multiplication operation in Figures 5 and 6 benefits from the broadcasting mechanism in the deep learning framework. The weights and features are copied to achieve the same shape as we indicated when constructing a mechanism similar to self-attention where the row and column vectors on each channel share the same weights.

3.3. Improved Convolutional Row-Column Attention Block (ICRCAB)

Our proposed CRCAB suffers inherently from a huge memory overhead in constructing a row-column self-attention mechanism for very wide and high features. As was mentioned in the previous subsection, during the construction of self-attention, the vectors of the rows and columns of features and weights need to be copied, which requires huge memory overhead for very high and wide features. To alleviate this problem, we propose the ICRCAB; the row direction is taken as an example.

As shown in Figure 7, in CRCAB, we treat each row as a vector, but now we treat each block as a vector containing H // N rows. In this way, we can control N to reduce the memory overhead and, to realize the self-attention mechanism, the vectors of the rows in the original CRCAB need to be copied H times; however, in the ICRCAB, they only need to be copied N times, so the memory consumption can be adjusted by N. The details in ICRCAB are different in certain respects compared to CRCAB. Here, we briefly describe certain details of ICRCAB. For the input features, we adjust the shape B C H W to B C N H//N W; then the squeeze operation is performed. In ICRCAB, we use the global average and maximum pooling to obtain the compressed features (the shape is B C N 1 1 and then we adjust the shape of features to B C N 1); the shape of weights obtained is B N N 1 after the convolutional operation. The obtained weights and input features need to be adjusted to six dimensions; the subsequent steps are the same as for the original CRCAB. The ICRCAB construction process for each vector of the row and column is shown in Equations (3) and (4). *N1* represents the number of row vectors in the row direction, *N2* represents the number of column vectors in the column direction; *N1* and *N2* are required to be specified in the build.

$$X_i = \partial_{i1}X_1 + \partial_{i2}X_2 + \ldots + \partial_{iN1}X_{N1} \quad (\partial_{i1} + \partial_{i2} + \ldots + \partial_{iN1} = 1)\ (i = 1, 2, 3, \ldots, N1) \tag{3}$$

$$X_j = \partial_{j1}X_1 + \partial_{j2}X_2 + \ldots + \partial_{jN2}X_{N2} \quad (\partial_{j1} + \partial_{j2} + \ldots + \partial_{jN2} = 1)\ (j = 1, 2, 3, \ldots, N2) \tag{4}$$

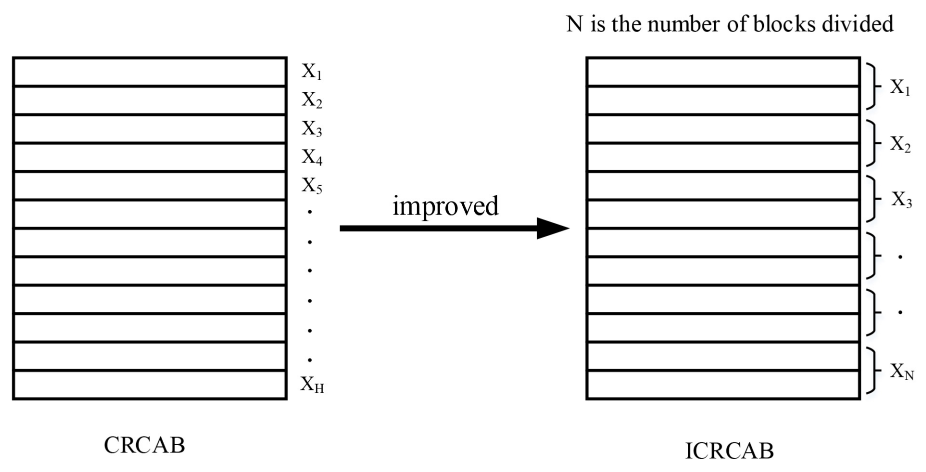

Figure 7. The feature row vectors in CRCAB (**left**) and ICRCAB (**right**).

4. Experiments

4.1. Classification

For the classification task, the Omni-MNIST [20] dataset is used in this paper. This dataset is an equirectangular projection generated by MNIST through back projection with a resolution of 60 × 60. As for the MNIST dataset, the Omni-MNIST data includes 100,000 training and 60,000 test samples with 10 classes. We use the convolution neural network to directly train the equirectangular images of the Omni-MNIST dataset to obtain EquirectCNN, which has four convolutional layers and one fully connected layer. The acquired features after the first layer of convolution are not processed by the max pooling and activation function and do not change the shape of the input, while the other convolution layers are processed using the max pooling and activation function. SphereNet [20] can also solve the problem of edge discontinuity and distortion by resampling. For comparison with the proposed ECDAB, the first convolutional layer of EquirectCNN is replaced by SphereNet to build SphereNetCNN. The question arises of why all the convolutional layers of the network are not constructed using the SphereNet. The reason is that Su et al. [36] also noted that the interpolation assumption of ShpereNet affects the performance of the network as the depth of the network deepens. Similarly, we trained ECDANet, which replaces the first convolutional layer of EquirectCNN with our proposed ECDAB. We trained the CRCANet based on the CRCAB, which removed the first layer of EquirectCNN and added a CRCAB in front of the last fully connected layer. Similarly, we replaced CRCAB with ICRCAB and constructed ICRCANet. In addition, we also trained ECDA-CRCANet and ECDA-ICRCANet, which combine ECDAB, CRCAB, and ICRCAB. ECDA-CRCANet adds a CRCAB in front of the last fully connected layer of ECDANet and the DAB also divides the image into two blocks. ECDA-ICRCANet replaces the CRCAB in ECDA-CRCANet with the ICRCAB. To achieve better performance from the network, we add the output features of EDCAB, CRCAB and ICRCAB to the input features to construct the residual structure.

During training the optimizer is Adam [44], with a base learning rate of 0.001 and batches of size 128 for 100 epochs. In the experiment, we found that increasing the number of segmentation blocks in DAB had a negative impact on the final classification accuracy. In general, we suggest that zero padding is performed when group convolution occurs after segmentation and that the higher the number of segmentation blocks, the more zero padding is performed. For the classification experiments, the shallow depth of the network model did not accommodate zero padding well, so the gain in distortion mitigation was not high. Finally, the number of segmentation blocks was set to two. For ICRCAB, we added it to the last layer of the network, which had an output resolution of 6 × 6. After considering the final accuracy, the input features were divided into three vectors of rows and columns.

To achieve greater rigor in the experiment, each network was trained five times, with 100 epochs each time and, finally, the average of the precision was taken. The experimental results are shown in Table 2. The change in test accuracy in the training process is shown in Figure 8; the test accuracy was obtained by taking the average of five experimental results. The experimental results showed that most of our proposed models achieved lower classification error rates than the baseline model, especially for the error rates obtained by ICRCANet and ECDA-ICRCANet, which were 2.8% and 4.01%, respectively, both lower than EquirectCNN.

Table 2. Classification error rates of the different models on Omni-MNIST.

Methods	Test Error Rate (%)
EquirectCNN	9.79
SphereNetCNN	9.46
ECDANet	9.68
CRCANet	**8.02**
ICRCANet	**6.54**
ECDA-CRCANet	**6.99**
ECDA-ICRCANet	**5.78**

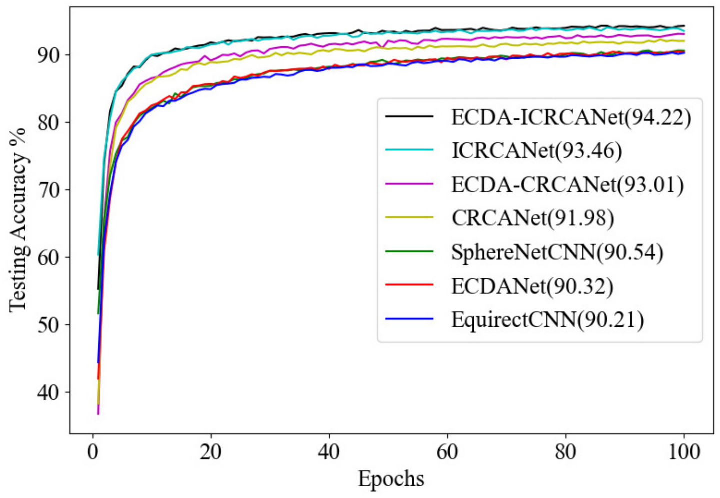

Figure 8. Changes in test accuracy during training.

To verify the effectiveness of our proposed CRCAB, we only performed rigorous experiments on CIFAR-10. Although the focus of this paper is on 360° omnidirectional images, CRCAB is equally valid for conventional images. The experiment undertaken was similar to that for the Omni-MNIST dataset, with the same network structure. It was found that our proposed CRCANet was 1.49% more accurate than the baseline model trained with traditional images.

4.2. Object Detection

Currently, there are few public datasets of 360° omnidirectional images, so we built the ERP-Detect dataset, with a total of 2305 images. The dataset is captured using a fisheye camera combined with a drone and tripod, sampled in different environments, such as roads, basketball courts, parking lots, etc., and then the images are rendered as ERP images. The closer the object is to the camera, the stronger the distortion it suffers. The field of view is further expanded by drone vision, which enables images to be rendered in different locations. In addition, there are many duplicate scenes in the images, so, for the experiment, we removed the duplicate images of the same scenes and then selected 669 images containing people (1086 in total), cars (1094 in total), buildings (625 in total), and motors (269 in total) to be annotated. Here, "motors" refers to electric motorcycles.

Finally, these images were divided into 541 images for training, 61 images for validation, and 67 images for testing.

We chose YOLOV3 [45] to validate our methods. We trained seven networks based on YOLOV3; the ERP images were sent directly into YOLOV3 for training to get EquirectYOLO. In addition, two layers of SphereNet were added in series in front of Darknet53 in YOLOV3 to build SphereNetYOLO. The two SphereNet layers in SphereNetYOLO were replaced with ECDAB. Note: Our two ECDABs were concatenated. In Darknet53, we added two CRCABs to obtain the improved Darknet53, as shown in the left part of Figure 9. Based on the improved Darknet53, we built CRCAYOLO and added the same two ECDABs in CRCAYOLO to construct ECDA-CRCAYOLO. CRCAB cannot be placed anywhere in the original Darknet53, which is limited by its own shortcomings, but our proposed ICRCAB is unaffected by this and can control memory consumption by adjusting N. Therefore, ICRCAB can be placed anywhere in Darknet53 just by adjusting N, as shown in the right part of Figure 9. Because the memory overhead of CRCAB is very large for very high and wide inputs, our hardware environment does not support its appearance in the same location as ICRCAB. At the same time, for very high and wide inputs, our improved ICRCAB not only has less memory overhead, but also obtains better test results than CRCAB. Similar to CRCAYOLO and ECDA-CRCAYOLO, we built ICRCAYOLO and ECDA-ICRCAYOLO using improved Darknet53 based on ICRCAB.

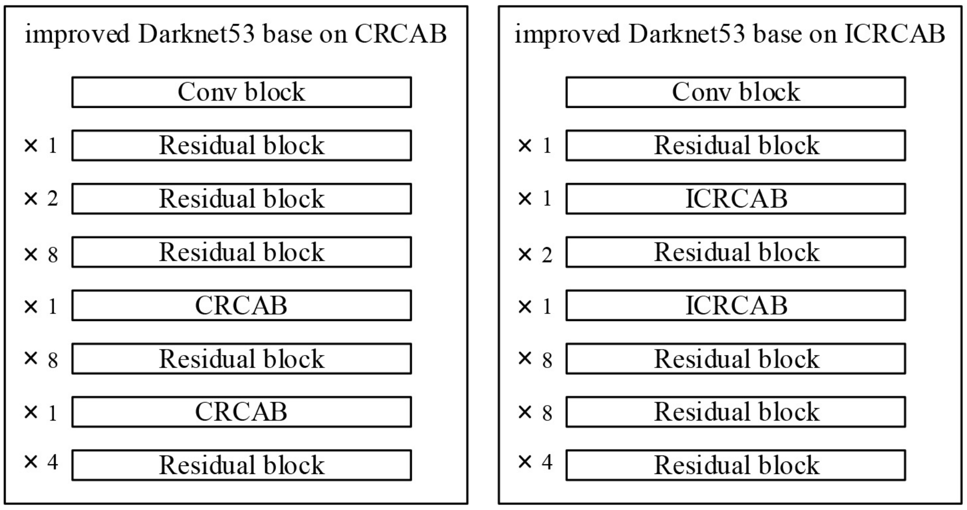

Figure 9. Improved Darknet53 based on CRCAB and ICRCAB.

During the experiment, we loaded the pre-training weights of Darknet53. The pre-training weights were obtained by training on the COCO [46] dataset. If the pre-training weights are not loaded, the model will become difficult to train. This is also the reason why many models described in related studies are not suitable for our research. In addition, to obtain improved robustness of the model, we used random data augmentation. Random data augmentation does not increase the number of training samples, but randomly enhances the input data, namely, color gamut distortion, flipping, random scaling height and the width of images. Finally, the resolution of the image, which is 416 $\times$ 416 is sent to the model. Due to the broadcasting mechanism in the deep learning framework, the implementation of CRCAB we proposed is simple.

During the training process, the optimizer still uses Adam, a base batch size of 10 for 100 epochs, and a learning rate of 0.0005, which is reduced by a multiple of 0.98 each iteration. In the target detection experiments, the number of segmentation blocks for ECDAB was inspired by SPHCONV [37], so the number of two segmentation blocks was set to 416 and 8, respectively. For ICRCAB, on the one hand, if the number of row vectors and column vectors is too large, there will be a huge memory overhead when establishing self-attention and it will gradually become CRCAB. On the other hand, if the number of row vectors and column vectors is too small, more information will be lost after compressing the features in the pooling layer. Finally, eight was chosen as the number of row and column vectors in ICRCAB.

To increase the rigor of the experiment, each model was trained five times. Similar to classification, the evaluation measure was finally obtained by averaging; the IOU was 0.5 during the test. The experimental results are shown in Table 3.

Table 3. Test results for the different models on ERP-Detect.

Methods	Building	Car	Motor	Person	mAP
EquirectYOLO	83.66	90.75	43.25	79.61	74.32
SphereNetYOLO	83.88	91.61	43.86	78.77	74.53
ECDAYOLO	**84.85**	90.61	**46.37**	78.56	**75.09**
CRCAYOLO	**86.87**	91.43	**48.46**	79.02	**76.45**
ICRCAYOLO	**85.75**	**91.67**	**53.31**	78.78	**77.38**
ECDA-CRCAYOLO	82.92	**92.03**	**52.48**	**80.23**	**76.92**
ECDA-ICRCAYOLO	**84.60**	90.85	**54.49**	**80.21**	**77.54**

There was a slight decrease in detection accuracy for cars and people in some of our proposed network models, but our models showed enhanced adaptability for buildings and motorbikes. According to mAP evaluation, all our proposed network models performed better than EquirectYOLO and ShpereNetYOLO, so the experimental results of object detection demonstrated that our proposed model was effective. We selected EquirectYOLO, ShpereNetYOLO, ECDAYOLO, and CEDA-ICRCAYOLO as object detection examples. As shown in Figure 10, it can be seen that our models produced better detection results than EquirectYOLO and ShpereNetYOLO.

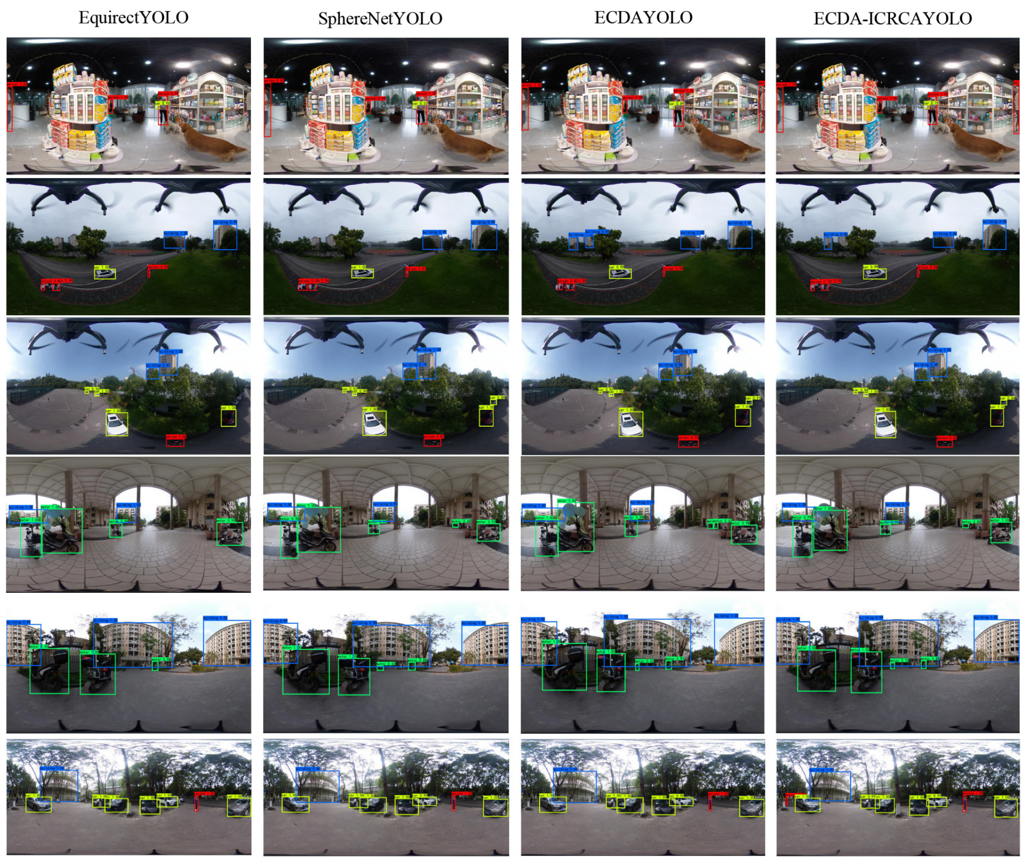

Figure 10. Object detection instance.

It is of note that, although the number of images in ERP-Detect after removing repeated scenes was only 669, this did not affect the effectiveness of our model. In addition, compared with the baseline model, our proposed model had a small increase in parameters. The baseline model (EquirectYOLO) had 235 MB of parameters, while our largest model, ECDA-CRCAYOLO, had 266 MB of parameters, which was an increase of 31 MB of parameters; however, this is acceptable for deep learning. CRCAB itself had very few parameters because, in addition to the last convolutional layer that fuses two row-column attention mechanisms, we used a convolution kernel of shape 3×1 and 1×3 for row and column direction when building attention. The reason for the increase in the number of parameters by 31 MB was the number of channels in the input features, and that the training speed of our proposed model was not much different from the baseline model EquirectYOLO.

5. Conclusions

In this paper, we proposed ECDAB to deal with distortion and edge discontinuity caused by 360° omnidirectional images through equirectangular projection and also proposed CRCAB, which can capture global contextual information. To address the shortcomings of CRCAB, ICRCAB was proposed, and then networks were built based on these modules. The performance of these networks was demonstrated to outperform the baseline model through classification and object detection experiments based on the Omni-MNIST and ERP-Detect datasets. Our proposed method can be combined with existing excellent pre-training models without additional overhead time to train pre-training weights and is computationally much less expensive than the proposed network for 360° omnidirectional images, since our proposed method does not require multi-projection or resampling.

In the future, for 360° omnidirectional images, ECDAB and ICRCAB can be migrated to the scene segmentation task to improve the performance of the network. Moreover, although we only performed classification experiments on CIFAR-10, CRCAB and ICRCAB can be applied to more complex models for other tasks relating to traditional images.

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

Funding: This research was funded by the Natural Science Foundation of Chongqing (CSTB2022NSCQ-MSX1200), the Science and Technology Research Program of Chongqing Municipal Education Commission (KJQN202200537), Chongqing Normal University Ph.D. Start-up Fund (21XLB035), and the Special Funds for Postdoctoral Research of Chongqing (2010010004713415).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets used and analyzed during the current study are available from the corresponding author on reasonable request.

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

References

1. Arena, F.; Ticali, D. The development of autonomous driving vehicles in tomorrow's smart cities mobility. *AIP Conf. Proc.* **2018**, *2040*, 140007.
2. Guo, H.; Peng, S.; Lin, H.; Wang, Q.; Zhang, G.; Bao, H.; Zhou, X. Neural 3D scene reconstruction with the manhattan-world assumption. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 5511–5520.
3. Schuemie, M.J.; Van Der Straaten, P.; Krijn, M.; Van Der Mast, C.A. Research on presence in virtual reality: A survey. *CyberPsychol. Behav.* **2001**, *4*, 183–201. [CrossRef] [PubMed]
4. Harvey, C.D.; Collman, F.; Dombeck, D.A.; Tank, D.W. Intracellular dynamics of hippocampal place cells during virtual navigation. *Nature* **2009**, *461*, 941–946. [CrossRef] [PubMed]

5. Pradhan, K.S.; Chawla, P.; Tiwari, R. HRDEL: High ranking deep ensemble learning-based lung cancer diagnosis model. *Expert Syst. Appl.* **2023**, *213*, 118956. [CrossRef]
6. Mishra, A.M.; Harnal, S.; Gautam, V.; Tiwari, R.; Upadhyay, S. Weed density estimation in soya bean crop using deep convolutional neural networks in smart agriculture. *J. Plant Dis. Prot.* **2022**, *129*, 593–604. [CrossRef]
7. Mittal, U.; Chawla, P.; Tiwari, R. EnsembleNet: A hybrid approach for vehicle detection and estimation of traffic density based on faster R-CNN and YOLO models. *Neural Comput. Appl.* **2022**, 1–20. [CrossRef]
8. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
9. Krizhevsky, A.; Sutskever, 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, 3–8 December 2012; Volume 25.
10. Lin, M.; Chen, Q.; Yan, S. Network in network. *arXiv* **2013**, arXiv:1312.4400.
11. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
12. 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, 7–12 June 2015; pp. 1–9.
13. 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.
14. Piciarelli, C. MS-Faster R-CNN: Multi-stream backbone for improved faster R-CNN object detection and aerial tracking from UAV images. *Remote Sens.* **2021**, *13*, 1670.
15. Gu, P. A multi-source data fusion decision-making method for disease and pest detection of grape foliage based on ShuffleNet V2. *Remote Sens.* **2021**, *13*, 5102.
16. Liu, Z.; Mao, H.; Wu, C.Y.; Feichtenhofer, C.; Darrell, T.; Xie, S. A convnet for the 2020s. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 11976–11986.
17. Li, D.; Hu, J.; Wang, C.; Li, X.; She, Q.; Zhu, L.; Zhang, T.; Chen, Q. Involution: Inverting the inherence of convolution for visual recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 12321–12330.
18. Yin,L.; Hong, P.; Zheng, G.; Chen, H.; Deng, W. A novel image recognition method based on densenet and dprn. *Appl. Sci.* **2022**, 12, 4232. [CrossRef]
19. Esteves, C.; Allen-Blanchette, C.; Makadia, A.; Daniilidis, K. Learning so(3) equivariant representations with spherical cnns. *Int. J. Comput. Vis.* **2020**, *128*, 588–600. [CrossRef]
20. Coors, B.; Condurache, A.P.; Geiger, A. Spherenet: Learning spherical representations for detection and classification in omnidirectional images. In Proceedings of the European conference on computer vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 518–533.
21. Lee, Y.; Jeong, J.; Yun, J.; Cho, W.; Yoon, K.J. Spherephd: Applying cnns on a spherical polyhedron representation of 360deg images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 9181–9189.
22. Orhan, S.; Bastanlar, Y. Semantic segmentation of outdoor panoramic images. *Signal Image Video Process.* **2022**, *16*, 643–650. [CrossRef]
23. Khasanova, R.; Frossard, P. Graph-based classification of omnidirectional images. In Proceedings of the IEEE International Conference on Computer Vision Workshops, Venice, Italy, 22–29 October 2017; pp. 869–878.
24. Li, Y.; Guo, Y.; Yan, Z.; Huang, X.; Duan, Y.; Ren, L. Omnifusion: 360 monocular depth estimation via geometry-aware fusion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 2801–2810.
25. Eder, M.; Shvets, M.; Lim, J.; Frahm, J.M. Tangent images for mitigating spherical distortion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 12426–12434.
26. Zhang, C.; Liwicki, S.; Smith, W.; Cipolla, R. Orientation-aware semantic segmentation on icosahedron spheres. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 3533–3541.
27. Zhao, P.; You, A.; Zhang, Y.; Liu, J.; Bian, K.; Tong, Y. Spherical criteria for fast and accurate 360 object detection. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 12959–12966.
28. Zhao, Q.; Zhu, C.; Dai, F.; Ma, Y.; Jin, G.; Zhang, Y. Distortion-aware CNNs for Spherical Images. In Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 13–19 July 2018; pp. 1198–1204.
29. Goodarzi, P.; Stellmacher, M.; Paetzold, M.; Hussein, A.; Matthes, E. Optimization of a cnn-based object detector for fisheye cameras. In Proceedings of the 2019 IEEE International Conference on Vehicular Electronics and Safety, Cairo, Egypt, 4–6 September 2019; pp. 1–7.
30. Zhao, C.; Sun, Q.; Zhang, C.; Tang, Y.; Qian, F. Monocular depth estimation based on deep learning: An overview. *Sci. China Technol. Sci.* **2020**, *63*, 1612–1627. [CrossRef]
31. Yang, W.; Qian, Y.; Kämäräinen, J.K.; Cricri, F.; Fan, L. Object detection in equirectangular panorama. In Proceedings of the 2018 24th International Conference on Pattern Recognition, Beijing, China, 20–24 August 2018; pp. 2190–2195.

32. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271.
33. Monroy, R.; Lutz, S.; Chalasani, T.; Smolic, A. Salnet360: Saliency maps for omni-directional images with cnn. *Signal Process. Image Commun.* **2018**, *69*, 26–34. [CrossRef]
34. Ruder, M.; Dosovitskiy, A.; Brox, T. Artistic style transfer for videos and spherical images. *Int. J. Comput. Vis.* **2018**, *126*, 1199–1219. [CrossRef]
35. Coxeter, H.S.M. *Introduction to Geometry*; John Wiley & Sons: New York, NY, USA; London, UK, 1961.
36. Su, Y.; Grauman, K. Kernel transformer networks for compact spherical convolution. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 9442–9451.
37. Su, Y.; Grauman, K. Learning spherical convolution for fast features from 360 imagery. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Volume 30.
38. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; Volume 28.
39. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; 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; Volume 30.
40. Yang, K.; Zhang, J.; Reiß, S.; Hu, X.; Stiefelhagen, R. Capturing omni-range context for omnidirectional segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 1376–1386.
41. Deng, X.; Wang, H.; Xu, M.; Guo, Y.; Song, Y.; Yang, L. Lau-net: Latitude adaptive upscaling network for omnidirectional image super-resolution. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 9189–9198.
42. Vaswani, A.; Ramachandran, P.; Srinivas, A.; Parmar, N.; Hechtman, B.; Shlens, J. Scaling local self-attention for parameter efficient visual backbones. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 12894–12904.
43. 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.
44. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
45. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
46. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 8–11 September 2014; pp. 740–755.