Real-time Segmenting Human Portrait at Anywhere

Ruifeng Yuan, Yuhao Cheng[♯], Yiqiang Yan, Haiyan Liu
Lenovo Research
Buidling1, No.10 Courtyard Xibeiwang East Road, Beijing, China
yuanrf1@lenovo.com, chengyh5@lenovo.com, yanyq@lenovo.com, ieliuhaiyan@163.com

Abstract

Real-time portrait segmentation is an important task for a wide range of human-centered applications. With the increase of mobile devices, such as mobile phones and personal computers, more and more human-centered applications are transferred to running on these devices to provide users with a better experience. So, lightweight model designing becomes indispensable for building applications on these limited-resource platforms. In this work, we propose a real-time segmentation U-shape architecture with a Re-parameter Compress Residual module (RCR module) and a bypass branch that can further improve the segmentation efficiency. In order to speed up during the inference phase, the RCR module is compressed during inference, and the bypass branch adds the missing edge information improving the learning skill of the network. Based on the experiments on the EG1800 and P3M-10K dataset compared with the state-of-the-art methods, the proposed method achieves better performance with less number of parameters. Specifically, our method reduces the number of parameters around 50% *while maintaining comparable high accuracy, and the details will be described in the experiment part.*

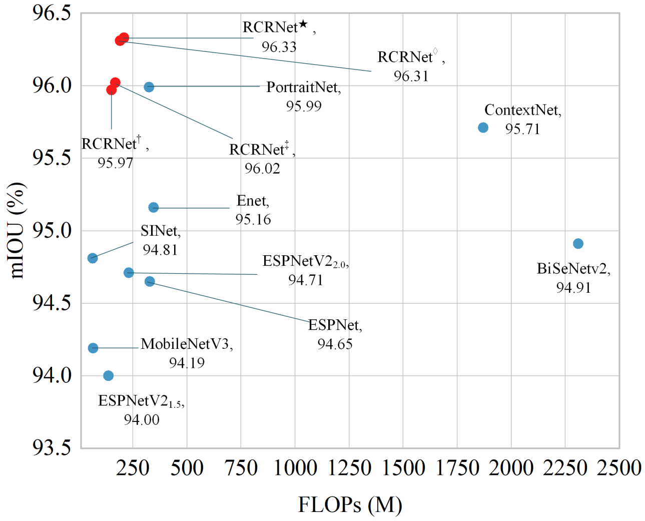

Figure 1. Speed-Accuracy Comparison. This figure shows the comparison between our methods(showing as the red dot) and the other methods(showing as the blue dot)on the EG1800 validation set. The dots in the upper left corner mean these models have a better balance of performance and efficiency.

1. Introduction

With the development of computer vision and artificial intelligence, we can easily classify images, detect objects in scenes, get the outline of the objects, etc. Meanwhile, as one of the key objects in daily life, humans derivate lots of subtasks such as person re-identification, human detection, human pose estimation, and so on. This kind of human-centered computer vision is challenging due to the complicated human shape, the differences between humans, the complexity of human action, and other factors. So, human-centered computer vision tasks have attracted much attention from academia and industry. Not only that, as the trend of work from everywhere, human-centered technologies are used in many devices such as laptops, mobiles, and other resource-limited devices. As a result, real-time or lightweight human-centered methods have thrived these years, and human portrait segmentation, as one of the fundamental technologies, has been developed. Human portrait segmentation has a wide application, such as getting the outline of the person, replacing the background, and so on. Although the previous methods of human portrait segmentation have state-of-the-art performance, they have too many parameters and can not run in real-time on devices whose computation power isn't as strong as some cloud servers with multiple GPUs. For this reason, this paper proposes a lightweight human portrait method. And Figure 1

[♯] Corresponding author
[*] This work is done when Haiyan Liu is an intern at Lenovo Research.

shows a detailed comparison between our methods and the other SOTA methods. Obviously, our methods achieve better performance with a smaller number of parameters. So, in this paper, we have the following contributions:

- Propose a new Re-parameter Compress Residual module to enhance usage the different level of information;
- Propose a new network with fewer weights whose performance is comparable with the other models having a large number of parameters;
- Numerous experiments were conducted to verify the superiority of the proposed method over other methods.

2. Related Work

Segmentation. The segmentation is to get the pixel-level mask of the object in the images or videos. With the development of deep learning, researchers have come up with many advanced methods in images, such as [1, 2, 8, 9, 13, 14, 21]. [1] firstly uses the fully connected convolution neural network to get the mask of the objects on the image; [2] uses the dilated convolution network to increase the receptive field to get the better performance on the segmentation task; [17] proposes using the *U* shape network structure to mix the information from the different size of the feature; [13] proposes a model with a spatial squeeze module and information blocking decoder to achieve outstanding performance with fewer parameters. Moreover, as the demand for video analysis increases, researchers have proposed some methods in the video domain, such as [16, 20, 24]. In the visual conference scenario, video segmentation methods are undoubtedly one of the key technologies to provide a better user experience. Nevertheless, the lack of high-quality training data hinders the usage of video segmentation methods based on supervised training. So in our paper, we use a training process to train the proposed model by using image data, which alleviates the eager need for the labeled video data.

Efficient Network Designing. Most of the deep learning networks have a large number of parameters that hinges on the usage of the deep learning methods used in some computation power-sensitive devices such as laptops, edge-computing nodes, mobile phones, and so on. In order to make these devices can use deep learning methods, researchers have proposed a series of designing of the network with less need for computation cost. [3–7, 18, 23] are some representative methods in this area. [6, 7, 18] design a new series of the network containing the module named depth-wise and path-wise convolution kernel to reduce the parameters without decreasing the performance. While [23] follows this design and improves the depth-wise by replacing the convolution computation with the shuffle operation. Meanwhile, [3] uses the re-parameterization method to increase the efficiency of the model in the inference phase. [5] propose a new module to get more feature maps through cheap operations, and in this way, it gets a better balance of efficiency and performance.

3. Method

In this section, we will elaborate on our method. Firstly, we will introduce the entire network architecture used in our proposed portrait segmentation method. And then, we will introduce the architecture of the proposed Re-parameter Compress Residual (RCR) module, which is specifically designed for real-time segmentation and serves as the encoder module and the decoder module that makes up the entire network. Then, we describe the re-parameter adaptive position encoding used in the encoder. At last, we will introduce the loss function used during the training process.

3.1. Overview

The overview of the RCR segmentation network (RCRNet) is shown in Figure 2. RCRNet contains an encoder module and a decoder module. Every stage uses two RCR modules shown in Figure 4. The output of the same layer of the encoder and the output of the upper layer of the decoder are added as the input of the decoder. In *Stage1*, we introduce an additional branch that uses the RCR modules and a Feature Fusion Module (FFM) used in the previous work [21]. After the second RCR module's output of this branch, we add a **SegHead** to guide the branch to learn edge information and guide the learning of *Stage1* in the decoder. Because the U-shaped structure is easy to lose edge information in the process of convolution, the branch module can ensure the retention of spatial information. Moreover, the SegHead's architecture is shown in the right-upper corner in Figure 2. It only consists of one 3×3 convolution kernel and one 1×1 convolution kernel with corresponding batch normalization and non-linear activation functions, and we find that even if the head has a simple structure, it can help the entire network keep more edge information. The training loss consists of edge loss and segmentation loss. In the following, we will detail each part of the proposed method.

3.2. Re-parameter Compress Residual Module

The key component of our proposed network is the RCR(Re-parameter Compress Residual) module. Figure 3(a) illustrates the layout of the RCR module. The number of feature map channels is adjusted in the first block, using $block1$ to denote the operations of the first block. Then, there are three branches in the second block. One branch, denoted by $block2_{Conv3}$, contains a depth-wise separable 3×3 convolution and a batch normalization layer. The second branch, $block2_{Conv1}$, contains a depth-wise separable 3×3 convolution and a batch normalization layer. The last branch does not have any computation operation, which

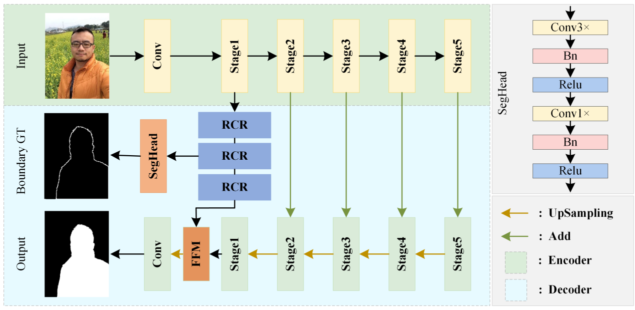

Figure 2. Overview of RCR segmentation network. The upper part is the encoder of the whole network that projects the RGB images into the high dimensions semantic feature. And the lower part is the decoding part which transfers the high-dimension information to the output. The whole architecture is designed as the shape of **U**, and the proposed module, RCR, is used in the connection between the encoder and decoder to enhance the usage of different level features without increasing a huge number of parameters. And the Feature Fusion Module(FFM) [21] in the decoder helps it leverage the high-level and low-level feature in a learning-based way.

only copies the first block's output to the next module. Because the contiguous feature maps often contain some similar information, the last branch passes this information directly to the final output. Therefore, the output map is calculated as follows:

$$\begin{aligned} x_{output} =& block2_{Conv3}(block1(x_{input}))+ \\ & block2_{Conv1}(block1(x_{input}))+ \\ & block1(x_{input}) \end{aligned} \tag{1}$$

where x_{input}, x_{output} respectively denote the RCR module input and output. In the condition of efficiency, we adopted add as the second block's two-branch fusion operation.

As the process of re-parameterization [3], the three branches in the second stage can merge into one, as shown in Figure 3(b). Specifically, the second block in the RCR module could convert into a single 3×3 convolution layer in the inference stage. For the identity branch, it can be viewed as 1×1 convolution layer with an identity matrix as the kernel, the 1×1 kernels added onto the central point of 3×3 kernel by zero-padding. The combined convolution layer and the batch normalization layer can merge into a convolution layer. The process of the convolution layer can be mathematically modeled as follows:

$$Conv(x) = W * x + b \tag{2}$$

The process of batch normalization layer can be mathematically modeled as follows:

$$BN(x) = \gamma * \frac{x - \mu}{\sigma} + \beta \tag{3}$$

Then, we can convert the convolution layer and its preceding Batch Normalization layer into a convolution layer with a bias vector. The process can finally be calculated as follows:

$$\begin{aligned} BN(Conv(x)) &= \gamma * \frac{W * x + b - \mu}{\sigma} + \beta \\ &= \frac{\gamma * W}{\sigma} * x + \frac{\gamma * (b - \mu) + \sigma * \beta}{\sigma} \end{aligned} \tag{4}$$

At inference, the convolution and batch-norm are linear operations. They can be easily folded into a single convolution layer with weights $\frac{\gamma * W}{\sigma}$ and bias $\frac{\gamma*(b-\mu)+\sigma*\beta}{\sigma}$. Batch-norm is folded into the preceding convolution layer in the two branches. After obtaining the batch-norm folded weights in each branch, the weights $W = W_{branch1} + W_{branch2}$ and bias $b = b_{branch1} + b_{branch2}$ for convolution layer at inference is obtained.

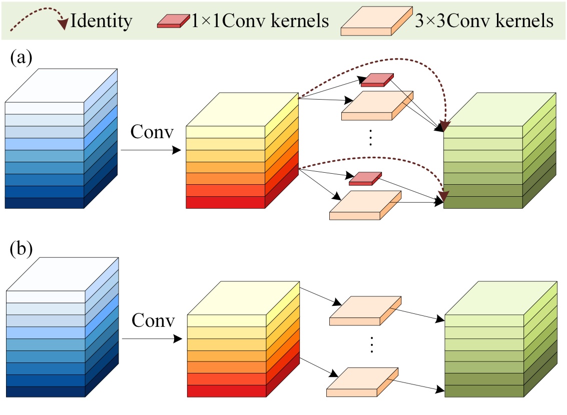

Figure 3. The architecture of the RCR(Re-parameter Compress Residual) module. (a) the module in the training phase. (b) the module in the inference phase, the identity and convolution branch in the second block can be merged into one convolution layer.

Based on the RCR module, we designed the network's every stage. As shown in Figure 4, there are two structures that mainly consist of two stacked RCR modules. The first RCR module acts as an expansion layer or compression layer, changing the number of channels. The second RCR module further extracts segmentation features. Then the shortcut is connected between the inputs and the outputs of these two RCR modules. The network's stages described above are A-structure. B-structure is for the case where the feature map needs downsampling or upsampling, the shortcut path is implemented by a downsampling layer, and a depthwise convolution with stride=2 is inserted between the two RCR modules. In practice, the primary convolution in the RCR module here is point-wise convolution for its efficiency.

3.3. Loss function

We use the similar loss function described in the [22], and the format is shown as the following. The L_m means

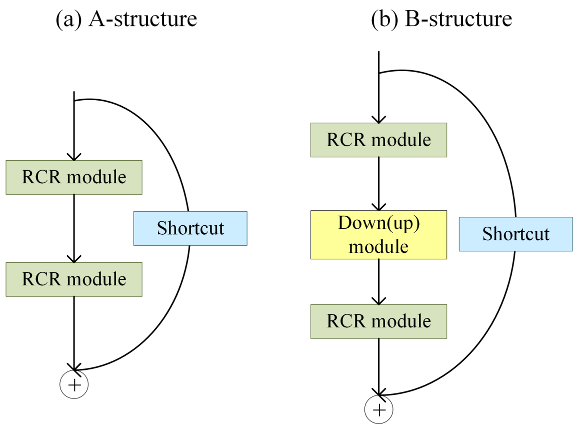

Figure 4. Different structures of the RCR module. (a) shows the A-type structure without the down or up sample module, which saves the number of parameters. While (b)shows the B-type structure having the extra module to improve the performance of the whole module.

the cross entropy loss between the ground truth and the predicted result. L_c is the constraint loss and L_e is the boundary loss, which refines the model. In the training process, we set the $\alpha = 1$ and $\beta = 0.3$.

$$L = L_m + \alpha \times L_c + \beta \times L_e \tag{5}$$

4. Experiments

We implement our method on three datasets: EG1800 [19] and P3M-10K [10] to evaluate the effectiveness of the RCR module and re-parameter adaptive position encoding, respectively. We first introduce the datasets, and implementation details. Then, we report our accuracy and speed results on different networks compared with other algorithms. Finally, we discuss the impact of components in our proposed approach.

4.1. Datasets

EG1800. We use the dataset images declared in [13]. The original dataset contains 1800 portrait images, mainly self-portraits with a mobile phone. Since several image URL links are invalid, the existing dataset that can be collected contain 1579 images. We use 1309 images as the training set and 270 images as the validating/testing dataset.

P3M-10k. P3M-10k includes 10421 face privacy-protected pictures and corresponding fine matting annotations. Among them, the training set has 9421 high-definition portrait pictures with blocked faces. The test set is divided into two parts: (1) *P3M-500-p* provides 500 faces whose privacy information is blocked and high-precision annotations to verify the matting effect of the model under privacy protection; (2) *P3M-500-np* provides 500 celebrity portrait images, whose face information can be disclosed to verify the generalization ability of the model on ordinary complete portraits.

4.2. Implementation Details

4.2.1 Data Augmentation

We followed the data augmentation method in [22]. We use deformation augmentation and texture augmentation to supplement the original training dataset, leading to better segmentation results. The deformation augmentation contains random horizontal flip, random rotation, random resize, and random translation. The texture augmentation contains random noise, image blur, random color change, random brightness change, random contrast change, and random sharpness change. The images are resized to 224 × 224.

These data augmentation methods can be divided into two categories: one is deformation augmentation and the other is texture augmentation. Deformation augmentation augments the position or size of the target but will not affect the texture. On the other hand, texture augmentation complements the texture information of the target while keeping the position and size.

The deformation augmentation methods used in our experiments include the following:

- Random Flip. For this augmentation, we choose to use the random horizontal flip because the portraits are often not eudipleural, and using horizontal flip could enhance the generalization of models. Meanwhile, for the portraits, the horizontal flip is more meaningful than the vertical flip.
- Random Rotation. We use the rotation in the range of $[-45°, 45°]$. The reason for using it is to simulate different poses of people.
- Random Resizing. We use the resizing factors in the $[0.5, 1.5]$ range to enlarge the number of images in different resolutions.
- Random Affine. We use the translation factors in $[-0.25, 0.25]$ in the random affine function.

The texture augmentation methods used in our experiments include the following:

- Random Noise. We randomly add the Gaussian noise in the images to enhance the model's generalization in different image qualities. The Gaussian noise has the parameter $\sigma = 10$.
- Image Blur. We use a blur kernel with a kernel size of 3 or 5 to randomly blur the training images.
- Random Color Change. We use this augmentation to make one image have different color representations with the same semantic meaning. The change factors are in $[0.4, 1.7]$.
- Random Brightness Change. This augmentation is to imitate the different lighting conditions. The change

Table 1. Segmentation accuracy comparison of different models on EG1800 datasets.

Method	FLOPs(M)	Params(M)	mIOU(%)
ENet [14]	346	0.355	95.16
BiSeNetV3 [4]	44829	14.24	94.91
PortraitNet [22]	325	2.080	95.99
ESPNet [11]	345	0.328	94.65
ESPNetV2$_{2.0}$ [12]	231	0.778	94.71
ESPNetV2$_{1.5}$ [12]	137	0.458	94.00
ContextNet [15]	1870	0.838	95.71
MobileNetV3 [6]	66	0.458	94.19
SINet [13]	**64**	**0.087**	94.81
RCRNet*	217	0.772	**96.33**

factors are in $[0.4, 1.7]$.

- Random Contrast Change. It can modify the contrast of the images. The change factors are in $[0.6, 1.5]$.
- Random Sharpness Change. The change in the sharpness of the images will cause the objects to have clear outlines or fuzzy outlines, which makes the model handle different situations. The change factors are in $[0.8 - 1.3]$.

Every operation in deformation augmentation and texture augmentation added up with a probability of 0.5 during training. After data augmentation, we normalize the input images.

4.2.2 Training Method

As we have mentioned before, one of the difficulties in using video segmentation in practice is the need for labeled video data. We use another way to simulate the video data by using the image data, and in this way, the trained model could have a better ability to deal with the video data. In detail, we expand the channels of each image from 3 to 4. The extra channel is the mask of the last frame, so we want to make the model get the temporal information by using the last frame's mask. However, in the image dataset, we can not get the last frame, or in other words, the prior frame does not exist, so in this situation, we will generate some fake prior masks to simulate the complex motion between frames. Specifically, we apply some affine transformation on the current image's ground truth mask during training. And then, the generated mask and the original image will be fed into the model together to generate the segmentation result of the current frame. In this way, we can use the image to train the model, which is suitable for the video data.

4.2.3 Experimental Setup

We conduct our experiments by using the PyTorch framework. We perform all experiments on NVIDIA RTX 6000 GPU with batch size 32. We use the Adam algorithm with momentum 0.9 and weight decay 5e-4. The initial learning rate is set as 0.01 and is multiplied by $0.95^{\frac{epoch}{20}}$ to adjust with 2000 epochs.

Table 2. Segmentation accuracy comparison of different models on P3M-10k datasets.

Method	FLOPs(M)	mIOU(%) P3M-500-p	mIOU(%) P3M-500-np
ENet [14]	346	95.64	95.91
PortraitNet [22]	325	95.51	95.75
BiSeNetv3 [4]	44829	94.25	94.31
RCRNet‡	177	95.31	95.61
RCRNet$^{\diamondsuit}$	190	95.45	95.66
RCRNet*	217	**95.76**	**96.15**

4.3. Evaluation Results

To demonstrate the effectiveness of the proposed model, we compared it with the existing popular models in the portrait segmentation field. Table 1 shows the quantitative comparison among the above different segmentation models. We can see that our method enables us to obtain higher segmentation accuracy. Compared with PortraitNet, which has state-of-the-art accuracy, our RCRNet* has achieved higher accuracy and reduced half of the FLOPs. Specifically, compared with the method with the second highest performance, our proposed method just use $0.772M$ parameters, which reduces more than 50% of $2.080M$.

Meanwhile, we also do experiments on the other human portrait segmentation dataset, P3M-10K. The experiment results are shown in Table 2. Based on the experiment results, we can know that no matter whether the images with privacy protection or not, our proposed method outperforms the other methods with less computation cost. Moreover, compared with the previous methods, our proposed method uses fewer parameters to achieve a good performance.

4.4. Ablation Study

The ablation study is shown in Table 3. RCRNet† uses the RCR module in the encoder while using the same decoder in [22]. Based on Table 3, we can know that the improvement of the encoder makes the new network have fewer FLOPs and the number of parameters, while its performance is comparable with the benchmark. That proves the efficiency of the proposed module in the encoder part. Moreover, because the architecture of the network can be divided into the encoder and decoder parts, we want to know whether the RCR module is useful in the decoder part.

Obviously, from Table 3, RCRNet‡ noting the network whose encoder and decoder both use the proposed RCR module, outperforms the RCRNet† with a slight increase

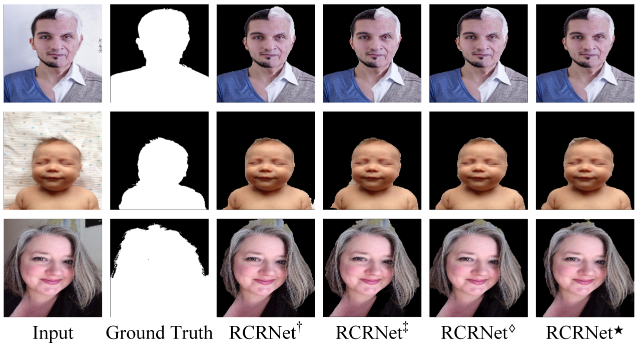

Figure 5. Segmentation results on the EG1800 validation dataset.

of the number of parameters. The experiment results mean our proposed RCR module can be used in both the encoder and decoder parts.

RCRNet$^\diamondsuit$ and RCRNet* show the result of the proposed method with the spatial path. From Table 3, we can know that the spatial path has the ability to enhance the performance of the portrait segmentation network, so the proposed network, RCRNet*, contains the spatial path.

Figure 5 shows several difficult portrait segmentation results generated by the four methods. We can see that the RCRNet† has slight flows in the details of the hair and some edges of the body. Furthermore, RCRNet‡ performed slightly better on these defects, but the naked eye was still able to distinguish the errors. The results of RCRNet$^\diamondsuit$ and RCRNet* have almost no incorrect segmentation, which is consistent with the ground truth. Compared with the other methods, the RCRNet* has higher segmentation accuracy and performance.

Table 3. Segmentation accuracy comparison of different models on EG1800 datasets. We test all the modifications of the RCRNet and get the result. RCRNet† improves the encoder part, RCRNet‡ improves the decoder part, the RCRNet$^\diamondsuit$ improves the encoder and the bypass, and RCRNet* changes the encoder, decoder and the bypass at the same time.

Method	FLOPs(M)	Params(M)	mIOU(%)
PortraitNet [22]	325	2.080	95.99
RCRNet†	157	0.607	95.97
RCRNet‡	177	0.608	96.02
RCRNet$^\diamondsuit$	190	0.610	96.31
RCRNet*	217	0.770	96.33

5. Conclusion

In this paper, we propose a new model named RCRNet, which uses the Re-parameter Compress Residual modules as the vital component. Based on the experimental results, our proposed method performs better with lower computation costs. The proposed lightweight portrait segmentation network leverages the RCR module. The module has multiple branches in training, which has stronger learning ability, and multiple branches can be combined into one branch in inference without accuracy loss. We also add a bypass branch between the low layers and the deep layers that improves segmentation accuracy. The experimental results demonstrate that our RCRNet has fewer parameters and higher segmentation efficiency, which can be applied in real-time portrait segmentation on mobile devices. In the future, we will continue researching this area and hope to solve some bad cases, such as the blurry edge and so on.

References

[1] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014. 2

[2] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 2

[3] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13733–13742, 2021. 2, 3

[4] Mingyuan Fan, Shenqi Lai, Junshi Huang, Xiaoming Wei, Zhenhua Chai, Junfeng Luo, and Xiaolin Wei. Rethinking bisenet for real-time semantic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9716–9725, 2021. 2, 5

[5] Kai Han, Yunhe Wang, Qi Tian, Jianyuan Guo, Chunjing Xu, and Chang Xu. Ghostnet: More features from cheap operations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1580–1589, 2020. 2

[6] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324, 2019. 2, 5

[7] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 2

[8] Dongjin Huang, Di Wu, Jinhua Liu, and Yushan Lv. Ddcnet: A lightweight network with variable receptive field for real-time portrait segmentation in complex environment. In *Advances in Computer Graphics: 39th Computer Graphics International Conference, CGI 2022, Virtual Event, September 12–16, 2022, Proceedings*, pages 465–476. Springer, 2023. 2

[9] Yong-Woon Kim, Yung-Cheol Byun, and Addapalli VN Krishna. Portrait segmentation using ensemble of heterogeneous deep-learning models. *Entropy*, 23(2):197, 2021. 2

[10] Sihan Ma, Jizhizi Li, Jing Zhang, He Zhang, and Dacheng Tao. Rethinking portrait matting with privacy preserving. *arXiv preprint arXiv:2203.16828*, 2022. 4

[11] Sachin Mehta, Mohammad Rastegari, Anat Caspi, Linda Shapiro, and Hannaneh Hajishirzi. Espnet: Efficient spatial pyramid of dilated convolutions for semantic segmentation. In *Proceedings of the european conference on computer vision (ECCV)*, pages 552–568, 2018. 5

[12] Sachin Mehta, Mohammad Rastegari, Linda Shapiro, and Hannaneh Hajishirzi. Espnetv2: A light-weight, power efficient, and general purpose convolutional neural network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9190–9200, 2019. 5

[13] Hyojin Park, Lars Sjosund, YoungJoon Yoo, Nicolas Monet, Jihwan Bang, and Nojun Kwak. Sinet: Extreme lightweight portrait segmentation networks with spatial squeeze module and information blocking decoder. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2066–2074, 2020. 2, 4, 5

[14] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint arXiv:1606.02147*, 2016. 2, 5

[15] Rudra PK Poudel, Ujwal Bonde, Stephan Liwicki, and Christopher Zach. Contextnet: Exploring context and detail for semantic segmentation in real-time. *arXiv preprint arXiv:1805.04554*, 2018. 5

[16] Andreas Robinson, Felix Jaremo Lawin, Martin Danelljan, Fahad Shahbaz Khan, and Michael Felsberg. Learning fast and robust target models for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7406–7415, 2020. 2

[17] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 2

[18] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 2

[19] Xiaoyong Shen, Aaron Hertzmann, Jiaya Jia, Sylvain Paris, Brian Price, Eli Shechtman, and Ian Sachs. Automatic portrait segmentation for image stylization. In *Computer Graphics Forum*, volume 35, pages 93–102. Wiley Online Library, 2016. 4

[20] Kai Xu, Longyin Wen, Guorong Li, Liefeng Bo, and Qingming Huang. Spatiotemporal cnn for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1379–1388, 2019. 2

[21] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 325–341, 2018. 2, 3

[22] Song-Hai Zhang, Xin Dong, Hui Li, Ruilong Li, and Yong-Liang Yang. Portraitnet: Real-time portrait segmentation network for mobile device. *Computers & Graphics*, 80:104–113, 2019. 3, 4, 5, 6

[23] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6848–6856, 2018. 2

[24] Bin Zhao, Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Generating masks from boxes by mining spatio-temporal consistencies in videos. *arXiv preprint arXiv:2101.02196*, 2021. 2