Flsnet: Fast and Light Segmentation Network

Qingwei Sun$^1$$^2$, Wei Chen$^2$, Jiangang Chao$^{2,*}$ and Hongbo Zhang$^1$

$^1$College of Aerospace Science and Engineering, National University of Defense Technology, Changsha 410005, China
$^2$China Astronaut Research and Training Center, Beijing 100094, China

$^*E$-mail: xjtucjg@163.com

Abstract. The use of deep learning for image segmentation has proven to be an efficient and accurate method, but with the complexity of the network structure, it takes up a lot of computing resources. The consumption of computing resources may be unacceptable during tasks. Aiming at this problem, a fast and light segmentation network (FLSNet) is proposed, which uses the Encoder-Decoder method to extract features. All convolutional layers use depthwise separable convolutions and the channel attention module is linked between Encoder and Decoder. Experiments are performed on the autonomous driving dataset CamVid. The results show that with a slight increase in segmentation accuracy, the model size becomes 8.65% of SegNet, the required computing resources are reduced by a dozen times, and the segmentation speed is increased by about 12%, which show that our network is efficient.

1. Introduction

At present, semantic segmentation technology has been applied to all aspects such as autonomous driving, indoor robots, and augmented reality. With the development of Fully Convolutional Network (FCN) [1], image segmentation technology based on deep learning has shown better and better performance, and on this basis, many excellent works have been proposed[2-5]. Great improvements have been made to the accuracy and speed of image semantic segmentation.

One thing that cannot be ignored is that with the deepening of the networks and the more complex structure design, the computing resources required by networks are also increasing. Simply performing the semantic segmentation task may not cause much impact, but if it is combined with other tasks, such as 3D reconstruction, it will not be completed smoothly because of high requirements on computer performance. At the same time, the existing network has a large amount of parameters and is difficult to deploy on mobile devices, which limits the application of semantic segmentation technology to a certain extent.

In response to the above problems, we proposed FLSNet, which uses depthwise separable convolutions to reduce the model's parameter amount. BachNorm [6] layers are used to speed up the network training speed, and the position indices of the max-pooling is saved to improve the accuracy of upsampling. At the same time, we also verified the impact of adding channel attention modules at the end of the Encoder. It has been proved by experiments that FLSNet is an efficient network, which can reduce the model size and increase the segmentation speed while ensuring accuracy.

The chapters of this article are arranged as follows. Chapter 2 explains the architecture of the model in detail. Chapter 3 shows the experimental data and procedures. Chapter 4 analyzes and compares the experimental results. Chapter 5 summarizes the full text.
2. Architecture

![Architecture Diagram]

**Figure 1.** Overview of the FLSNet architecture. The network is constructed using an encoder-decoder approach with depthwise separable convolutions. There is an attention channel module at the end of the encoder with the residual structure to connect to the decoder.

### 2.1. Overall Architecture

Similar to SegNet [2], we use the Encoder-Decoder mode, where Encoder is used to extract image features, and Decoder is used to restore the image structure. There is no fully connected layer between Encoder and Decoder. The entire network consists of convolutional layers. In order to reduce the model parameters, all convolutional layers are replaced with depthwise separable convolutions. The depthwise separable convolutions are composed of depthwise convolutions and pointwise convolutions. The activation function RuLU [7] is used after each convolutional layer. In order to speed up the training process, BachNorm [6] layer is used for data normalization after each convolution, whose implementation in caffe[8] is BachNorm layer and Scale layer. The specific architecture is shown in figure 1.

### 2.2. Encoder

In the Encoder phase, a backbone network similar to MobileNet[9] is used, except that the convolutional layer with stride 2 is not used in the downsampling, but the max-pooling is used. This has the advantage that the indices of the pooling can be extracted, which will improve the accuracy of upsampling. We use a total of four max-pooling, with a $2 \times 2$ window and stride 2. The size of the feature image becomes one-half of the original after every max-pooling. After the fourth pooling, five depthwise separable convolutions are used to fully extract the high-dimensional features of the image.

### 2.3. Decoder

In the Decoder stage, a backbone network similar to SegNet [2] is used, whose shape is opposite to VGG16[10]. Corresponding to Encoder, Decoder uses four upsampling and uses indices of the max-pooling. After each upsampling, the feature map becomes twice the original. After the last layer of convolution, Softmax is used to calculate the class probability of each pixel.

### 2.4. Attention channel module

We also explored the use of the Squeeze-and-Excitation (SE) [11] module in the semantic segmentation network, which we call attention channel module. Unlike the reference[5], we only use this module in the last layer of the Encoder. The reason to use only one channel attention module is to reduce the complexity of the network model, and we can intuitively understand that the relationship between the channels of the highest dimensional feature may be the most useful to us. The global average pooling is
used to extract the comprehensive features of each feature channel, and the result of the last convolution is changed from $14 \times 14 \times 512$ to $1 \times 1 \times 512$.

Calculated as follows:

$$a_c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} u_c(i, j)$$  \hspace{1cm} (1)

Where $u_c(i, j)$ is the pixel value of the original feature map and $a_c$ is the pixel value after global average pooling.

A $1 \times 1 \times 512 \times 32$ convolutional layer is used to compress the number of channels to 32, and then a $1 \times 1 \times 32 \times 512$ convolutional layer is used to restore the number of channels. The weight $w$ of each channel is calculated by Sigmoid and multiplied by each pixel of the result of Encoder. At the same time, we use the residual network[12] to add the result of the channel attention module to the result of the Encoder. The overall calculation formula is:

$$u'(i, j) = u(i, j) + w \cdot u(i, j)$$  \hspace{1cm} (2)

Where $u'(i, j)$ is the pixel value of the final feature image.

3. Experimental

3.1. Experimental data

The dataset used in our experiment is CamVid[13], which is a dataset for autonomous driving with 367 training images and 233 testing images. CamVid covers 11 types of road targets in day and night scenes, including roads, cars, buildings, trees and other structures. The data images are three channels, the label images are single channel, and the original resolution is $360 \times 480$. To speed up training, we resize the resolution of these images to $224 \times 224$.

3.2. Training method

3.2.1 SegNet. In order to verify the performance of our network, we use the SegNet network model without Bayesian as a reference benchmark. SegNet is a well-known semantic segmentation network that balances the accuracy of segmentation and the consumption of computing resources with good results. We did not get the SegNet pre-trained weights, and we re-trained SegNet for comparison fairness. The specific method is to fine-tune based on the pre-trained weights of VGG16. The optimization method is Stochastic Gradient Descent (SGD)[14]. We use the step learning strategy with initial learning rate 0.01. After each iteration of 4000 the learning rate becomes one tenth of the previous one. The momentum is set to 0.9, and the bach size is set to 8. We iterated 10,000 times on an NVIDIA GTX1060 GPU and selected the one with the highest accuracy. The loss curve is shown in figure 2:
3.2.2 **FLSNet.** We trained two new networks, one is FLSNet and the other is FLSNet_basic with the channel attention module removed. This is done to compare the the channel attention module's contribution to semantic segmentation. Due to equipment limitations, we trained FLSNet_basic in two stages. We first iterated it 80,000 times on a Nvidia GTX1050 GPU with a bach size of 1 and then fine-turn it on an Nvidia GTX1060 GPU with a bach size of 8. Unlike SegNet, we do not use any pre-trained data, but rather complete end-to-end training. As can be seen from figure 3, our network training converges quickly. The pre-trained data of FLSNet_basic is used to fine-tune FLSNet to obtain the best result after convergence.

4. **Results and analysis**

![Figure 3. The loss curve of FLSNet_basic. The left (a) is trained with bach size 1. The right (b) is fine-turned with bach size 8.](image)

*Test samples*
Figure 4. Test results on the autonomous driving dataset CamVid. It can be seen that FLSNet is the network with the best segmentation accuracy. Meanwhile, FLSNet and FLSNet_basic have great advantages in segmentation time and parameter amount. For details, refer to table 1. FLSNet has improved accuracy over FLSNet_basic because of the addition of the attention channel module.

The effect of different networks is shown in figure 4. All three networks can segment the scene more accurately. In order to compare the segmentation performance of different networks, we use a variety of indicators to test it, as follows:

Pixel Accuracy (PA), which indicates the percentage of correct pixels in the total pixels that are marked:

$$PA = \frac{\sum_{i=0}^{k} P_{ii}}{\sum_{i=0}^{k} \sum_{j=0}^{k} P_{ij}}$$  (3)

Mean Pixel Accuracy (MPA), which indicates the proportion of pixels correctly classified in each class and takes the average:

$$MPA = \frac{1}{k+1} \sum_{i=0}^{k} \frac{P_{ii}}{\sum_{j=0}^{k} P_{ij}}$$  (4)

Mean Intersection over Union (MIoU):

$$MIoU = \frac{1}{K+1} \sum_{i=0}^{k} \frac{P_{ii}}{\sum_{j=0}^{k} P_{ij} + \sum_{j=0}^{k} P_{ji} - P_{ii}}$$  (5)

Frequency Weighted Intersection over Union (FWIoU):

$$FWIoU = \frac{1}{\sum_{i=0}^{k} \sum_{j=0}^{k} P_{ij} \sum_{i=0}^{k} P_{ij} + \sum_{j=0}^{k} P_{ji} - P_{ii}}$$  (6)
In addition, there are model parameters (Parameters), floating point operations (FLOPS), the size of weight (Weights), and actual split time (Time) calculated with a Nvidia GTX1050 GPU, where parameters and FLOPS measure the computing power required by the model. The comparison of each parameter is shown in Table 1.

|          | SegNet | FLSNet_basic | FLSNet |
|----------|--------|--------------|--------|
| PA       | 0.73401 | 0.74241      | 0.75143 |
| MPA      | 0.56305 | 0.55285      | 0.55497 |
| MIoU     | 0.42685 | 0.42373      | 0.42871 |
| FWIoU    | 0.62810 | 0.65001      | 0.66440 |
| Weights  | 117.8M  | 10.2M        | 10.3M  |
| Time     | 96ms   | 84ms         | 86ms   |
| Parameters | 29433472 | 2519680     | 2552448 |
| FLOPS    | 30924472320 | 1164233728 | 1164266496 |

Due to our short training time, the visualization effect presented by all networks has not yet reached the optimal level. Compared with the official SegNet, the accuracy of our retrained SegNet has decreased. This may be because we have resized the image, the amount of parameters that the network needs to calculate is reduced, and the training is not enough.

Compared with SegNet, FLSNet_basic and FLSNet have a slight increase in accuracy. Parameters, FLOPS, Weights of FLSNet_basic and FLSNet have been reduced by an order of magnitude, which means that the requirements for computer performance have been reduced by an order of magnitude, which is very meaningful for deployment to mobile terminals or drones. And it is also meaningful to combine this network with task like 3D reconstruction. At the same time, the segmentation time was reduced by about 12%, which is also a great improvement.

The accuracy of FLSNet is improved compared to FLSNet basic, and the Weights and Parameters are slightly increased. This is caused by the addition of a channel attention module, and it also proves that the module is helpful for semantic segmentation tasks. At the same time, it can be predicted that if the channel attention module is loaded into each convolutional layer, the accuracy may be greatly improved, but at the same time the speed is reduced.

5. Conclusion

In this paper, a fast and light segmentation network FLSNet is proposed to solve the problem of excessive consumption of computing resources by semantic segmentation, and the performance of the network is verified on the autonomous driving data set. We mainly compare with the SegNet, and the results show that FLSNet can greatly reduce the model parameter amount with a slight increase of the accuracy, and it also greatly improves the actual segmentation speed. At the same time, we verified that adding the channel attention module at the end of the Encoder can improve the segmentation accuracy. FLSNet is an efficient network, which can be easily deployed on mobile devices or complete some complex tasks.

Acknowledgments

We are very grateful for the technical support for the fourth pre-research project of manned spaceflight (No. 060601) for the publication of this paper. At the same time, thanks to the National University of Defense Technology for their valuable comments on the thesis.

References

[1] LONG J, SHELHAMER E, DARRELL T, et al. Fully Convolutional Networks for Semantic Segmentation [J]. 2014, 39(4): 640-51.
[2] BADRINARAYANAN V, KENDALL A, CIPOLLA R, et al. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation [J]. 2019.
[3] CHEN L C, PAPANDREOU G, KOKKINOS I, et al. DeepLab: Semantic Image Segmentation with
Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs [J]. 2016, PP(99): 834-48.

[4] RONNEBERGER O, FISCHER P, BROX T. U-Net: Convolutional Networks for Biomedical Image Segmentation; proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, F, 2015 [C].

[5] ZHAO H, SHI J, QI X, et al. Pyramid Scene Parsing Network [J].

[6] IOFFE S, SZEGEDY C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift [M]. 2015.

[7] NAIR V, HINTON G E. Rectified Linear Units Improve Restricted Boltzmann Machines Vinod Nair; proceedings of the Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel, F, 2010 [C].

[8] JIA Y, SHELHAMER E, DONAHUE J, et al. Caffe: Convolutional Architecture for Fast Feature Embedding [J].

[9] HOWARD A G, ZHU M, CHEN B, et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications [J].

[10] NOH H, HONG S, HAN B. Learning Deconvolution Network for Semantic Segmentation [M]. 2015.

[11] HU J, SHEN L, ALBANIE S, et al. Squeeze-and-Excitation Networks [J].

[12] HE K, ZHANG X, REN S, et al. Deep Residual Learning for Image Recognition; proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), F, 2016 [C].

[13] BROSTOW G J, FAUQUEUR J, LETTERS R. Semantic object classes in video: A high-definition ground truth database [J]. 30(2): 88-97.

[14] BOTTOU L. Large-Scale Machine Learning with Stochastic Gradient Descent, Heidelberg, F, 2010 [C]. Physica-Verlag HD.