A semantic segmentation network based on multi-branch structures and multi-scale modules

Jiangxin Hui (1815200841@qq.com)
Xi'an University of Posts and Telecommunications
Hong Zhang
Xi'an University of Posts and Telecommunications

Article

Keywords:

Posted Date: May 20th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1626129/v1

A semantic segmentation network based on multi-branch structures and multi-scale modules

Jiangxin Hui[1], Hong Zhang[2]
[1,2] School of Automation, Xi'an University of Posts and Telecommunications, Xi'an, 710121, China

* Corresponding authors: Jiangxin Hui (e-mail: 1815200841@qq.com)

Abstract

A novel semantic segmentation model is proposed to improve segmentation accuracy for small and obstructed image targets. This multi-branch structure increased gradient flow paths and prevented vanishing gradients. The architecture was based on a DeepLabv3+ framework and utilized grouped convolutions in each bottleneck of the backbone network (ResNet50), to reduce both the number of model parameters and the model size. The network was also re-parameterized to increase speed during the inference period, which had little effect on the segmentation results. A multi-scale hierarchical attention module (MHAM) was applied in parallel with atrous spatial pyramid pooling (ASPP) in the encoder, to fuse feature information output from the two modules and achieve adaptive segmentation of multi-scale targets. Transfer learning and data augmentation were also used to accelerate convergence and further improve model robustness. The proposed network was evaluated using an aerial semantic segmentation benchmark (AeroScapes), to assess segmentation performance for objects at different scales. The mean intersection over union (mIoU), calculated for the validation set, improved by 43.12% and 47.51% compared with the DeepLabv3+ (Xception65) network and the DeepLabv3+ (ResNet50), respectively. In addition, the proposed network achieved a higher mIoU (84.98%) and a higher mean pixel accuracy (mPA, 97.57%) than six other advanced semantic segmentation networks (U-Net, RefineNet, PSPNet, DADA, DSRL, and HANet), as well as higher mIoUs than comparable algorithms for the CityScapes (84.96%) and ADE20K (52.72%) datasets. It was also found that doubling the number of channels after the grouping convolutions did not significantly change the number of model parameters or the model size. However, the acquired features were more detailed and the images were more complete, indicating the proposed model achieved better segmentation accuracy for small and occluded targets.

Introduction

The continued growth of image processing and computer vision has led to an increased interest in semantic segmentation [1] in recent years, particularly for targets that are difficult to identify. Images and videos captured by drones offer the specific advantages of flexible viewpoints, continuous fields of view, and wide surveillance ranges. However, the images collected by drones often include small or obstructed objects, for which detection segmentation is both difficult and critical. For example, the unmanned aerial vehicle (UAV) AeroScapes dataset includes multi-scale targets and a large number of small or occluded targets that pose a challenge for segmentation. If these regions of interest cannot be accurately segmented [2], the final processed result may be seriously affected, despite successes in other parts of the image.

Conventional image segmentation methods divide an entire image into several non-overlapping regions based on grayscale, color, texture, and shape. Unsupervised learning is an effective segmentation technique, in which marked semantic labels for segmented results are unnecessary. In recent years, artificial intelligence (AI) has improved the development of computer vision technology in multiple fields. AI has also been applied to semantic image segmentation for pedestrian detection [3], autonomous driving [4], facial recognition [5], biomedicine [6], and other areas requiring a segmentation map with object category labels.

Semantic segmentation is the process of segmenting each pixel in an image exhibiting a semantic value with a particular label, allowing for simple analysis by assigning semantic definitions to specific regions of the image. Early semantic segmentation models primarily relied on convolutional neural networks (CNNs) with VGG-16 serving as the backbone. While broadly successful, small-scale object features were often lost with this approach. The development of a fully convolutional network (FCN) [7], proposed by Long et al., led to the use of deep-learning models for semantic segmentation. FCNs involve end-to-end and pixel-to-pixel training, which can be used to transfer input images of any size to same-sized output images using reasoning and learning. Ronneberger et al. [8] presented a symmetric encoder-decoder architecture (U-Net), which made full use of output characteristics in each layer by using dense jump connections. In this process, the encoder performs deep-level feature extraction of the image to generate high-level semantic information and the decoder splices and fuses feature maps at different resolutions (through jump connections) to produce segmentation results. Unlike a U-Net, the SegNet proposed by Badrinarayanan et al. [9] includes a decoder that utilizes the pooling index of corresponding encoders to perform upsampling, which significantly reduces the number of parameters. Stochastic gradient descent is then applied for end-to-end training. As a result, storage and calculation times during inference are excellent. The PSPNet proposed by Zhao et al. [10] introduced a pyramid pooling module (PPM) to aggregate multi-scale contextual information, which improved the network's ability to capture global details. Lin et al. [11] proposed a general multi-path optimization network, RefineNet, which takes advantage of down-sampled feature maps. Long-range residual connections are used to form short-range internal connections and establish correspondence with a ResNet. The gradient can then be effectively transmitted across the entire network.

Chen et al. proposed the DeepLab series of models for accurately segmenting small objects. Specifically, DeepLabv1 [12] integrates deep CNNs (DCNNs) and conditional random field DenseCRFs to increase the impact of acquired details and compensate for low image resolution using maximum pooling and down-sampling operations. The invariance of the included spatial transformation also increased the resulting accuracy. DeepLabv2 [13] excluded the down-sampling step from the last few maximum pooling layers of the deep convolutional network and used atrous convolutions to calculate a feature map with a higher sampling density. The atrous spatial pyramid pooling (ASPP) module simultaneously captures richer image details at multiple scales. In DeepLabv3 [14], a cascaded or parallel atrous convolution module was included to identify multi-scale information. The ASPP module was also enhanced while the CRF was omitted. DeepLabv3+ [15] introduced an encoder-decoder module to better integrate multi-scale information and applied a deep separable convolution to the ASPP and decoder modules, to further improve segmentation performance.

The difficulty of segmenting small-scale and occluded objects has been addressed using a variety of networks in recent years. For example, Xia et al. improved the kernel correlation filter algorithm to increase segmentation accuracy for obstructed objects [16]. Chen et al. utilized feature map attention mechanisms to develop a super-resolution reconstruction algorithm [17]. Li et al. proposed a point-wise affinity propagation module based on the FPN framework for small object annotation in aerial images [18]. Chen et al. developed a novel image annotation algorithm utilizing intermediate layer features in deep learning [19]. Zhang et al. employed self-regularization to the semantic segmentation of commonly mislabeled objects [20]. Chen et al. proposed an image inpainting method based on a new encoder

combined with a context loss function [21]. Semantic segmentation methods based on DeepLabv3+, exhibiting grouped convolutions and multi-branch architectures, have also been used to modify bottleneck blocks in the backbone network while keeping parameter quantities nearly constant.

In this study, we designed a multi-scale hierarchical attention module (MHAM) and an ASPP parallel structure to produce a new semantic segmentation network exhibiting multi-branch structures and multi-scale modules (MBMS). The primary contributions of this work are as follows:

(1) The backbone bottleneck block in DeepLabv3+ was reconstructed. First, the standard convolution for the bottleneck layer in the middle of each block was adapted to perform group convolutions. Channels in the bottleneck block were then doubled, which not only preserved parameter quantities but also compensated for decreased resolution in the feature map due to an increased number of layers in the network during down-sampling. As a result, the network could extract more detailed feature information.

(2) A multi-branch architecture was applied to each modified bottleneck block. During the training period, 1×1 convolutional branches were added to the bottleneck blocks to fuse rich image features. This prevented any issues caused by too few branches in the refining of image features. The structure was then re-parameterized in the inference phase by merging the identity and 1×1 convolutional branches into a 1×1 convolution, to improve inference speed.

(3) A multi-scale hierarchical attention module was introduced to robustly enhance features between adjacent channels and locations. It was connected in parallel with the ASPP to increase segmentation disadvantages for convolutions with larger atrous rates. This was done to further enrich features in network output images exhibiting low-resolution maps or small targets.

(4) Data augmentation was performed on the unmanned aerial vehicle (UAV) AeroScapes dataset to increase the diversity of data, as measured by size and morphology. This improved segmentation accuracy for small and occluded targets. Transfer learning was also performed during training to accelerate convergence and improve model robustness.

The remainder of this article is organized as follows. Materials and methods are presented in Section 2. The proposed semantic network model is described in Section 3 and the training process and ablation experiments are introduced in Section 4. Experimental results and a comparative analysis are provided in Section 5 and Section 6 concludes the paper.

Related work

Multi-branch architecture

Deep convolutional networks can perform multiple convolution or pooling operations on input images in parallel, splicing the outputs into a feature map [22-24]. The depth and width of such networks can be increased without requiring longer computational runtimes, while multi-branch topologies can extract more feature information from input images. ResNet [25] uses a two-branch structure while DenseNet [26] connects shallow features with high-level information to generate a more complex topology. High-performance CNNs can be constructed from neural architecture search (NAS) [27-30] and manual space designs [31], in a process similar to efficient neural architecture search (ENAS) [32]. However, large quantities of calculations and human resources are required.

Furthermore, the large size of some models generated by NAS prevents them from being trained using ordinary graphical processing units (GPUs). Considering that complex models reduce the degree of parallelism and limit model speed for inferencing, we propose a multi-branch architecture with grouped convolutions. This model is based on a ResNet and is implemented in the training process. During the inference phase, a multi-branch structure is re-parameterized and adjusted to improve the accuracy and speed of segmentation.

Multi-scale aggregation

Semantic segmentation considers the scale of individual objects and utilizes multi-scale aggregation to adapt to images of different sizes. Farabet et al. [33] trained multi-scale CNNs using original pixels and extracted feature vectors encoded in dense multi-scale regions centered on each pixel. Lin et al. [34] used an image pyramid method involving multi-resolution images as input to fuse features at multiple scales, achieving an intersection-over-union (IoU) score of 78.0 for the PASCAL VOC 2012 dataset. Liu et al. [35] proposed a multi-scale patch generator to produce patches from the original image or convolution feature map, thereby improving model generalizability. Pinheiro et al. [36] input images of various sizes into different layers of a recursive CNN, to improve segmentation results.

Although these techniques improved segmentation accuracy, they require large calculation quantities and extensive GPU memory. Other correlation methods [37-40] have used a simple summation operation to fuse feature maps at different scales, ignoring some essential features. In the proposed model, a multi-scale hierarchical attention module was connected in parallel with ASPP to focus on small objects and overcome the limitations of conventional atrous convolution. Segmentation results for the AeroScapes dataset improved as a result.

Encoder-decoder

The encoder-decoder structure has been widely used for object detection and semantic segmentation in computer-vision tasks. In the encoder module, a feature map exhibiting semantic information is generated from an input image through neural network learning. In the decoder module, object details and spatial dimensions for low-resolution feature maps (produced by the encoder module) are gradually restored and used to classify pixels. Coarse high-level and delicate low-level information are merged in FCN networks when restoring low-resolution feature maps. Deconvolution [41] is then applied to up-sampling and end-to-end learning is performed through the back-propagation of pixel loss. Noh et al. [42] proposed a deconvolutional network mirrored by a convolutional network, which combined de-pooling with deconvolution to expand activation. A semantic segmentation map was then output with the same size as the input image. Unlike the models discussed above, SegNet stores the index of the largest position in a pooling process and a decoder uses these stored pooling indexes to up-sample the feature map. The U-Net expansion path deconvolves a combination of high-resolution feature maps output by the shrinking path process (combined with the output from up-sampling) to produce more accurate results. Ghiasi et al. [43] reconstructed a network architecture and used a Laplacian pyramid to refine the boundaries of low-resolution feature maps. The performance of these networks demonstrated the vital role of an encoder-decoder structure in semantic segmentation. In this study, an encoder-decoder design was used to improve the restoration of low-resolution feature maps.

Proposed semantic segmentation network

Network structure

The DeepLabv3+ network uses an encoder-decoder architecture with shallow features and high-level semantic information. The encoder is comprised of a deep CNN and ASPP, while the decoder performs feature restoration after fusing low-level features. An atrous convolution is used to extract features from objects or scenes in an input image, followed by 1×1 and 3×3 convolutions with atrous rates of 6, 12, and 18. Global average pooling is then applied to extract and distinguish features at different scales. High-level semantic feature maps acquired in the decoding process are then fused with shallow features to

compensate for boundary information lost in the down-sampling process. Finally, linear interpolation is used to restore object boundaries and produce semantic segmentation maps.

Edge information is often lost or blurred in DeepLabv3+ because of multiple down-sampling steps in the backbone network. Segmentation accuracy for small objects also suffers because of the atrous nature of ASPP. In view of this, we propose a new semantic segmentation network with multi-branch structures and multi-scale modules (MBMS DeepLabv3+), the design of which is shown in Fig 1. An encoder-decoder architecture was utilized in which ordinary convolutions were replaced by grouped convolutions in the bottleneck layer of the DeepLabv3+ backbone network ResNet50. The number of channels was also doubled and a multi-branch architecture was included (the new backbone network was named ResNet50_M). The MHAM module and ASPP were connected in parallel and a 1×1 convolution was used to enhance the nonlinearity of the encoder structure.

In the proposed network, the decoder accepts shallow features from the ResNet50_M encoder and uses 1×1 convolutions to reduce channel dimensions, such that the number of channels in the feature map is equal to that of the feature map obtained by four up-sampling layers. After fusing shallow features with deep up-sampled features, 3×3 convolutions are used to refine the feature map. Up-sampling is then performed four times to produce an output with the same resolution as that of the original image. Deepening of the MBMS DeepLabv3+ allows the multi-branch architecture to acquire more robust feature representations in deep layers of the network and prevent vanishing gradients. The MHAM module also reduces missing or incorrect segmentation results for small objects.

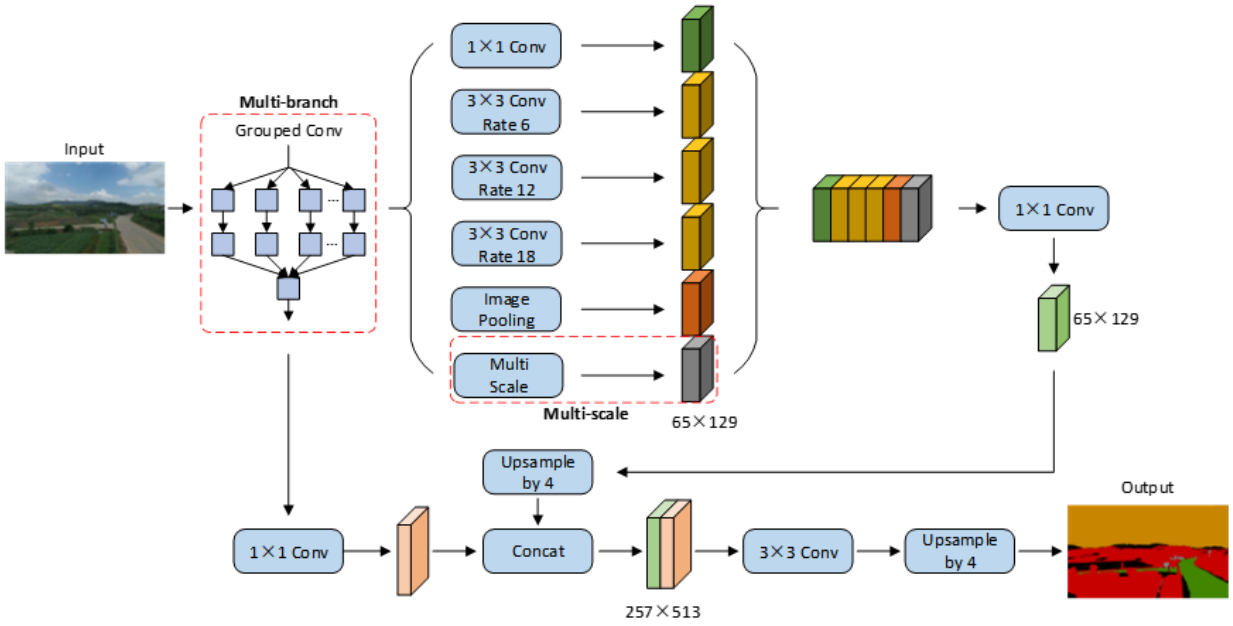

Figure 1. The structure of the proposed MBMS DeepLabv3+ model.

Grouped convolution

We propose modifying standard convolutions in the central bottleneck layer of a ResNet50 block to grouped convolutions, to minimize computational costs while improving segmentation accuracy. In deep CNNs, weighted operations performed by neurons are fundamental transformations produced by fully connected and convolutional layers. These can be expressed as an aggregation transformation:

$$y = \sum_{i=1}^{D} \omega_i x_i \tag{1}$$

where $\mathrm{x} = [x_1, x_2, \mathrm{L}\ , x_D]$ is the D-channel input vector in a neuron, ω_i is the filter weight for the i^{th} channel, and y is the output. The function $T_i(\mathrm{x})$ was used instead of a simple neuronal transformation $\omega_i x_i$. As a result, Eq. (1) can be expressed as:

$$y = \sum_{i=1}^{C} T_i(\mathrm{x}) \tag{2}$$

where $T_i(\mathrm{x})$ can represent any function with a cardinality of C, which denotes the size of a set of aggregated transformations. The dimensionality of this term controls the number of conversions and contributes more to the final segmentation result than the width or depth values. The transformation T_i is similar to that of a simple neuron, where x is projected into a low-dimensional space and a transformation is then performed.

In VGG architectures, the same layers are used repeatedly. Inspired by this, we designed a transformation function with the same topology in each layer. The transformation function T_i then assumed a bottleneck-shaped architecture [25], as illustrated in Fig 2. Using Eq. (2) as the residual function then gives:

$$y = \mathrm{x} + \sum_{i=1}^{C} T_i(\mathrm{x}) \tag{3}$$

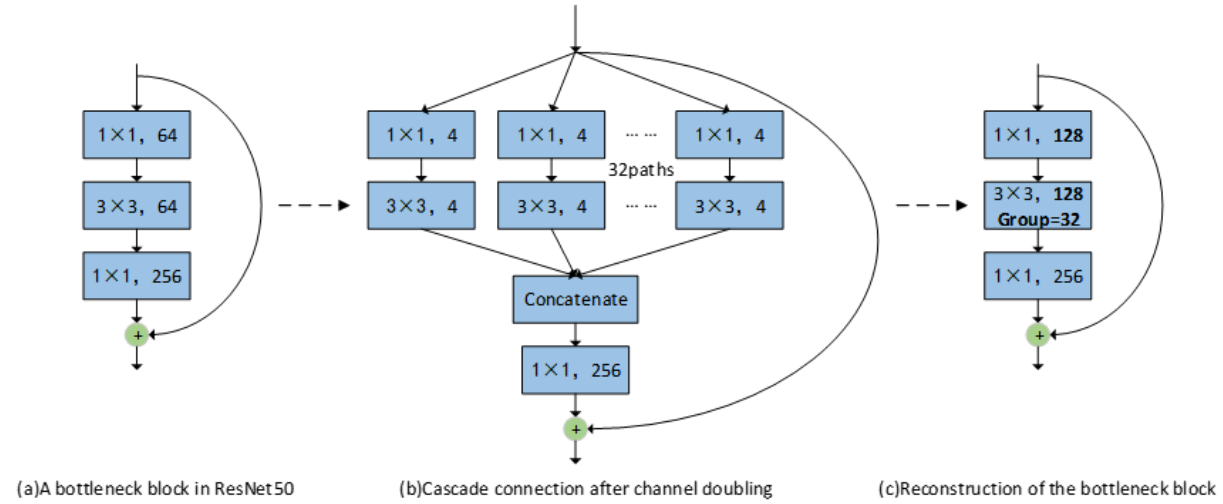

Figure 2. A schematic diagram of a grouped convolution in a bottleneck block.

Fig 2(a) shows a bottleneck block in the ResNet50 DeepLabv3+ backbone network. Fig 2(b) shows a bottleneck block connected in cascade after doubling the number of bottleneck-block channels. The same topology was used in multiple paths and a grouped convolution [44] (with 32 groups) was designed to minimize consumption. This reconstructed bottleneck block is shown in Fig 2(c), where a single wide embedding layer (e.g., 1×1, 128-dimensions) was used to replace all 1×1 low-dimensional embedding layers. The grouped layer included a 32-group convolution with four input and output channels. The output of each group was stitched to form the output of the layer. Compared to Fig 2(a), the connections in Fig 2(c) are broader and sparser and the grouped convolutions make the bottleneck block more concise. In Fig 2(a), the number of parameters in the original bottleneck block is $256 \cdot 64 + 3 \cdot 3 \cdot 64 \cdot 64 + 64 \cdot 256 \approx 70\mathrm{k}$ and the number of parameters in Fig 2(c) is given by:

$$C \cdot \left(256 \cdot d + 3 \cdot 3 \cdot d \cdot d + d \cdot 256\right) \tag{4}$$

When $C = 32$ and $d = 4$, the number of parameters is approximately 70k. The cardinality C and bottleneck width d are provided for the same number of parameters in Table 1. Table 2 compares the number of bottleneck block parameters without grouped convolutions (denoted ResNet50_NG) to those with grouped convolutions (denoted ResNet50_WG) in ResNet50. The results show that although the number of channels increased, the complexity of the model did not change significantly.

Table 1. The values of cardinality C and width d.

cardinality C	1	2	4	8	16	32
width of bottleneck d	64	40	24	14	8	4

Table 2. The model capacity of the bottleneck block of ResNet50 and ResNet50 using grouped convolution.

Stage	**ResNet50_NG**	**ResNet50_WG**
1	$3\times\begin{bmatrix}1\times1,64\\3\times3,64\\1\times1,256\end{bmatrix}$	$3\times\begin{bmatrix}1\times1,128\\3\times3,128\\1\times1,256\end{bmatrix},C=32$
2	$4\times\begin{bmatrix}1\times1,128\\3\times3,128\\1\times1,512\end{bmatrix}$	$4\times\begin{bmatrix}1\times1,256\\3\times3,256\\1\times1,512\end{bmatrix},C=32$
3	$6\times\begin{bmatrix}1\times1,256\\3\times3,256\\1\times1,1024\end{bmatrix}$	$6\times\begin{bmatrix}1\times1,512\\3\times3,512\\1\times1,1024\end{bmatrix},C=32$
4	$3\times\begin{bmatrix}1\times1,512\\3\times3,512\\1\times1,2048\end{bmatrix}$	$3\times\begin{bmatrix}1\times1,1024\\3\times3,1024\\1\times1,2048\end{bmatrix},C=32$
Number of Parameters	21.4×10^6	21.5×10^6

Multi-branch architecture

The ResNet model can be considered a collection of shallow models [45], due to a multi-branch architecture that avoids vanishing gradient issues after training. We propose a structured re-parameterization method based on ResNet, developed by adding 1×1 convolutional branches to the bottleneck block. This approach helps to prevent vanishing gradients in deep networks and improves network converge. In addition, various gradient flow paths were added to the network because the residual structure includes multiple branches. These networks were then integrated into a single structure, making training and model integration more efficient.

Identity and 1×1 convolution branches were used to construct the bottleneck block during the training period. A structural re-parameterization technique was included to fuse 1×1 convolutions during the inferencing phase. In this structural re-parameterization process, the identity branch was treated as a degenerate 1×1 convolution and the multi-branch architecture was transformed after training. The advantages of this approach are as follows. (1) Reducing the number of branches can accelerate model inference speed. (2) Hardware memory utilization is typically improved. (3) The re-parameterized structure is more flexible making it easier to adjust the width of each layer. Fig 3 shows the re-

parameterization process for the multi-branch architecture including structural parameters. Here, $C_1 = C_2 = 2$ and the kernel for the 1×1 layer is a 2×2 matrix.

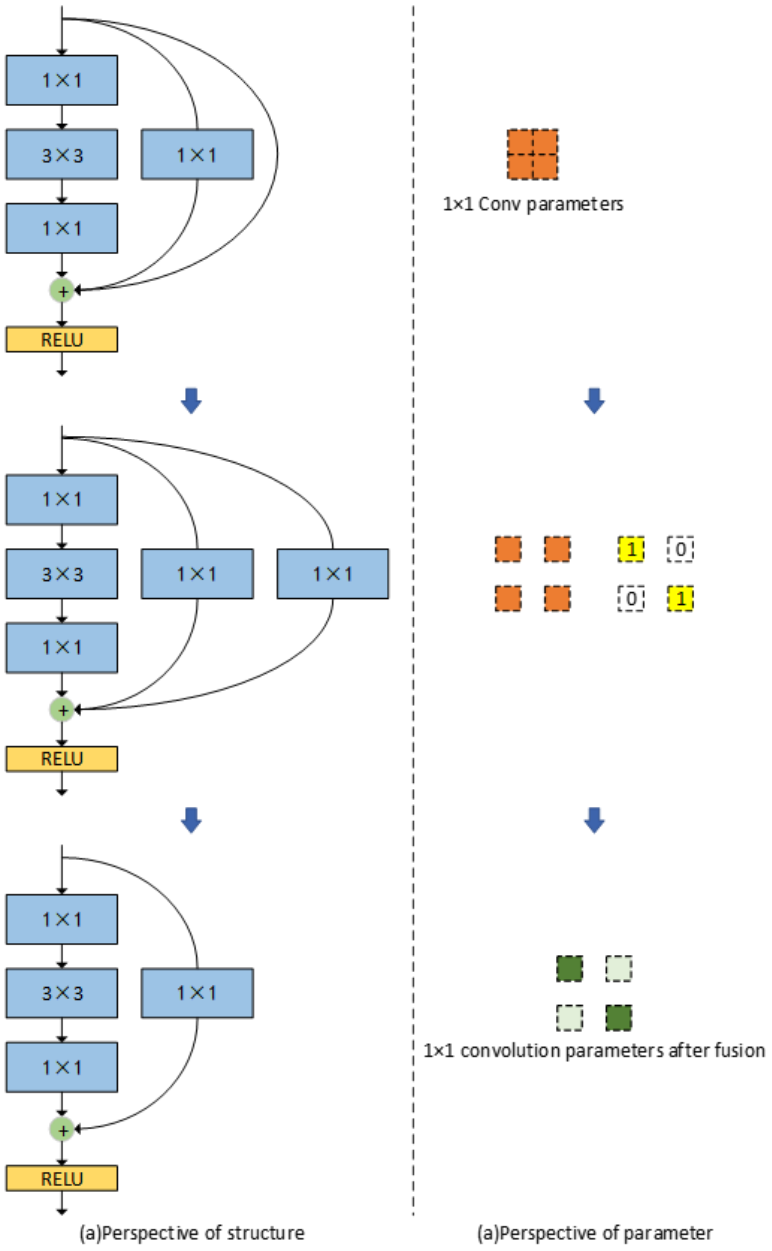

Figure 3. The re-parameterized structure of the multi-branch architecture.

Information flow in the training period was modeled as $y = \mathrm{x} + g(\mathrm{x}) + f(\mathrm{x})$, where x, $g(\mathrm{x})$, and $f(\mathrm{x})$ are implemented using the identity branch, 1×1 convolutions, and primary information flow in the ResNet bottleneck block, respectively. The model was constructed in the training period by stacking similar blocks, which can be viewed as a collection of 3^n members with n blocks. Information flow in the inference phase can be equivalently transformed into $y = f(\mathrm{x}) + h(\mathrm{x})$, where h is implemented using a fused 1×1 convolutional layer.

The term $W^{(1)} \in ¡^{C_2 \times C_1 \times 1 \times 1}$ represents the kernel of a 1×1 convolutional branch with C_1 input channels and C_2 output channels. In addition, $\mu^{(1)}$, $\sigma^{(1)}$, $\gamma^{(1)}$, and $\beta^{(1)}$ represent the accumulated mean, standard deviation, learned scaling factor, and bias in the batch normalization (BN) layer after a 1×1 convolution, respectively. The terms $\mu^{(0)}$, $\sigma^{(0)}$, $\gamma^{(0)}$, and $\beta^{(0)}$ specifically refer to the identity branch. Inputs and outputs are represented by $M^{(1)} \in ¡^{N \times C_1 \times H_1 \times W_1}$ and $M^{(2)} \in ¡^{N \times C_2 \times H_2 \times W_2}$, respectively, and * denotes a convolution operation. In the case of $C_1 = C_2$, $H_1 = H_2$, and $W_1 = W_2$, the fusion process for 1×1 convolutions and identity residual structures in the reasoning period can be represented as follows:

$$\begin{aligned} M^{(2)} &= bn\left(M^{(1)} * W^{(1)}, \mu^{(1)}, \sigma^{(1)}, \gamma^{(1)}, \beta^{(1)}\right) \\ &+ bn\left(M^{(1)}, \mu^{(0)}, \sigma^{(0)}, \gamma^{(0)}, \beta^{(0)}\right) \end{aligned} \tag{5}$$

where the BN function in the inference stage is given by:

$$bn\left(M, \mu, \sigma, \gamma, \beta\right)_{:,i,:,:} = \left(M_{:,i,:,:} - \mu_i\right)\frac{\gamma_i}{\sigma_i} + \beta_i \tag{6}$$

and $\forall 1 \le i \le C_2$. Each of the convolutional and BN layers following this step were converted to convolutions with deviations:

$$W'_{i,:,:,:} = \frac{\gamma_i}{\sigma_i} W_{i,:,:,:}, \; b'_i = -\frac{\mu_i \gamma_i}{\sigma_i} + \beta_i \tag{7}$$

where $\{W', b'\}$ denotes the kernel and deviation transformed by $\{W, \mu, \sigma, \gamma, \beta\}$, $\forall 1 \le i \le C_2$. As a result:

$$bn\left(M * W, \mu, \sigma, \gamma, \beta\right)_{:,i,:,:} = \left(M * W'\right)_{:,i,:,:} + b'_i \tag{8}$$

Since identity mapping can be considered a 1×1 convolution utilizing an identity matrix as the kernel, the identity branch can be converted into a 1×1 convolution. Two 1×1 kernels and two deviation vectors were then acquired and the final deviation vector was obtained by summing the two deviation vectors. The final output of the inference period was then:

$$\begin{aligned} M^{(2)} &= \left(M^{(1)} * W^{(1)\prime}\right) + b_i^{(1)\prime} + \left(M^{(1)} * W^{(0)\prime}\right) + b_i^{(0)\prime} \\ &= \left[M^{(1)} * \left(W^{(1)\prime} + W^{(0)\prime}\right)\right] + \left(b_i^{(1)\prime} + b_i^{(0)\prime}\right) \\ &= \left(M^{(1)} * W\right) + b \end{aligned} \tag{9}$$

where $W^{(0)} \in ¡^{C_2 \times C_1}$ denotes the kernel of the identity branch and W and b represent the kernel and bias for the fused 1×1 convolution, respectively.

Multi-branch architecture with grouped convolutions

We designed a multi-branch architecture with grouped convolutions, based on the proposed grouped convolution and multi-branch structure (i.e., ResNet50_M). The architecture in each stage is shown in Fig 4, where panel (a) represents the training-period architecture, (b) shows the inference-stage architecture, i denotes the i th stage of ResNet50_M, and n is the block number. If the input channel is 128×i, the output channel is 128×i×2, where i =1, 2, 3, 4 and n =2, 3, 5, 2. Fusing the 1×1 convolution and identity branches from the training period into a 1×1 convolution during the inference period increased the training speed.

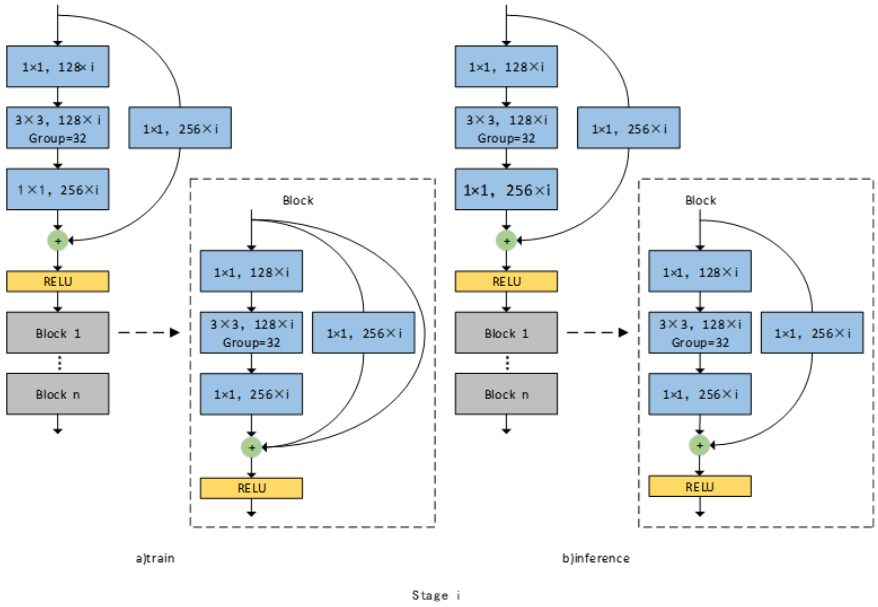

Figure 4. The four ResNet50_M stages.

Multi-scale hierarchical attention module

In semantic segmentation, the prediction of large objects or global images must be performed at a relatively low inferential resolution, to allow the receptive field to capture more background regions. In contrast, the prediction of image details should be performed at higher inferential resolutions. We addressed the adaptive prediction of objects at different scales by applying a multi-scale prediction based on the pixel levels and proposing a multi-scale hierarchical attention module (MHAM), for which the network structure is shown in Fig 5. The MHAM can be classified as two modules: multi-scale hierarchical channel attention and multi-scale hierarchical position attention. The network is executed hierarchically to integrate multiple scales and make predictions.

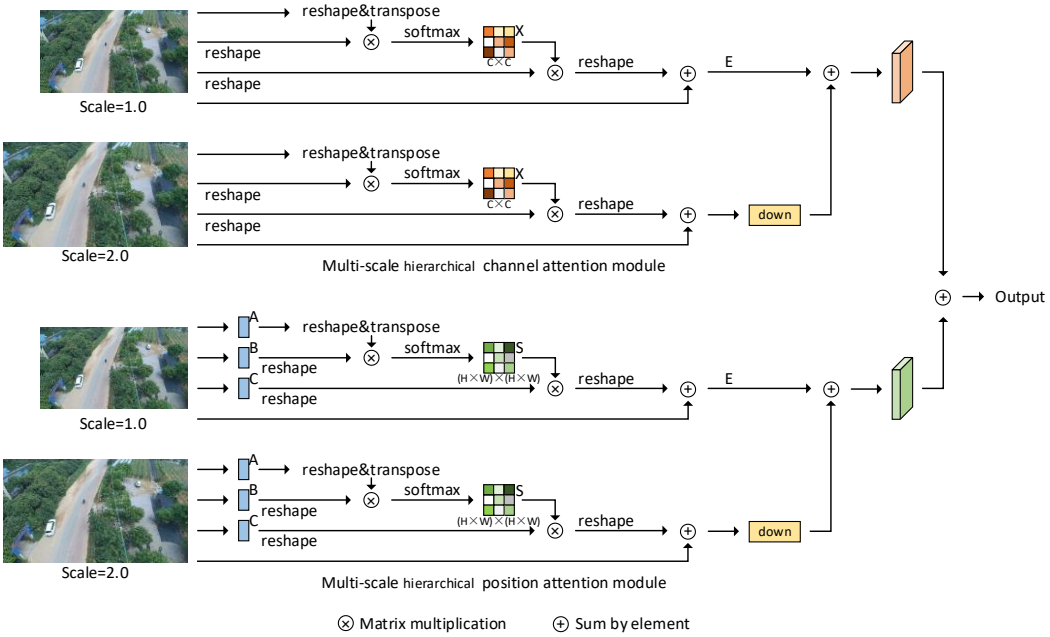

Figure 5. The multi-scale hierarchical attention module.

The channel attention module reflects the dependencies of different model channels. High-level semantic feature maps in different channels can be considered predictions for a specific category, with inter-relative semantics. Interdependencies between different channel feature maps were then emphasized and specific semantic features were promoted as follows. The channel attention map $X \in R^{C\times C}$ was first calculated from the input feature map $A \in R^{C\times H\times W}$. We then changed the shape of A to $A' \in R^{C\times N}$, where $N = H \times W$, and then multiplied A' by its transpose matrix $\left(A'\right)^T$. Finally, a softmax layer was included to produce the channel attention map $X\left[x_{ji}\right]$:

$$x_{ji} = \frac{\exp\left[A_i' \cdot \left(A'\right)^T{}_j\right]}{\sum_{i=1}^{C}\exp\left[A_i' \cdot \left(A'\right)^T{}_j\right]} \tag{10}$$

where A_i' is the i th element value of the matrix $A' \in R^{C\times N}$, $\left(A'\right)^T{}_j$ is the j th element value of $\left(A'\right)^T$, and x_{ji} is the effect from the i th to the j th channel. The output $E \in R^{C\times H\times W}$ is then given by:

$$E_j = \alpha\sum_{i=1}^{C}\left(x_{ji}A_i\right) + A_j \tag{11}$$

where α is a scale parameter initialized to 0. The final output feature for each channel is a feature-weighted sum of the original and other channels. Channel attention mechanisms can be used to model the long-term semantic dependencies between different feature mappings and enhance the discriminability of features.

The position attention module can also simulate contextual relationships between local features to enhance similar features at different locations. A convolution operation can be applied to the feature map A to obtain three new feature maps: B, C, and D, such that $\{B, C, D\} \in R^{C\times H\times W}$. These maps can then be changed to B', C', and $D' \in R^{C\times N}$, where $N = H \times W$. The inclusion of a softmax layer produces the position attention map S:

$$s_{ji} = \frac{\exp\left(B_i \cdot C_j\right)}{\sum_{i=1}^{N}\exp\left(B_i \cdot C_j\right)} \tag{12}$$

where s_{ji} is the effect from the i th position to the j th position and N is the number of pixels. The final output $E \in R^{C\times H\times W}$ is given by:

$$E_j = \beta\sum_{i=1}^{N}\left(s_{ji}D_i\right) + A_j \tag{13}$$

where β is a scale parameter initialized to 0. This expression indicates the final output feature for each position is a weighted sum of the original and all other position features. Thus, the position attention mechanism exhibits a global contextual perspective and can selectively weight aggregated contexts based on a position attention graph. Similar semantic features can be reinforced, and the semantic consistency maintained.

Parallel and cascaded MHAM modules with ASPP

Over-expansion in ASPP was addressed by connecting the MHAM in parallel or cascade with an ASPP to extract edge features from small objects and simulate correlation features for large objects. The resulting network structure similar to that described by Liu et al. [46] is shown in Fig 6 where panel (a) is in a parallel style and panel (b) is in a cascaded style.

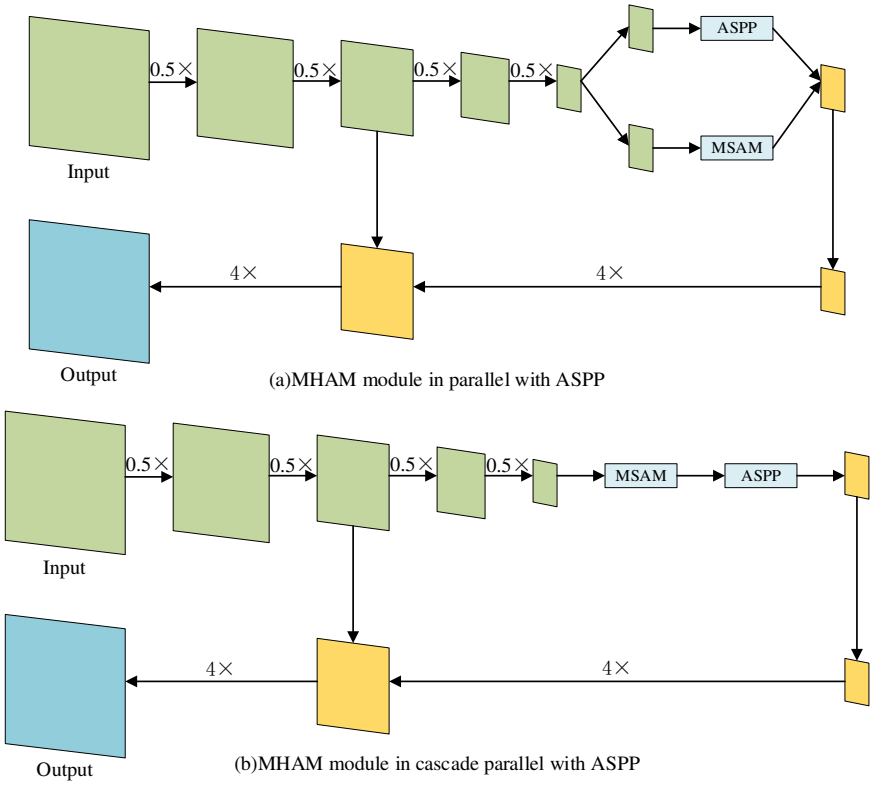

Figure 6. The two MHAM and ASPP structures.

In Fig 6(a), image features were extracted by the backbone network, and the model was divided into two branches, to process extracted feature maps separately prior to fusion. In Fig 6(b), the feature maps were extracted by a deep CNN. The MHAM module then enhanced individual pixel features and ASPP was used for multi-scale object segmentation.

Data augmentation

Data augmentation is crucial in image segmentation and is often used with an insufficient dataset or for models with large parameter quantities. To some extent, data augmentation can effectively solve problems resulting from an unbalanced distribution of data or too few category objects, thereby improving the robustness and generalizability of the model. Here, we primarily used flip, rotation, Gaussian blur, and image color dithering. The seven data augmentation operations are shown in Fig 7.

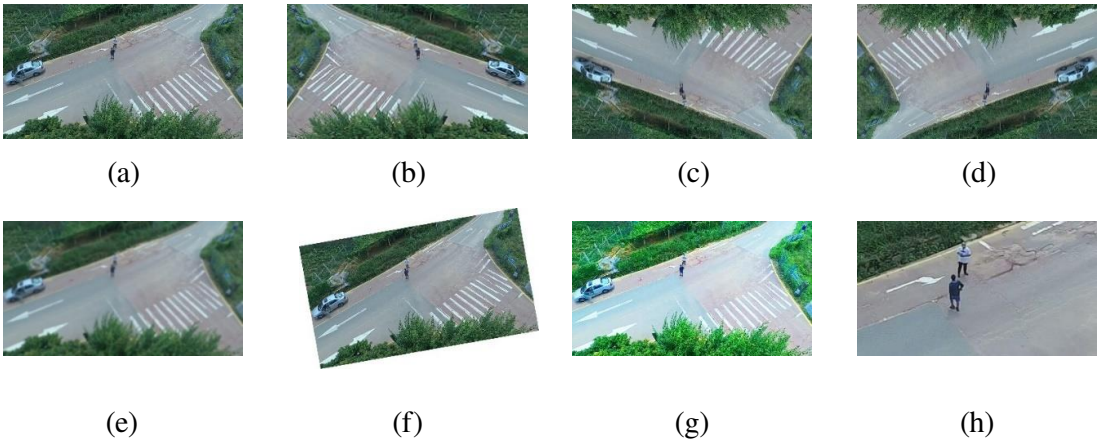

Figure 7. Schematic representations of data augmentation. (a) The original image, (b) horizontal flip, (c) vertical flip, (d) horizontal and vertical flip, (e) Gaussian blur applied to the original image using a radius

of 4, (f) counterclockwise rotation by 10°, (g) color jitter applied to the image with a brightness, saturation, and contrast of 50, and (h) a randomly magnified image area.

Training and ablation experiment

The AeroScapes dataset [47] was used to train the network. Specifically, a set of 3,269 aerial scene images were extracted from 141 video sequences (captured by UAVs) with a resolution of 1280×720. These images contained one background class and 11 foreground object classes (i.e., person, bike, car, drone, boat, animal, obstacle, construction, vegetation, road, and sky). Augmentation produced a dataset with a sample size of 6,538, which was divided into three parts at a ratio of 8:1:1 for training, validation, and testing, respectively.

Comparative experiments with multiple backbones

The feature extraction network was pre-trained using the CityScapes dataset and the pre-training model was transferred to the proposed network to accelerate convergence and stabilize training. This learning transfer effectively prevented problems such as gradient disappearance or gradient explosion, which allowed the neural network to converge faster and more effectively, thereby improving learning efficiency and robustness. A group of feature extraction networks were selected following the DeepLabv3 training protocol, including Xception65, ResNet50, and ResNet50_M (i.e., the "poly" policy [48]). The number of samples per input network were set to 12, 6, and 6, respectively. The initial learning rate was set to 0.1 and the momentum was set to 0.9. Training was performed on four 1080Ti GPUs and was halted after the loss converged to 0.05209, 0.03906, and 0.03615 for each number of samples per input network. A control variable approach was used, followed by a series of comparison experiments. Table 3 shows the results of comparisons for three backbone network attenuation steps and mean IoU (mIoU) [49] metric values calculated as:

$$\mathrm{mIoU} = \frac{1}{k+1}\sum_{i=0}^{k}\frac{Y_{ii}}{\sum_{j=0}^{k}Y_{ij}+\sum_{j=0}^{k}Y_{ji}-Y_{ii}} \tag{14}$$

where $k+1$ denotes k object categories and one background category, Y is the number of pixels, and Y_{ij} is the pixel belonging to category i but predicted to belong to category j. Y_{ji} is the pixel belonging to category j but predicted to belong to category i, while Y_{ii} indicates a pixel that is predicted correctly. Larger mIoU values represent more accurate prediction results.

Table 3. Performance comparison of three different backbone networks.

Backbone	Attenuation steps	mIoU/%
Xception65	30000	41.16
Resnet50	30000	37.13
ResNet50_M	60000	75.57

It is evident from Table 3 that the total number of attenuation steps required by ResNet50_M was 60,000, while that of the original DeepLabv3+ was 30,000. The accuracy of ResNet50_M for the test set was 75.57%, which is 34.41% and 38.44% higher than the original DeepLabv3+ with Xception65 and ResNet50, respectively. These comparisons verify that group convolutions can reduce the number of training parameters and prevent overfitting. The multi-branch architecture can also extend the receptive field and improve segmentation accuracy.

Ablation experiment with attention module

Atrous convolutions in ASPP produced a discontinuity in the convolution kernel, preventing all pixels from being calculated and making dense prediction results unsatisfactory at the pixel level. Since atrous

convolutions offer the advantage of extracting long-range information, they are suitable for segmentation of large objects. For this reason, we propose a multi-scale hierarchical attention module. Ablation experiments were conducted with both ASPP in parallel and cascaded, to determine its necessity and potential for improvement. Corresponding mIoU values are shown in Table 4. These results illustrate mIoU values are highest when the MHAM and ASPP are connected in parallel.

Table 4. ASPP performance when connected with MHAM in parallel and cascade.

Network model	mIou/%
ASPP	41.16
MHAM and ASPP in parallel	47.83
MHAM and ASPP in cascade	43.65

Segmentation experiments and comparative analysis

The effectiveness of the proposed MBMS DeepLabv3+ network was verified using the AeroScapes dataset. Semantic segmentation results were compared with DeepLabv3+, U-Net, RefineNet, PSPNet, DADA [50], DSRL [51], HANet [52], SegFormer [53], and SETR [54]. mPA [49] and mIoU were used as segmentation evaluation metrics, with the latter calculated as follows:

$$\mathrm{mPA} = \frac{1}{k+1}\sum_{i=0}^{k}\frac{Y_{ii}}{\sum_{j=0}^{k} Y_{ij}} \tag{15}$$

Segmentation results for different object types

Twelve object categories were segmented, including: person, bike, car, drone, boat, animal, obstacle, construction, vegetation, road, and sky. Resulting mIoU values, obtained using different backbone networks, are shown in Table 5. As seen in the table, DeepLabv3+ with an Xception65 backbone produced an mIoU value of 0 for four categories: bike, drone, boat, and animal. DeepLabv3+ with ResNet50 produced an mIoU of 0 for six categories: bike, car, drone, boat, animal, and obstacle. The lowest mIoU values occurred for excessive numbers of unidentified categories. It is worth noting the proposed MBMS DeepLabv3+ network effectively identified small-scale objects like bike, car, boat, and animal. In addition, the accuracy improved for each of these categories, while that of boat and animal increased significantly.

Table 5. MIoU of the three structures for 12 categories of objects.

Method	Xception65 DeepLabv3+	ResNet50 DeepLabv3+	MBMS DeepLabv3+
Background	72.62	78.27	**92.21**
Person	51.55	27.29	**80.50**
Bike	0	0	**67.80**
Car	61.13	0	**91.00**
Drone	0	0	**71.60**
Boat	0	0	**89.80**
Animal	0	0	**73.69**
Obstacle	20.64	0	**70.80**
Construction	60.62	65.29	**89.42**
Vegetation	87.96	91.28	**96.32**
Road	87.83	92.23	**97.70**
Sky	59.98	95.28	**98.78**

Six representative images were used to visualize the results from three models. These images included objects at different scales and are shown in Fig 8, where panel (a) displays the original images, (b) the ground truth, (c) segmentation results for Xception65 DeepLabv3+, (d) results from ResNet50 DeepLabv3+, and (e) results from the proposed MBMS DeepLabv3+. Objects to be identified are outlined in red in all images.

The proposed MBMS DeepLabv3+ network produced the best segmentation results for the obstacle embedded in the vegetation and the two small objects (person and car) in the figure at the top of panel (a). Xception65 DeepLabv3+ and ResNet50 DeepLabv3+ failed to adequately segment these structures. Results for the second image showed that MBMS DeepLabv3+ completely segmented the obstacle and the person obscured by trees. However, other networks were unable to segment these structures. MBMS DeepLabv3+ also segmented the objects more completely than other networks in the third image. Despite a relatively complex background in the fourth image, MBMS DeepLabv3+ segmented the person and obstacle, although the drone was identified as belonging to the incorrect category. Results from other networks were worse than those of the proposed MBMS DeepLabv3+ network. In the fifth and sixth images, MBMS DeepLabv3+ achieved complete segmentation results for small obstacles, animals, and objects exhibiting feature similarities.

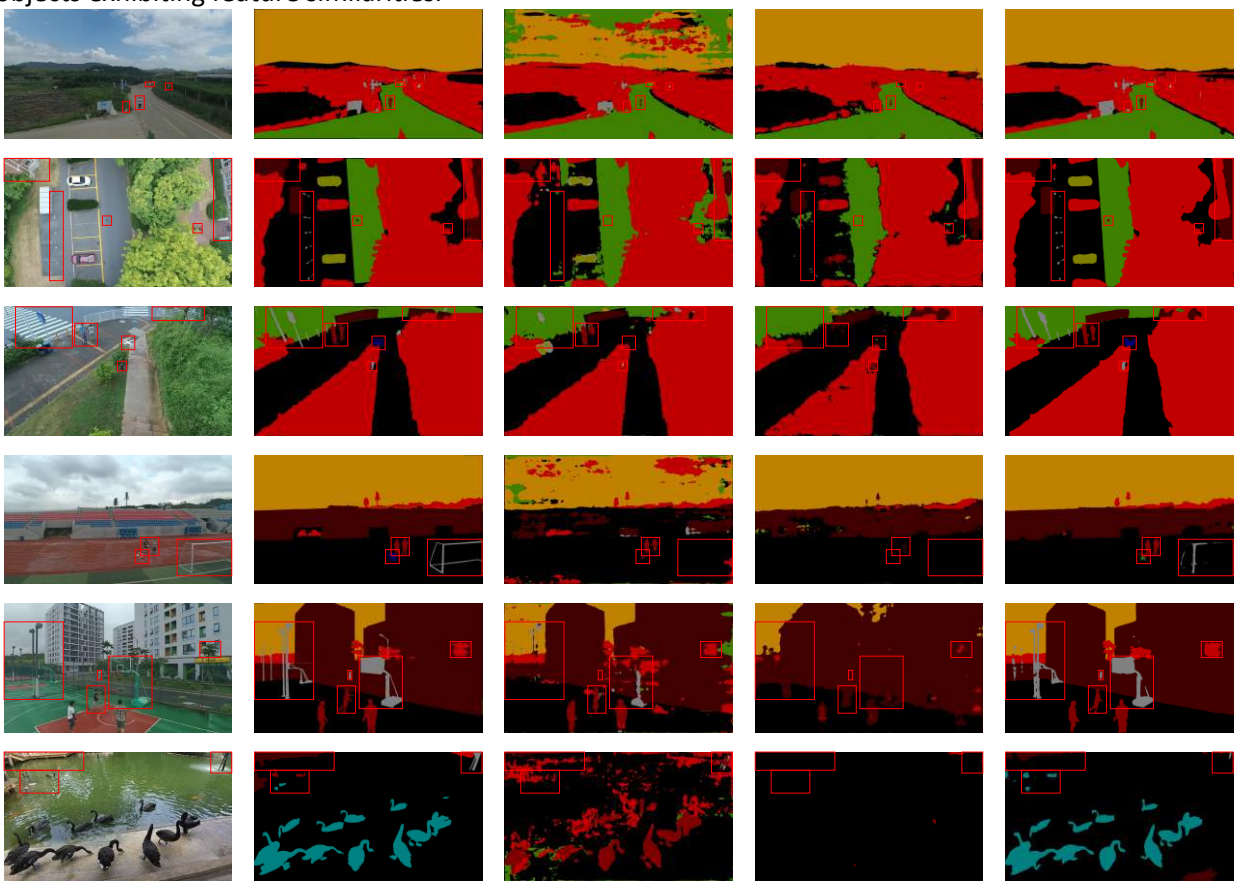

Figure 8. A comparison of segmentation results from three methods. (a) The original image, (b) ground truth, (c) Xception65 DeepLabv3+, (d) ResNet50 DeepLabv3+, and (e) MBMS DeepLabv3+.

Comparison of segmentation results for multiple models in AeroScapes

The segmentation accuracy of the proposed MBMS DeepLabv3+ network was further evaluated by comparing it with nine advanced models, as shown in Table 6.

Table 6. Segmentation performance metrics for seven networks applied to the AeroScapes dataset.

Model	mIoU	Accuracy
U-Net	59.07	68.33
RefineNet	63.09	70.82
PSPNet	80.88	88.18
DADA	81.53	88.75
DSRL	82.48	89.72
HANet	83.07	89.93
MBMS DeepLabv3+	84.98	97.57

The table shows that segmentation accuracy for the proposed network was higher than that of other semantic segmentation models applied to the AeroScapes dataset. The results of visual comparison experiments in Fig 9 show that our model achieved the highest recognition accuracy for small or occluded objects, while refining image edges.

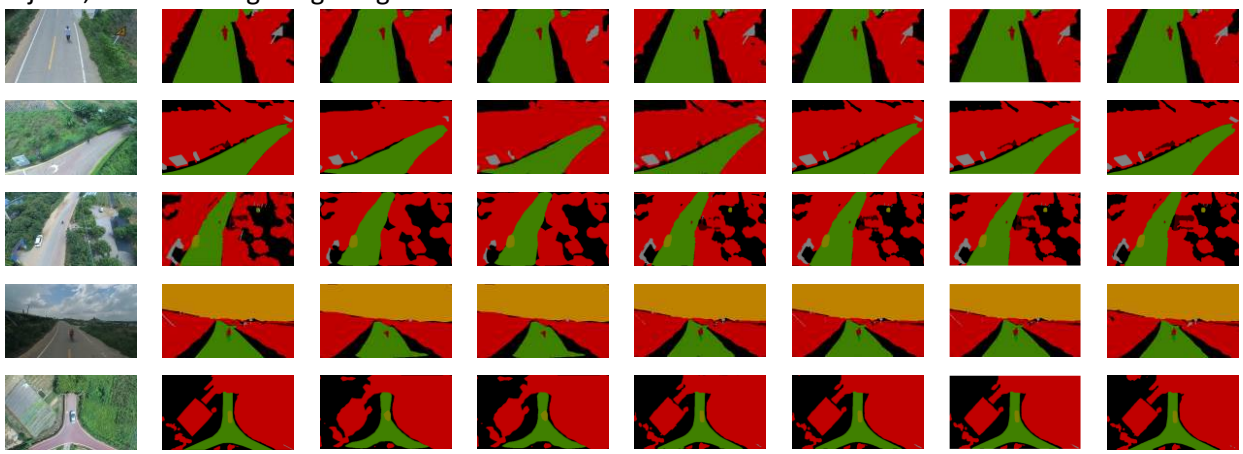

Figure 9. A visual comparison of seven methods. (a) The original image, (b) ground truth, (c) U-Net, (d) SegFormer-B5, (e) SETR-PUP, (f) DADA, (g) HANet, and (h) MBMS DeepLabv3+.

Comparison of segmentation results for multiple models on other datasets

In addition to the AeroScapes dataset, the proposed network was also evaluated using the CityScapes and ADE20K datasets, as shown in Table 7. All experimental results are based on a test set with the same training environment and configuration. Measured mIoU values reflect the accuracy of the network.

Table 7. Segmentation performance metrics for six networks applied to the CityScapes and ADE20K datasets.

Model	CityScapes (mIoU)	ADE20K (mIoU)
RefineNet	73.63	40.75
PSPNet	78.50	44.48
SETR-PUP	82.15	50.28
SegFormer-B4	83.27	51.21
SegFormer-B5	83.95	51.83
MBMS DeepLabv3+	84.96	52.72

These results demonstrate that our model can achieve an accuracy similar to that of other state-of-the-art networks. For example, MBMS DeepLabv3+ produced an accuracy improvement of 1.01% over SegFormer-B5 for the CityScapes dataset and an improvement of 0.89% for the ADE20K dataset. In summary, the proposed network outperformed conventional models and again demonstrated superior semantic segmentation performance.

Conclusions

A new semantic segmentation model, developed from a DeepLabv3+ framework, was proposed to improve segmentation accuracy for small and obstructed image targets. The network architecture exhibited a multi-branch structure and included multi-scale hierarchical attention. Grouped convolutions were utilized in each bottleneck of the ResNet50 backbone to reduce model size. The network was pre-trained using the CityScapes dataset and then trained and validated using the AeroScapes dataset. A series of validation experiments, involving comparisons with conventional semantic segmentation models, produced the following conclusions:

(1) Grouped convolutions can significantly reduce the number of network parameters and the model size. Channel doubling did not affect the number of parameters but did improve performance.

(2) The multi-branch architecture further fused features from multiple branches during the training period, resulting in a more detailed and comprehensive fusion of feature maps. It also re-parameterized the structure during the inference period and improved segmentation accuracy while increasing inference speed.

(3) Connecting the multi-scale hierarchical attention module in parallel with ASPP significantly improved the accuracy and retention of object edges.

(4) The mIoU (84.98%) and accuracy (97.57%) metrics produced by the proposed MBMS DeepLabv3+ network for the AeroScapes dataset were the highest among tested algorithms.

(5) Visual segmentation results verified the proposed model had the fewest instances of mis-segmentation and missing segmentation for small objects, occluded objects, similar features, and complex background scenes in the AeroScapes dataset.

In summary, the proposed MBMS DeepLabv3+ network can effectively segment small-scale and occluded objects. However, segmentation accuracy must be improved for images and objects with severely deficient edge features. In addition, the application of model compression and network pruning could be used to further optimize the network, making it more lightweight for planned future work.

Availability of data material

The datasets generated or analyzed during this study are available in the reference[47].

References

1. Dai J, He K, Sun J. IMAGE SEMANTIC SEGMENTATION, (2016).
2. Lan M, Zhang Y, Zhang L, Du B. Global Context based Automatic Road Segmentation via Dilated Convolutional Neural Network. Information Sciences 2020, 535, 156-171.
3. Xie X, Wang Z. Multi-scale Semantic Segmentation Enriched Features for Pedestrian Detection. 2018 24th International Conference on Pattern Recognition (ICPR), Beijing, China, 2008; pp. 2196-2201.
4. Wang D. End-to-end Autopilot Based on Branch Network with Auxiliary Task. Application of IC 2019, 36, 50-53.
5. Benini S, Khan K, Leonardi R, Mauro M, Migliorati P. A FAce semantic SEGmentation repository for face image analysis. Data in Brief 2019, 24.

6. Londhe AN, Atulkar M. Semantic segmentation of ECG waves using hybrid channel-mix convolutional and bidirectional LSTM. Biomedical Signal Processing and Control 2021, 63, 148-162.
7. Long J, Shelhamer E, Darrell T. Fully Convolutional Networks for Semantic Segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 2015, 39, 640-651.
8. Weng W, Zhu X. Convolutional Networks for Biomedical Image Segmentation. IEEE Access 2021, 99, 1-1.
9. Badrinarayanan V, Kendall A, Cipolla R. A deep convolutional encoder-decoder architecture for image segmentation. IEEE transactions on pattern analysis and machine intelligence 2017, 39, 2481-2495.
10. Zhao H, Shi J, Qi X, et al. Pyramid Scene Parsing Network. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Hawaii, USA, 2017, pp. 2881-2890.
11. Lin G, Milan A, Shen C, Reid I. Multi-path Refinement Networks for High-Resolution Semantic Segmentation. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp, 5168-5177.
12. Chen LC, Papandreou G, Kokkinos I, Yuille A, Murphy K. Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. Computer Science, 2014, 4, 357-361.
13. Chen LC, Papandreou G, Kokkinos I, Murphy K, Yuille A. Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2018, 40, 834-848.
14. Chen LC, Papandreou G, Schroff F, Adam H. Rethinking atrous convolution for semantic image segmentation, 2017.
15. Chen LC, Zhu Y, Papandreou G, Schroff F, Adam H. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. Springer, 2018; pp.833-851.
16. Xia R, Chen Y, Ren B. Improved anti-occlusion object tracking algorithm using Unscented Rauch-Tung-Striebel smoother and kernel correlation filter. Journal of King Saud University-Computer and Information Sciences. 2022 Feb 19.
17. Chen Y, Liu L, Phonevilay V, Gu K, Xia R, Xie J, Zhang Q, Yang K. Image super-resolution reconstruction based on feature map attention mechanism. Applied Intelligence. 2021 Jul;51(7):4367-80.
18. Li X, He H, Li X, Li D, Cheng G, Shi J, Weng L, Tong Y, Lin Z. Pointflow: Flowing semantics through points for aerial image segmentation. InProceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition 2021 (pp. 4217-4226).
19. Chen Y, Liu L, Tao J, Chen X, Xia R, Zhang Q, Xiong J, Yang K, Xie J. The image annotation algorithm using convolutional features from intermediate layer of deep learning. Multimedia Tools and Applications. 2021 Jan;80(3):4237-61.
20. Zhang D, Zhang H, Tang J, Hua XS, Sun Q. Self-Regulation for Semantic Segmentation. InProceedings of the IEEE/CVF International Conference on Computer Vision 2021 (pp. 6953-6963).
21. Chen Y, Liu L, Tao J, Xia R, Zhang Q, Yang K, Xiong J, Chen X. The improved image inpainting algorithm via encoder and similarity constraint. The Visual Computer. 2021 Jul;37(7):1691-705.
22. Szegedy C, Wei L, Jia Y, Sermanet P, Rabinovich A. Going deeper with convolutions. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015.
23. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the Inception Architecture for Computer Vision. 2015.
24. Szegedy C, Ioffe S, Vanhoucke V, Alemi A. Inception-v4, inception-resnet and the impact of residual connections on learning. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, USA, 2017.

25. He K, Zhang X, Ren S, et al. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016.
26. Huang G, Liu Z, Laurens V. Densely Connected Convolutional Networks. IEEE Computer Society, 2016, 4700-4708.
27. Zoph B, Vasudevan V, Shlens J, Le QV. Learning Transferable Architectures for Scalable Image Recognition. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 2018.
28. Real E, Aggarwal A, Huang Y, Le QV. Regularized evolution for image classifier architecture search. Proceedings of the AAAI Conference on Artificial Intelligence. 2018; pp.4780-4789.
29. Liu C, Zoph B, Neumann M, Shlens J, Hua W, Li LJ. Progressive neural architecture search. European conference on computer vision (ECCV), arXiv, 2018.
30. Tan M, Le QV. Rethinking Model Scaling for Convolutional Neural Networks. International Conference on Machine Learning (97), 2019.
31. Radosavovic I, Kosaraju RP, Girshick R, He K, Dollar P (2020). Designing network design spaces. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, USA, 2020.
32. Pham H, Guan M Y, Zoph B, Le QV, Dean J. Efficient Neural Architecture Search via Parameter Sharing. International Conference on Machine Learning, 2018, 80, 4095-4104.
33. Deep I, Related S, Deep I. Learning hierarchical features for scene labeling. IEEE Transactions on Pattern Analysis & Machine Intelligence 2013, 35, 1915-1929.
34. Lin G, Shen C, Hengel A. Efficient piecewise training of deep structured models for semantic segmentation. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016.
35. Liu S, Qi X, Shi J, Zhang H, Jia J. Multi-scale Patch Aggregation (MPA) for Simultaneous Detection and Segmentation. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016.
36. Pinheiro P, Collobert R. Recurrent convolutional neural networks for scene parsing. 2013.
37. Riharan B, Arbelaez P, Girshick R, et al. Hypercolumns for object segmentation and fine-grained localization. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015.
38. Mostajabi M, Yadollahpour P, Shakhnarovich G. Feedforward semantic segmentation with zoom-out features. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015.
39. Bing S, Zhen Z, Wang B, Gang W. Scene Segmentation with DAG-Recurrent Neural Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence 2018, 40, 1480-1493.
40. Chao P, Zhang X, Gang Y, Luo G, Jian S. Large Kernel Matters — Improve Semantic Segmentation by Global Convolutional Network. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
41. Fergus R, Taylor GW, Zeiler M. Adaptive deconvolutional networks for mid and highlevel feature learning. International Conference on Computer Vision, Barcelona, Spain, 2011.
42. Noh H, Hong S, Han B. Learning Deconvolution Network for Semantic Segmentation. 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 2016.
43. Ghiasi G, Fowlkes CC. Laplacian Reconstruction and Refinement for Semantic Segmentation. Springer International Publishing 2016, 4, 519-534.
44. Krizhevsky A, Sutskever I, Hinton G. ImageNet Classification with Deep Convolutional Neural Networks. Advances in neural information processing systems 2012, 25, 1097-1105.

45. Veit A, Wilber M, Belongie S. Residual Networks Behave Like Ensembles of Relatively Shallow Networks. Advances in Neural Information Processing Systems 2016, 29, 550-558.
46. Liu WC, Shu YZ, Tang SM, Liu JM. Remote Sensing Image Segmentation Using Dual Attention Mechanism Deeplabv3+ Algorithm. Tropical Geography, 2020, 40(2):303-313.
47. Nigam I, Huang C, Ramanan D. Ensemble Knowledge Transfer for Semantic Segmentation. IEEE Winter Conference on Applications of Computer Vision, Lake Tahoe, NV, USA, 2018.
48. Liu W, Rabinovich A, Berg AC. Looking Wider to See Better. computer science, 2015, 1506-1512.
49. Hao S, Zhou Y, Guo Y. A Brief Survey on Semantic Segmentation with Deep Learning. Neurocomputing 2020, 406, 302-321.
50. Vu TH, Jain H, Bucher M, Cord M, Perez P. Depth-aware Domain Adaptation in Semantic Segmentation. 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), 2019.
51. Wang L, Li D, Zhu Y, Tian L, Shan Y. Dual Super-Resolution Learning for Semantic Segmentation. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, USA, 2020.
52. Choi S, Kim JT, Choo J. Cars Can't Fly Up in the Sky: Improving Urban-Scene Segmentation via Height-Driven Attention Networks. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 2020.
53. Enze X, Wenhai W, Zhiding Y. Simple and Efficient Design for Semantic Segmentation with Transformers. Computer Vision and Pattern Recognition 2021.
54. Zheng S, Lu J, Zhao H, Zhu X, Zhang L. Rethinking semantic gmentation from a sequence-to-sequence perspective with transformers. IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021.