AMMNet: an Attention-based Multi-scale Matting Network

AMMNET: AN ATTENTION-BASED MULTI-SCALE MATTING NETWORK

BY

CHENXIAO NIU, B.Eng.

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

Master of Applied Science (2019) McMaster University

(Electrical & Computer Engineering) Hamilton, Ontario, Canada

TITLE:	AMMNet: an Attention-based Multi-scale Matting Network
AUTHOR:	Chenxiao Niu B.Eng., (Electric and Information Engineering) Beijing University of Posts and Telecommunications, Beijing, China
SUPERVISOR:	Dr. Jun Chen
NUMBER OF PAGES:	x, 49

To dear Miss.Tang

Abstract

Matting, which aims to separate the foreground object from the background of an image, is an important problem in computer vision. Most existing methods rely on auxiliary information such as trimaps or scibbles to alleviate the difficulty arising from the underdetermined nature of the matting problem. However, such methods tend to be sensitive to the quality of auxiliary information, and are unsuitable for real-time deployment. In this paper, we propose a novel Attention-based Multi-scale Matting Network (AMMNet), which can estimate the alpha matte from a given RGB image without resorting to any auxiliary information. The proposed AMMNet consists of three (sub-)networks: 1) a multi-scale neural network designed to provide the semantic information of the foreground object, 2) a Unet-like network for attention mask generation, and 3) a Convolutional Neural Network (CNN) customized to integrate high- and low-level features extracted by the first two (sub-)networks. The AMMNet is generic in nature and can be trained end-to-end in a straightforward manner. The experimental results indicate that the performance of AMMNet is competitive against the state-of-the-art matting methods, which either require additional side information or are tailored to images with a specific type of content (e.g., portrait).

Acknowledgements

First of all, I would like to take this opportunity to express my heartily and deepest gratitude to my supervisor, Prof. Jun Chen, for his patient and consistent supervision and strong support throughout my Master program. He not only provided me with a new perspective to view knowledge, but also cultivated my eager to seek for truth and science. His knowledgeable and kind advise will always be remembered and cherished. I would not be able to finish this work without his help.

Second, I want to give my thanks to Dr. Martin Brooks from ShapeVision Inc. for his support of this project. And the discussions and advise he gave me about this project are also helpful. It is a great honor for me to work with him.

Furthermore, I would like to thank Dr. Sorina Dumitrescu and Dr. Jiankang Zhang for being members of my defence committee. I feel grateful for their time for reviewing my paper and providing valuable feedback.

Last but not least, I am also lucky to have my parents and girlfriend Yanyan for their love and support. With their firm belief and determination, I gain the confidence and power to overcome all the obstacles that happened in the program.

To them, I dedicate this thesis.

Notation and abbreviations

Adam	Adaptive moment estimation algorithm
CNN	Convolutional Neural Network
GPU	Graphics Processing Unit
MSE	Mean Square Error
ReLu	Rectified Linear unit
FCN	Fully Convolutional Network
ResBlock	Residual Block
DenseBlock	Densely Block

Contents

List of Figures

Chapter 1

Introduction and Problem Statement

In many real-world applications, one needs to perform so-called matting, which is to accurately separate the foreground and the background of images or videos. For instance, the matting technology is widely used in the film industry and photography to improve professional work. It can also be leveraged to faciliate the creation of advertisements for e-commerce websites (e.g., generating customized fashion model designs).

The key equation underlying the matting problem is as follows:

$$\begin{aligned} I_i^{(c)} &= \alpha_i F_i^{(c)} + (1-\alpha_i) B_i^{(c)}, \\ c &\in \{R, G, B\}, \quad i = 1, \cdots, N. \end{aligned} \tag{1.1}$$

Here $I_i^{(c)}$, $F_i^{(c)}$, and $B_i^{(c)}$ are respectively the intensities of color channel c of pixel i of the RGB image, the foreground and the background; the matte value $\alpha_i \in [0,1]$ specifies how the foreground and the background are superimposed at pixel i; N is the total number of pixels in the image. Matting, which aims to recover the alpha matte $\alpha \triangleq (\alpha_i)_{i=1}^N$ based on the RGB image $\mathcal{I} \triangleq (\mathcal{I}^{(c)})_{c\in\{R,G,B\}}$ (with $\mathcal{I}^{(c)} \triangleq (I_i^{(c)})_{i=1}^N$), is an underdetermined problem

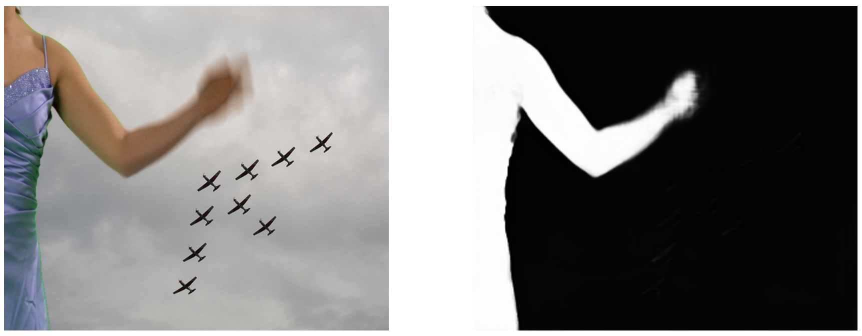
Levin *et al.* (2008) since there are fewer equations than unknowns (see Eq. (1.1)).

To deal with the underdetermined nature of the matting problem, most existing methods Aksoy *et al.* (2017a); Chen *et al.* (2013); Levin *et al.* (2008); Xu *et al.* (2017) rely on extra constraints offered by auxiliary information, such as trimaps or scribbles, acquired through user interactions. However, such methods suffer from two major issues. Firstly, they are not suitable for real-time deployment, and the required interactions could be difficult for nonprofessional users to provide. Secondly, their performance tend to be sensitive to the quality of auxiliary information; indeed, sometimes even well-created strokes or trimaps cannot guarantee good matting results.

In view of the shortcomings of interactive methods, there is a clear demand for fully automated image matting methods amenable to large-scale real-time implementation. One possible approach is to modify interactive methods by replacing manually-supplied auxiliary information with its algorithmically-generated substitute. However, the resulting methods are likely to inherit some of the shortcomings of interactive methods, say, the sensitivity issue. In fact, the quality of algorithmically-generated auxiliary information is even harder to control. For example, trimaps created by segmentation methods often suffer from the simple boundary erosion problem, which may jeopardize the final matting result; furthermore, classification errors introduced by segmentation tend to exacerbate the situation. Note that the aforementioned shortcomings are largely the consequence of the sequential architecture of such automated matting methods, where one essentially first identifies a coarse matte using high-level semantic information then produces a refined matte by exploiting low-level features. The sensitivity to the quality of the coarse matte is simply a manifestation of the bottleneck effect of this sequential architecture.

In this work, we take a fundamentally different approach to automated image matting.

Specifically, we construct the final matting result by leveraging high- and low-level features simultaneously. This is accomplished by adopting a novel parallel architecture, which, in contrast to its sequential counterpart, places different levels of features on an equal footing and consequently is immune from the detrimental bottleneck effect. In the proposed parallel architecture, the upper branch is a multi-scale network designed to learn semantic features while the lower branch supplements the upper branch by attending to high-frequency details via the attention mechanism; finally the extracted features are integrated to produce the matting result. The overall system is generic in nature, fully automated, and end-to-end trainable. The experimental results indicate that this system can effectively take advantage of different levels of features, and its performance is competitive against the state-of-the-art, which either require manually-supplied auxiliary information or are tailored to a special category of images. See Figure 1.1 for an alpha matting example.



(a) Input (b) Output

Figure 1.1: Direct alpha matting result. Left: Original RGB image. Right: Alpha matte.

The rest of this thesis is organized as follows: In Chapter 2, we review many alpha matting methods and some related techniques. Chapter 3 introduces the proposed Attention Based Mult-Scale Matting Network (AMMNet) together with a detailed explanation of the network architecture and its building blocks. In Chapter 4, we discusses details of the losses used in the AMMNet architecture, and shows some analysis of the designed loss. Chapter

5 illustrates detail implementation of our network, and also shows the experimental results of AMMNet and the performance comparison with other state-of-the-art methods. Finally, Chapter 6 makes the conclusion and discusses our future work of image matting problem.

Chapter 2

Related Work and Background

Over the past few decades, many matting methods have been developed. They roughly fall into two categories: traditional methods and deep-learning-based methods. Most traditional methods predict the mattes through samplingChuang *et al.* (2001); Gastal and Oliveira (2001); He *et al.* (2011); Shahrian *et al.* (2013); Wang and Cohen (2007) or propagatingAksoy *et al.* (2017a); Chen *et al.* (2013); Grady *et al.* (2005); Levin *et al.* (2008); Sun *et al.* (2004) on color or low-level features. Recently deep-learning-based matting methods Cho *et al.* (2016); Chen *et al.* (2018); Xu *et al.* (2017) have gained popularity due to their superior performance. Cho *et al*. Cho *et al.* (2016) propose a CNN for image matting, which makes use of the results of closed-form matting Levin *et al.* (2008) and KNN matting Chen *et al.* (2013) as supplementary information. The work by Xu *et al*. Xu *et al.* (2017), which is based on an encoder-decoder structure, represents the state-of-the-art of general image matting. However, all the above methods require auxiliary information, such as trimaps or scribbles, from user interactions. Such auxiliary information localizes the foreground object and provides semantic information. As a consequence, the remaining task is simply to resolve the residual uncertainties (say, predict the alpha values in the unknown region of the trimap), and low-level features typically suffice for this purpose.

Fully automated image matting has recently received much attention. Quan *et al*. Chen *et al.* (2018) develop a sub-network called T-Net to automatically generate trimaps, which are then fed into an encoder-decoder network Xu *et al.* (2017) for matte prediction. This solution realizes automated matting, but still needs trimaps to pre-train the T-Net. Shen *et al*. Shen *et al.* (2016a) use a CNN to generate trimap labels from a pre-defined mask shape and the RGB image, then produce the final matting result using a closed-form method. For all these methods, it is important to ensure the quality of algorithmically-generated trimap-like information, which is a challenging task in general. To alleviate this difficulty, essentially all existing automated matting methods are designed specifically for a particular category of images (mostly portrait) so that the associated semantic information can be leveraged to facilitate the generation of trimap-like information.

It is instructive to compare matting with segmentation. The former is a (pixel-level) regression problem while the latter is a classification problem. Segmentation mostly relies on high-level semantic features. Indeed, popular segmentation methods (e.g., multi-scale and dilated convolutions Yu and Koltun (2015), PSPNet Zhao *et al.* (2015)) all make use of Fully Convolutional Networks (FCNs) Long *et al.* (2015) with large receptive field to enable the learning of semantic patterns. In contrast, both high- and low-level features are needed for matting. Nevertheless, it is by no means obvious how such features should be strategically combined to produce the final matte since their exact roles are hard to delineate. Most existing matting methods exploit high- and low-level features in a sequential manner. It is conceivable that joint exploitation of these heterogeneous features might deliver better matting results. The main difficulty is to come up with an effective way to realize this somewhat abstract idea. A potential solution is offered by attention mechanisms, which have found applications to a wide range of tasks in natural language processing Chorowski *et al.* (2015); Bahdanau *et al.* (2014); Sankaran *et al.* (2016) and computer vision Zhang *et al.* (2018); Li *et al.* (2018); Liu *et al.* (2018). It will be seen that attention mechanisms enable the system to learn, via a data-driven approach, the appropriate contributions of different features to the final matting result without pre-defined ordering or priority.

2.0.1 Background

Sigmoid Sigmoid function is used as activation function in machine learning. It has a formulation as follows:

$$f(x) = \frac{1}{1+e^{-x}} \tag{2.1}$$

Sigmoid function maps the value of a function from range $[-\infty, \infty]$ to $[0, 1]$, which im-

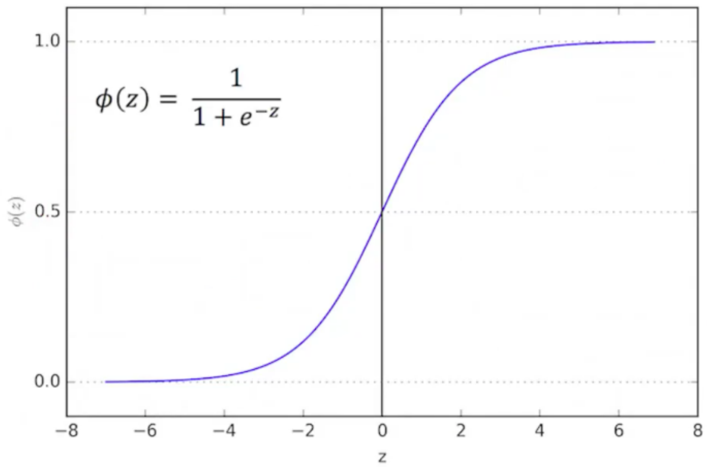


Figure 2.1: Sigmoid function.

ports nonlinearity to the neural network. Therefore, deep neural network can approximate arbitrary nonlinear function and can solve complicated problems. The differentiability of sigmoid function also ensures backpropagation of the training step. It is used in classification and image reconstruction tasks, where value range of $[0, 1]$ needed. The curve of sigmoid function can be seen from Fig. 2.1.

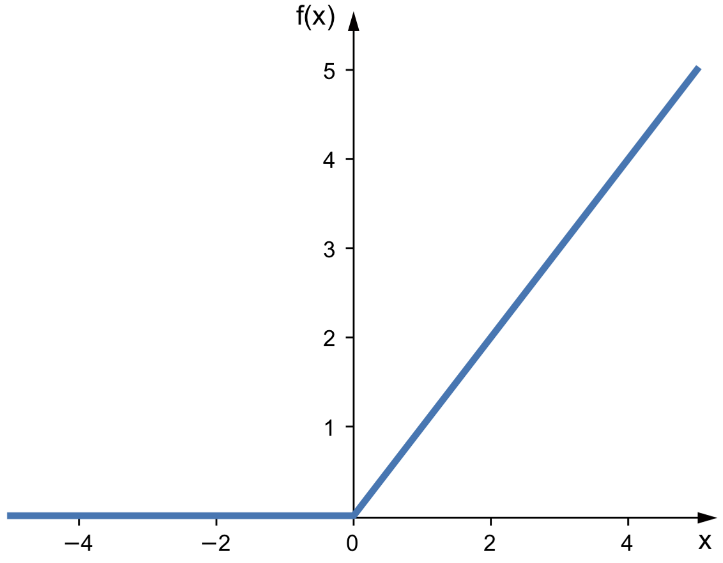


Figure 2.2: ReLU function.

ReLU The use of sigmoid function has some drawbacks such as gradient vanishing and computational inefficient. Rectified Linear Unit (ReLU) on the contrary is easy to compute and can avoid gradient vanishing. Another functionality o ReLU function is it increase the sparsity of deep neural network. A ReLU is simply defined as $f(x) = max(0, x)$, and the nonlinear function is represented in Fig. 2.2. As we can see from the formula, the function is zero for negative values, and it grows linearly for positive values.

Residual Block Residual Network (ResNet), created by He *et al.* (2016a) makes it possible to train a deep neural network without gradient vanishing or explosion. The main concept of ResNet is Residual Block (ResBlock) that can be seen from Fig. 2.3.

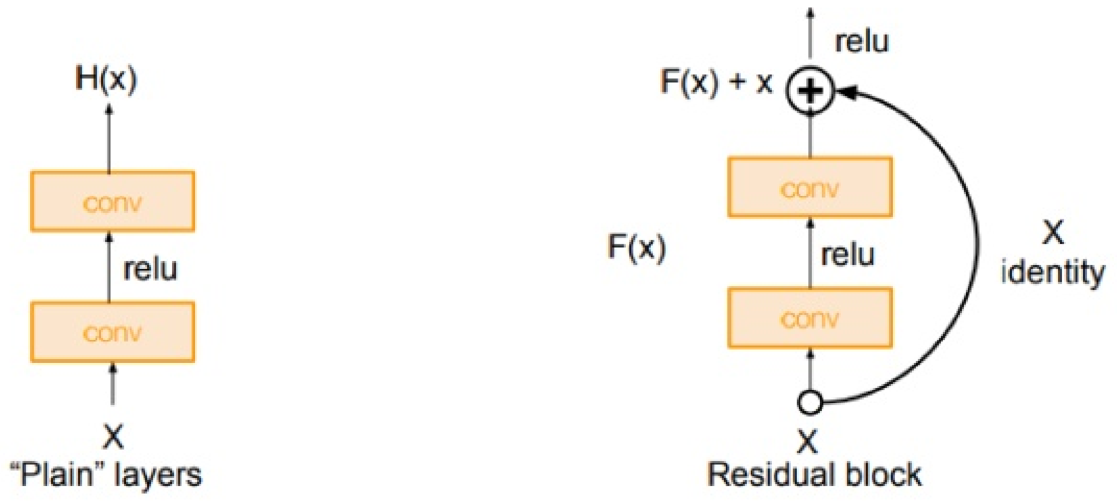


Figure 2.3: Structure of Resblock. The left figure is a conventional convolutional layers while the right figure is a standard resblock.

Instead of learning a specific function $\mathcal{H}(x)$, ResBlock learns residual $\mathcal{F}(x)$ from the data and combines x in a form of $\mathcal{H}(x) = \mathcal{F}(x) + x$. It can be accomplished by adding x as identity function in ResBlock. The intuition of ResBlock is that it is easy to solve $\mathcal{F}(x) = 0$ rather than $\mathcal{F}(x) = x$ using stack of non-linear convolutional layers. By using ResBlocks, one can make a deeper and powerful neural network and have quicker convergence.

DenseBlock DenseBlock Huang *et al.* (2017a) can be seen from Fig. 2.4, is block of densely connected convolutional layers compare to ResBlock. The problems arise when CNNs go deeper because of a long path for input data to the output. DenseBlock is used as the next step on the way to keep increasing the depth of deep convolutional netowks. It also

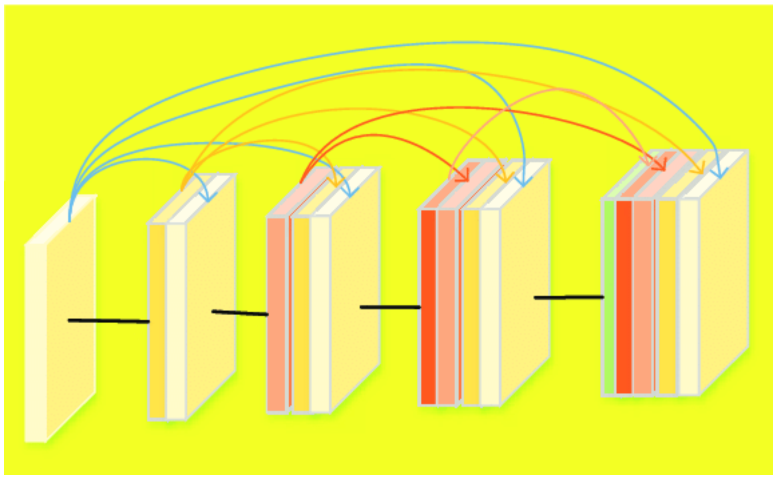


Figure 2.4: The structure of DenseBlock.

reduces the parameters of the neural network and increase sparsity in a large scale. Furthermore, some variations of ResBlocks have proven that many layers are barely contributing and can be dropped. In fact, the number of parameters of ResBlocks are big because every layer has its weights to learn. Instead, DenseBlocks layers are very narrow, and they just add a small set of new feature-maps. Another problem with very deep networks is the problems to train. DenseBlocks solve this issue since each layer has direct access to the gradients from the loss function and the original input image.

MaxPool Pooling is a method used in CNNs to down-sample data, and in computer vision area such method also enlarge receptive field which preserve high-level features of images. The pooling layer is usually placed after the Convolutional layer. The utility of pooling layer is to reduce the spatial dimension of the input volume for next layers. Note that it only affects weight and height but not depth. Fig. 2.5 shows how to operate max pool. The max pool layer is similar to convolution layer, but instead of doing convolution

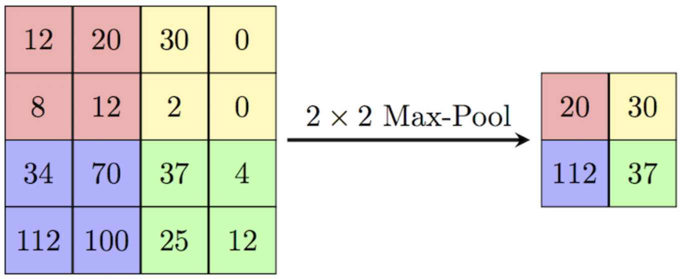


Figure 2.5: Max pool.

operation, it selects the max values in the receptive fields of the input, saving the indices and then producing a summarized output volume. The gradient of the backward pass is a zero matrix with the max index replaced by the value from last layer.Because there is no gradient for non max neurons, changing them slightly does not change the output. So gradient from the next layer is passed only to the neurons that achieved max, and all the other neurons get zero gradient.

Pixel shuffle Pixel shuffle Shi *et al.* (2016a) enlarges the size of feature map after convolution. To be specific, it transforms tensor with shape of $(*, r^2C, H, W)$ to shape $(*, C, rH, rW)$. In other words, pixel shuffle uses data in channels to resize tensor to larger scale, which can be used as a reverse operation of max pool.

Chapter 3

The Proposed Algorithm

In this section, we describe the structure and the functionality of the proposed Attention-based Multi-scale Matting Network (AMMNet). The AMMNet aims to learn an end-to-end mapping between a given RGB image and its associated alpha matte. Fig. 3.1 shows the whole pipeline. The AMMNet takes a three-channel RGB image as the input, and outputs a one-channel alpha matte. It does not need any auxiliary information such as trimaps or scribbles.

The AMMNet consists of three sub-networks: Multi-scale Semantic Extractor (MSExt), Attention Proposal Network (APNet), and Fusion Network (FNet). In the MSExt, 'nearest' mode interpolation is used to resize the original image to different scales. The APNet uses the original image as the input. The feature maps from MSExt and APNet are multiplied before being fed into the FNet. The AMMNet can simultaneously grasp global semantic information of the salient object and discern fine details such as hair, holes and semi-transparency. The three sub-networks are jointly trained end-to-end, without any individual sub-network pre-training. We discuss the sub-networks in detail in the following subsections.

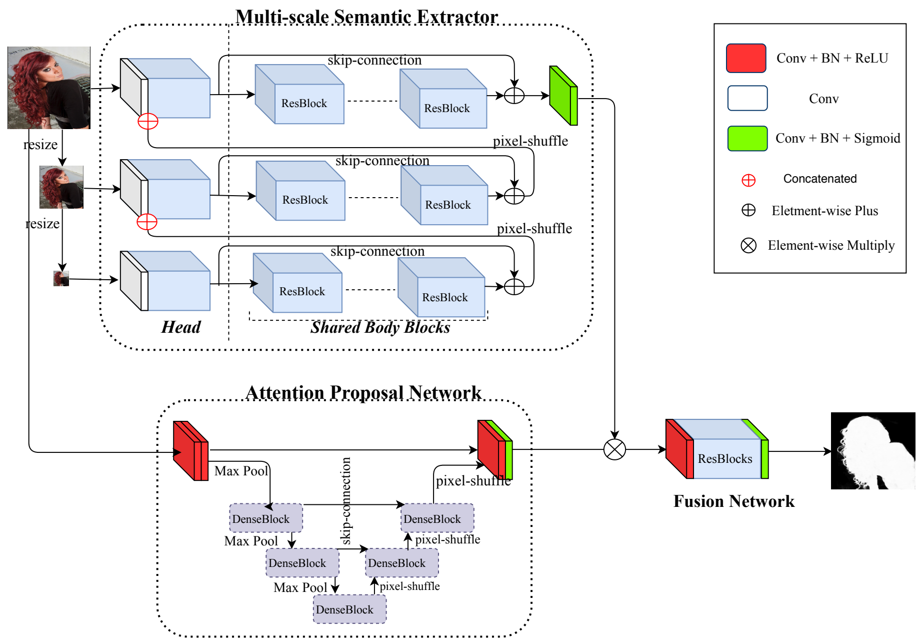


Figure 3.1: The architecture of AMSMNet, having three parts: upper part is the Multi-scale Semantic Extractor (Sec.3.0.1), lower part is the Attention Proposal Network (Sec.3.0.2) and the rightmost part is the final Fusion Network (Sec.3.0.3).

3.0.1 Multi-scale Semantic Extractor

The MSExt is designed to provide semantic information in the form of rough pixel-wise prediction of the foreground object. We use one convolutional layer as scale reader, and employ a pre-processing module (which is inspired by Lim *et al.* (2017)) as the ”head” for each scale, which consists of two residual blocks each with $5 * 5$ kernels. By using a large kernel size, one can reduce the scale-related influence while guaranteeing a large receptive field.

We construct each scale body with ResBlocks He *et al.* (2016b), which have been proved useful for various high- and low-level computer vision tasks. We pass each small-scale feature map to its successor scale by concatenating it with the features after the scale reader; in this step we choose pixel-shuffle Shi *et al.* (2016b) to ensure scale unification. Another measure to preserve the large receptive field is to use skip-connections across the scale body, which is also useful for avoiding vanishing gradients and boosting the training process. At the end of the largest-scale level, we use a convolutional layer with sigmoid activation as the output layer of MSExt.

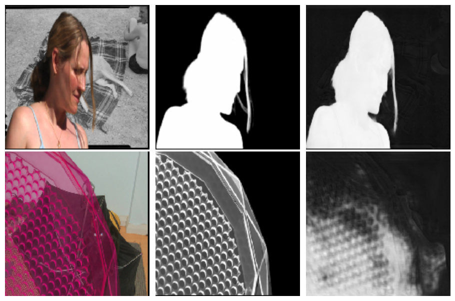


Figure 3.2: The first row shows an example of segmentation. The second and third rows are bad and good results of our attempt to transfer segmentation network to do alpha matting (with the matting specific loss in Sec.4).

The multi-scale structure is commonly used for semantic segmentation tasks. Note that

Semantic segmentation can be viewed as a simplified version of the image matting problem. Indeed, 1) the Ground Truth (GT) of segmentation is typically ”coarse” as compared to that of matting; 2) the segmentation problem concerns classification of blocks of pixels, whereas every pixel counts in the alpha matting problem; 3) segmentation produces a discrete result, but matting outputs floating-point values. The multi-scale structure developed for semantic segmentation is generally inadequate for image matting since much more expressive power is needed to handle the latter. To demonstrate this, we use the MSExt alone to perform alpha matting. It can be seen from Fig. 3.2 that the MSExt fails to deliver a satisfactory matte when the foreground object has a complex topological structure and/or is semi-transparent. See also Sec. 5.0.3 for some related experiment and discussion.

3.0.2 Attention Proposal Network

In view of the above discussion, we introduce an attention mechanism to enhance the expressibility of the whole system. Specifically, we use the APNet to supplement the MSExt by attending to high frequency features in the image, such as boundary, hair or holes.

As shown in Fig.3.1, the APN consists of a contracting path (left side) and an expansive path (right side). With the original RGB image as input, the contracting path extracts low level features using two convolutional layers. Max pooling is then used to reduce the size of data and to enlarger the receptive field. We then use Denseblock Huang *et al.* (2017b) with three layers within each block, and with dense growth rate to eight. There is a sequence of two denseblocks in the contracting path with maxpool downsampling between them. At the bottom is a shared Denseblock that is used by both the right and left part. In the expansive path, the structure is like a mirror version of the contracting path; pixelshuffle is used to keep the the feature maps align together dimensionally. Along the path of the APN, low-level features from an earlier stage might be lost as the network propagates forward. Therefore, a combination of low-level and high-level features in the expansive path is used to prevent information loss. This skip-connection structure is a key feature of the attention map. Lastly, two convolutional layers and a sigmoid activation function are used to output a floating point value between zero and one for each pixel.

The APNet takes the original RGB image as the input, and outputs a feature map attention mask. As shown in Fig. 3.1, the APNet consists of a contracting path (left side) and an expansive path (right side). This architecture is inspired by Unet Ronneberger *et al.* (2015). The contracting path extracts low-level features using two convolutional layers. Max pooling is used to reduce the size of data and to enlarge the receptive field. We adopt DenseBlocks Huang *et al.* (2017b) with three layers in each block and set the dense growth rate to 8. There are two DenseBlocks in the contracting path with maxpool downsampling between them. At the bottom is a DenseBlock shared by the two sides. The expansive path is mirror-symmetrical to the contracting path. Pixel-shuffle is used to keep the feature maps dimension-wise compatible. We introduce skip-connections between the contracting path and the expansive path to prevent information loss. Two convolutional layers and a sigmoid activation function are empolyed at the output end of APNet. The output of APNet, i.e., the feature map attention mask, is of the same size as that of MSExt, and consists of floating-point values between 0 and 1.

3.0.3 Fusion Network

The FNet takes the element-wise product of the outputs of MSExt and APNet as the input, passes it through three ResBlocks for feature integration, and produces the final alpha matte via a sigmoid function. Note that element-wise multiplication plays at least two roles:

1) it highlights the important semantic features and conceal the irrelevant ones (e.g., this happens when the attention mask assigns roughly the same weight to the elements of a semantic feature map); 2) it enriches semantic features with low-level details (e.g., this happens when the semantic feature is roughly a constant map and its elements are weighted diferently by the attention mask).

Chapter 4

Losses

In order to train the proposed AMMnet, we need to introduce a loss function that quantifies the difference between the predicted alpha matte $\hat{\alpha} \triangleq (\hat{\alpha}_i)_{i=1}^N$ and the GT α. The loss function adopted in this work has several components as described below.

4.0.1 Alpha-prediction Loss

The alpha-prediction loss Xu *et al.* (2017) provides a simple means for direct comparison of $\hat{\alpha}$ and α, and is defined as

$$\mathcal{L}_\alpha = \frac{1}{N} \sum_{i=1}^{N} f_S(\hat{\alpha}_i - \alpha_i), \tag{4.1}$$

where f_S is the smooth ℓ_1 loss, i.e.,

$$f_S(x) = \begin{cases} 0.5x^2, & \text{if } |x| < 1, \\ |x| - 0.5, & \text{otherwise.} \end{cases}$$

4.0.2 Compositional Loss

The compositional loss Xu *et al.* (2017) reflects the distortion in the RGB domain induced by the matte prediction error, and is defined as

$$\mathcal{L}_{com} = \frac{1}{3N} \sum_{i=1}^{N} \sum_{c \in \{R,G,B\}} f_S(\hat{I}_i^{(c)} - I_i^{(c)}), \tag{4.2}$$

where

$$\begin{gathered} \hat{I}_i^{(c)} = \hat{\alpha}_i F_i^{(c)} + (1 - \hat{\alpha}_i) B_i^{(c)}, \\ c \in \{R, G, B\}, \quad i = 1, \cdots, N. \end{gathered}$$

4.0.3 Content Loss

The alpha-prediction loss and the compositional loss are not necessarily good indicators of perceptual distortion. It is believed that the content loss Johnson *et al.* (2016) defined using certain feature maps produced by properly trained neural networks can better serve the purpose. However, evaluating the content loss of the predicted alpha matte $\hat{\alpha}$ is not very suitable because $\hat{\alpha}$ is largely deprived of content; it is also not very appropriate to evaluate the content loss of the composed RGB image $\hat{\mathcal{I}} \triangleq (\hat{\mathcal{I}}^{(c)})_{c \in \{R,G,B\}}$ (with $\hat{\mathcal{I}}^{(c)} \triangleq (\hat{I}_i^{(c)})_{i=1}^{N}$) since $\hat{\mathcal{I}}$ carries too much background information that is irrelevant to the alpha matte. For this reason, we take $\mathcal{I}_{\hat{\alpha}} \triangleq (\hat{\alpha} \odot \mathcal{I}^{(c)})_{c \in \{R,G,B\}}$ as the evaluation target, compared against its GT $\mathcal{I}_{\alpha} \triangleq (\alpha \odot \mathcal{I}^{(c)})_{c \in \{R,G,B\}}$, where $\odot$ denotes element-wise multiplication. Specifically, we define the content loss as

$$\mathcal{L}_{con} = \frac{1}{4} \sum_{j=1}^{4} MSE(\hat{\sigma}_j, \sigma_j), \tag{4.3}$$

where $\hat{\sigma}_j$ (σ_j) is the feature map of the j-th convolutional layer of the pre-trained VGG induced by $\mathcal{I}_{\hat{\alpha}}$ ($\mathcal{I}_{\alpha}$), $j = 1, 2, 3, 4$, and MSE denotes the Mean Squared Error.

4.0.4 Overall Loss

We define the overall loss as a weighted sum of $\mathcal{L}_{\alpha}$, $\mathcal{L}_{com}$, $\mathcal{L}_{con}$, and the square of the ℓ_2 norm of the parameter ensemble $\mathcal{W}$ of AMMnet:

$$\mathcal{L}_{all} = w_1\mathcal{L}_{\alpha} + w_2\mathcal{L}_{com} + w_3\mathcal{L}_{con} + \lambda\|\mathcal{W}\|^2, \tag{4.4}$$

where $w_1 = w_2 = 20$, $w_3 = 0.1$, $\lambda = 0.0004$. We choose a small value for w_3 (as compared to w_1 and w_2) because the content loss tends to be numerically large. The last term in Eq. (4.4) plays the role of ℓ_2 regularization (also known as ridge regression), which is used to penalize large network parameters and avoid overfitting.

Functionality of the Loss Function Fig. 4.1 is an illustration of the guided gradient map back-propagated from Fnet to MSExt and APNet. This map is used to update the parameters of MSExt and APNet. It can be seen that, under our chosen loss function, the gradient map carries the depth-like information, which is clearly useful for delineating the boundary between the foreground and the background of the RGB image and generating the alpha matte.

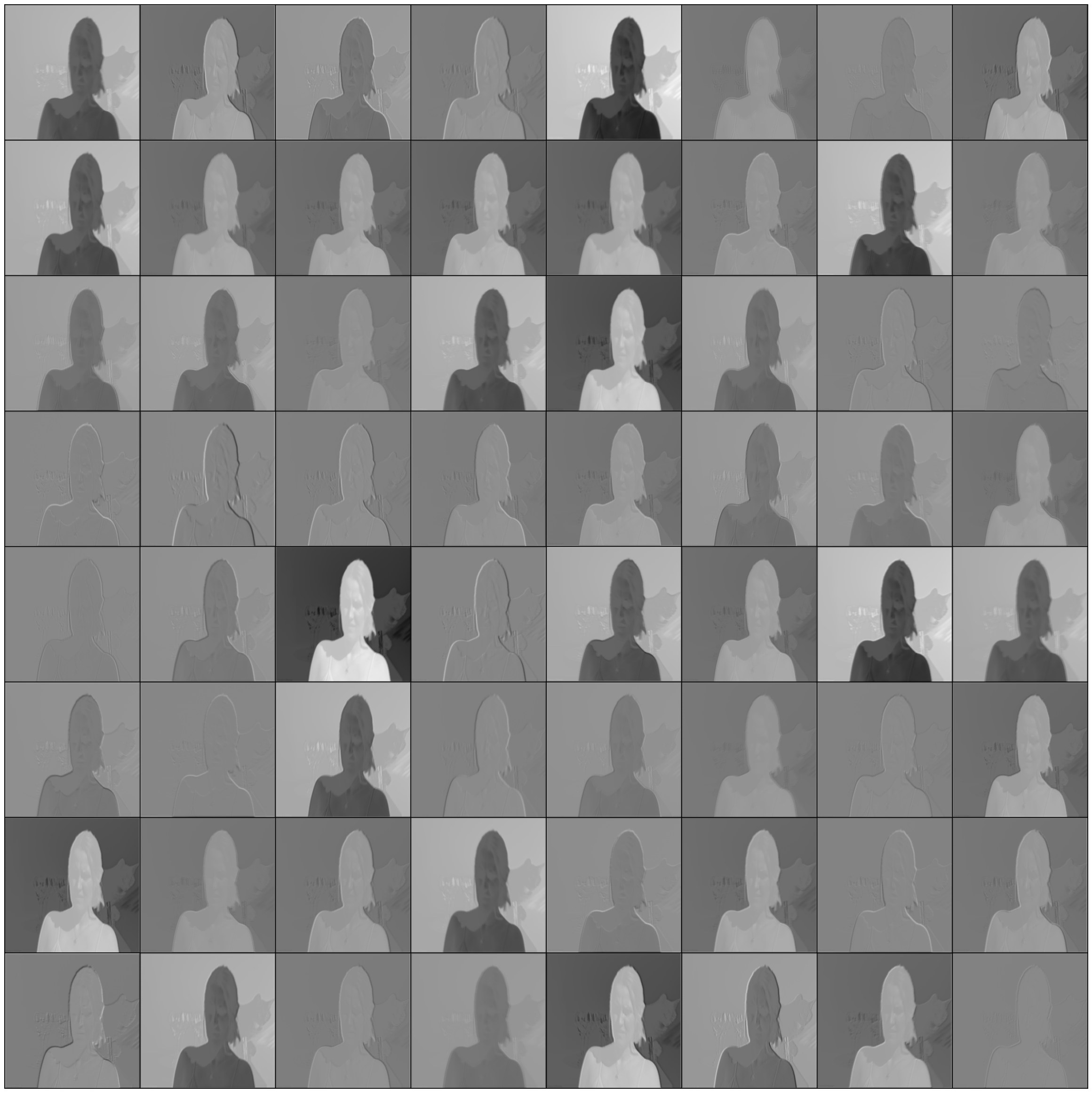


Figure 4.1: Guided gradient that back-propagates from fusion network, which will be used to update the weights of MSSE and APN.

Chapter 5

Experiments and Analysis

5.0.1 Datasets and Evaluation Metrics

We adopt two matting-related datasets for performance comparisons: the DAPM dataset Shen *et al.* (2016b) and the DIM dataset Xu *et al.* (2017). The DAPM dataset Shen *et al.* (2016b) consists of $19,000$ portrait images with a variety of re-scaling and color-transformation applied. The DIM dataset consists of $49,300$ images, created using 493 distinct foreground objects.

We consider four performance metrics: the Sum of Squared Errors (SSE) and the Sum of Absolute Difference (SAD), Gradient, and Connectivity Rhemann *et al.* (2009) defined respectively as

$$SSE(\hat{\alpha}, \alpha) = \sum_i (\hat{\alpha}_i - \alpha_i)^2, \tag{5.1}$$

$$SAD(\hat{\alpha}, \alpha) = \sum_i |\hat{\alpha}_i - \alpha_i|, \tag{5.2}$$

$$G(\hat{\alpha}, \alpha) = \sum_i |\hat{\nabla}_i - \nabla_i|, \tag{5.3}$$

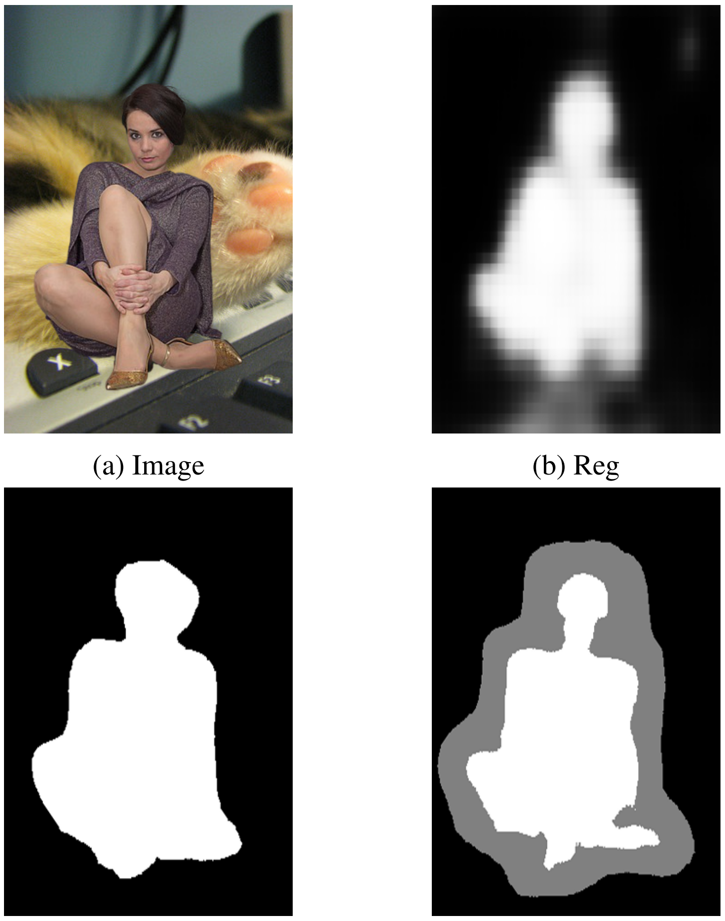
$$C(\hat{\alpha}, \alpha) = \sum_i |\hat{\varphi}_i - \varphi_i|, \tag{5.4}$$

where $\hat{\nabla}_i$ (∇_i) and $\hat{\varphi}_i$ (φ_i) denote, respectively, the gradient and the connectivity of $\hat{\alpha}$ (α) at pixel i. In (5.1)–(5.4), the summation is taken over all pixels for comparisons with trimap-free methods and is taken over the unknown region of the trimap for comparisons with trimap-dependent methods.

5.0.2 Training Details

Our experiments are implemented using the PyTorch (v1.0.1) framework on NVIDIA GTX 1080Ti GPU with 12GB memory. We train the AMMNet with RGB image patches containing both foreground and background information so that it can learn high-level semantic features and low-level details simultaneously; all the image patches are resized to 320×320. The input is standardized by ImageNet mean and variance Deng *et al.* (2009). We adopt random vertical flipping with probability 0.5. The overall loss in Eq. (4.4) is used for end-to-end training. All the parameters of AMMNet are initialized with Xavier random variables. We use the standard Adam optimizer in the PyTorch toolbox and set the batch size to 8. The initial learning rate is chosen to be 10^{-4}. The training process shows convergence after 60 epochs.

5.0.3 Comparison



(a) Image (b) Reg

(c) Seg (d) FCN8s+

Figure 5.1: Various adapted FCN-8s output. (b) is trained by alpha loss and compositional loss with ground-truth alpha. (c) is trained by cross- entropy loss and binarized alpha. (d) is trained by cross-entropy loss and three-valued trimaps.

To evaluate the effectiveness of the proposed AMMNet, we shall compare it with both automated matting methods and the interactive methods. Specifically, SHM Chen *et al.* (2018) is chosen as the baseline for automated matting, and the comparisons are performed on the DAPM dataset; as to interactive methods, we consider Closed Form (CF) matting Levin *et al.* (2008), KNN matting Chen *et al.* (2013), Information Flow Matting

Methods	SSE ($\times 10^3$)	SAD	Gradient ($\times 10^3$)	Connectivity ($\times 10^3$)
SHM Chen *et al.* (2018)	5.4	**10.4**	5.8	**9.3**
Our Method (AMMNet)	**5.1**	13.7	**4.5**	9.8
FCN-8s Seg	6.4	9.0	25.0	9.5
FCN-8s Reg	5.9	12.5	16.0	13.4
KNN Chen *et al.* (2013)	10.6	15.7	13.4	16.2
CF Levin *et al.* (2008)	10.9	15.4	10.6	15.8
IFM Aksoy *et al.* (2017b)	10.2	15.6	10.6	16.2
SM Gastal and Oliveira (2010)	9.0	12.7	16.3	12.9
DIM Xu *et al.* (2017)	3.6	6.6	4.7	6.6
Our Method (AMMNet)	**1.0**	**3.3**	**4.2**	**3.4**

Table 5.1: Performance comparisons of our method and some existing ones. The first two rows show the comparisons between two trimap-free methods, SHM and ours (in italic font), on the DAPM dataset, where the evaluation is performed over the whole image. The last eight rows show the comparisons of our method and some trimap-dependent methods (as well as FCN-8s Seg and FCN-8s Reg) on the DIM dataset, where the evaluation is performed over the unknown region specified by the trimap generated for FCN-8s+X. The best results are highlighted in bold.

(IFM) Aksoy *et al.* (2017b), Shared Matting (SM) Gastal and Oliveira (2010) as well as Deep Image Matting (DIM) Xu *et al.* (2017), and perform the comparisons on the DIM dataset. For the traditional interactive matting methods, the implementation offered by the Affinity-Based Matting Toolbox Aksoy (2017) is adopted; DIM is based on our PyTorch re-implementiation and is trained according to the approach outlined in Xu *et al.* (2017). We use FCN-8s Long *et al.* (2015) to supply trimaps to the aforementioned interactive methods (denoted as FCN-8s+X). Note that FCN-8s Long *et al.* (2015) was originally designed to perform segmentation. Here we adapt it for slightly different purposes. FCN-8s Seg is an adapted version for hard separation of foreground and background; it is trained with binary matte under the cross-entropy loss, where the binary matte is obtained by thresholding the GT alpha matte using the indicator function

$$Ind(x) = \begin{cases} 1, & \text{if } x > 0, \\ 0, & \text{otherwise.} \end{cases}$$

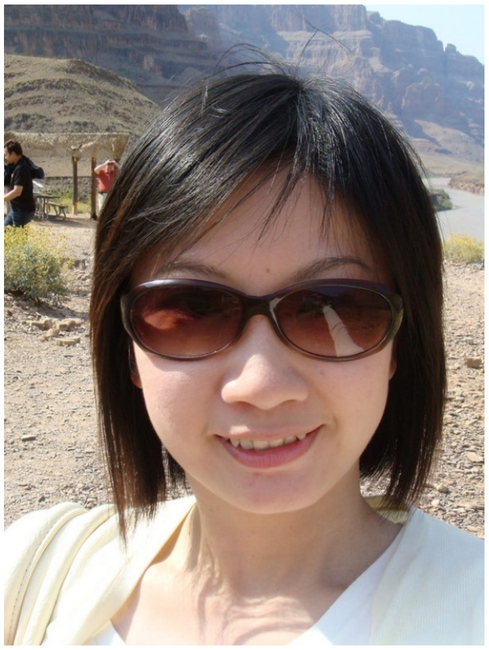
FCN-8s Reg is another adapted version, which can perform regression instead of segmentation; it is trained with the GT alpha matte under the alpha prediction loss and the compositional loss. For FCN-8s+X, we use FCN-8s to generate a ternary matte (consisting of foreground, background, and unknown regions) based on the input RGB image. This FCN-8s is trained with hand-crafted trimaps, and the training process is guided by the cross-entropy loss; in particular, the hand-crafted trimaps are obtained from the GT alpha matte using the standard dilating and eroding operations (with the MORPH_ELLIPSE kernel of size $d = 20$). An illustrative example of the outputs of the above adapted versions of FCN-8s can be found in Fig. 5.1.

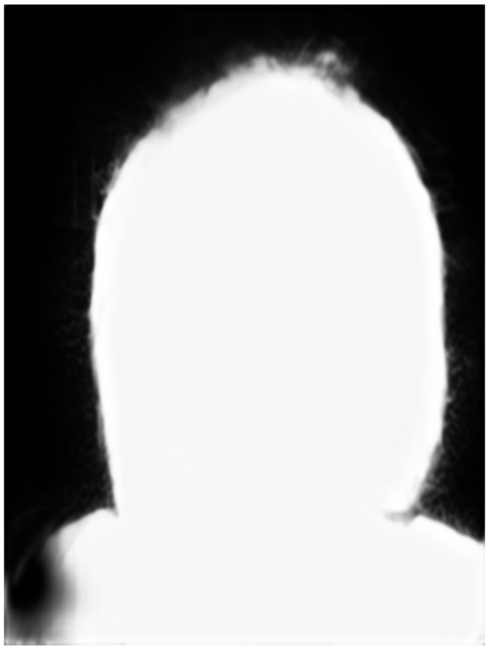
According to Table 5.1, FCN-8s Seg and FCN-8s Reg perform poorly on the DIM dataset. Indeed, it can be seen from Fig. 5.1 that FCN-8s Seg produces coarse boundaries while the matting results of FCN-8s Reg tend to be blurred. This can be attributed to the fact that the FCN-8s architecture, designed for the purpose of semantic segmentation, is ill-suited to matting. On the other hand, with the trimaps predicted by FCN-8s, the interactive methods can focus on learning delicate low-level details and generate more satisfactory matting results. In light of Table 5.1, DIM achieves the best performance among the interactive methods under consideration; it is also encouraging to see that the performance of the proposed automated method is only slightly inferior to that of DIM. Similar observations can be made from Fig. 5.3.

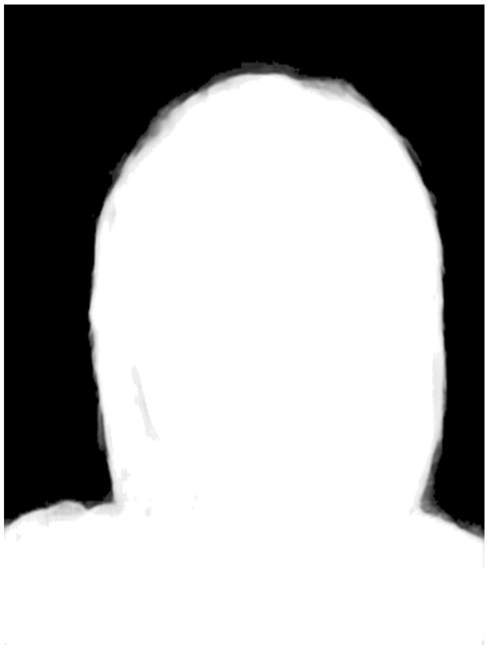
Analysis of Evaluation Results Table 5.1 provides the evaluation results of the proposed AMMNet and its competitors. It can be seen that SHM and our method achieve similar performance on the DAPM dataset. However, it should be noted that SHM is designed specifically for human matting whereas our method is generic in nature. The visual comparisons shown in Fig. 5.2 also indicate that the matting results of our method are comparable to, if not slightly better than, those of SHM.

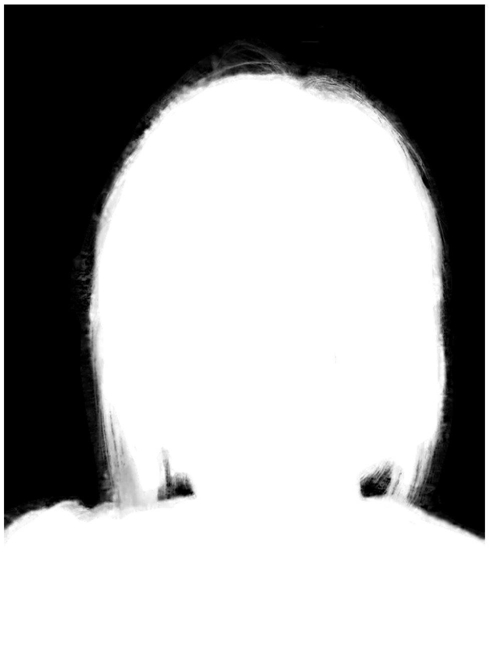
5.0.4 Sensitivity to the Quality of Trimaps

To better understand the advantage of our trimap-free method, it is instructive to study how sensitive the interactive methods are to the quality of trimaps. We use a subset of the DIM dataset for this study. The trimaps are generated by performing the standard dilating and eroding operations on the GT alpha matte. We control the quality of trimaps by adjusting the kernel size d. Specifically, we choose $d \in \{5, 20, 30, 60, 90, 120, 150, 180, 210, 300, 400, 500, 600\}$. Note that the quality of trimap degrades (in the sense that the unkown region gets enlarged)

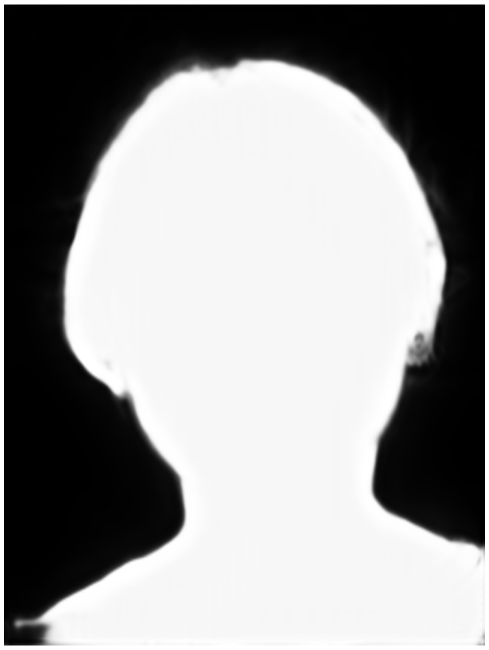

(a) Inputs

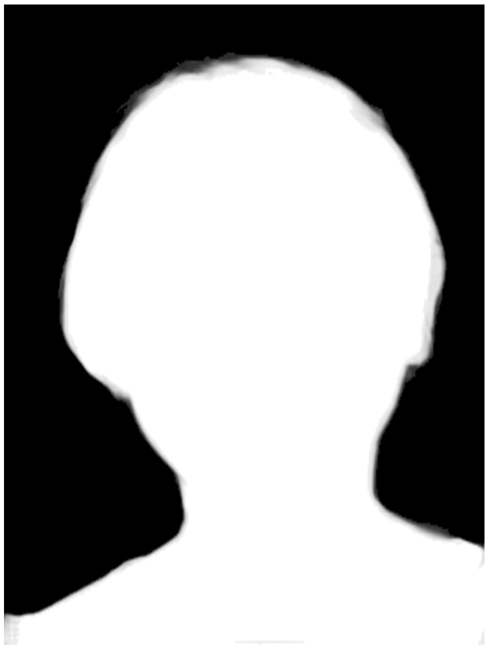

(b) SHMChen *et al.* (2018)

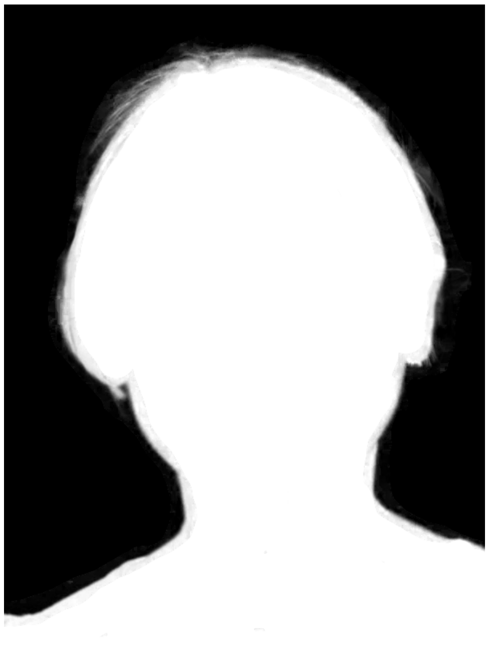

(c) Ours

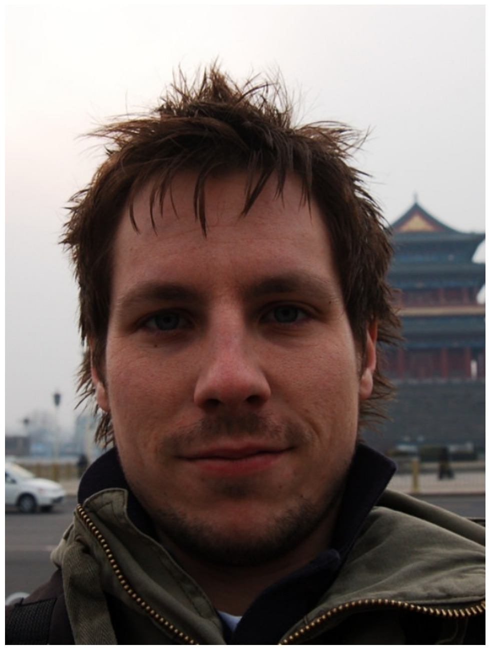

(d) GT

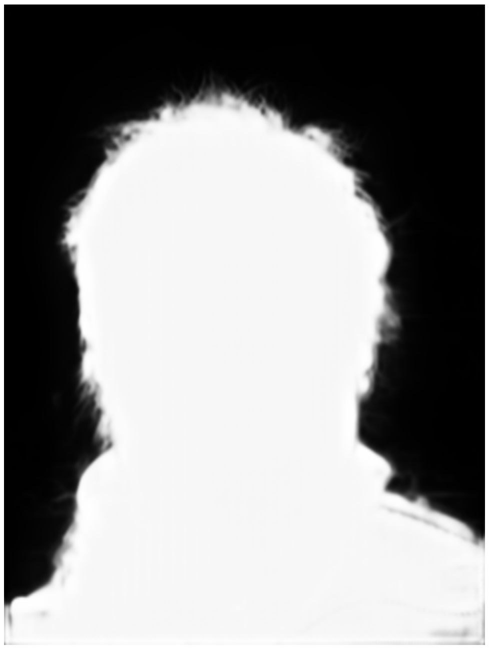
(e) Inputs

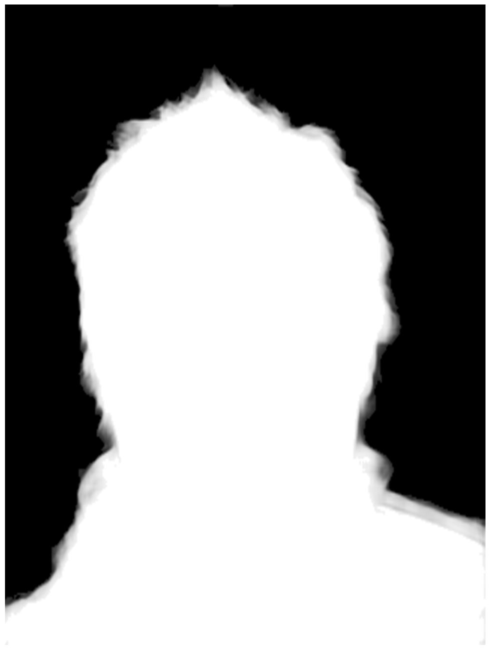

(f) SHMChen *et al.* (2018)


(g) Ours


(h) GT


(i) Inputs


(j) SHMChen *et al.* (2018)


(k) Ours

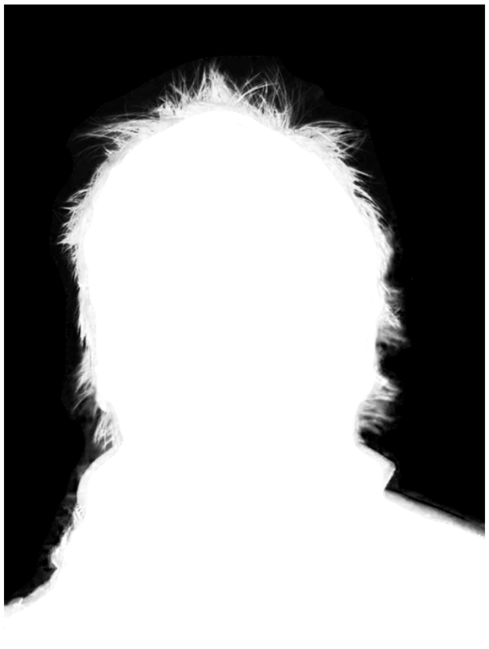
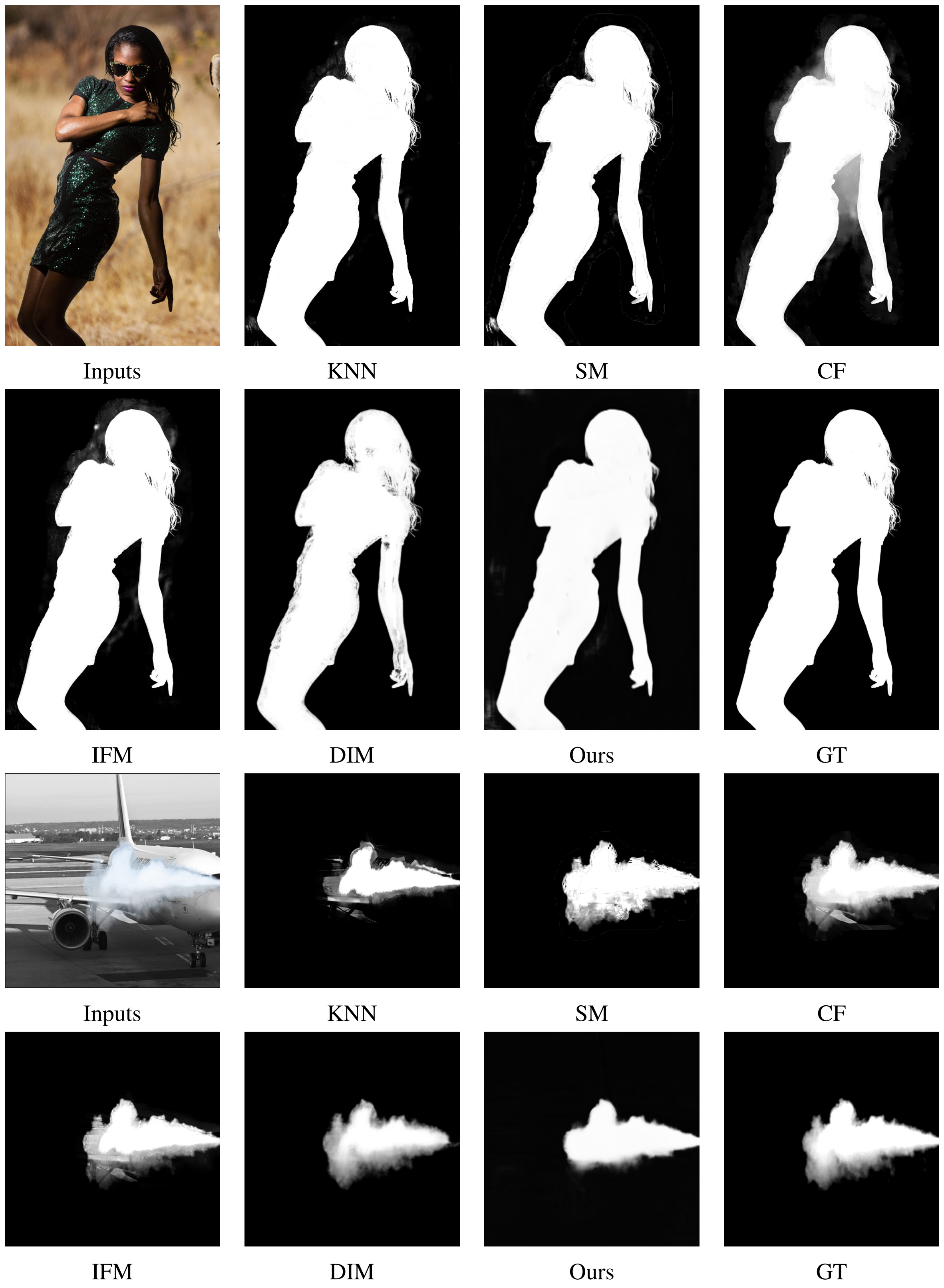
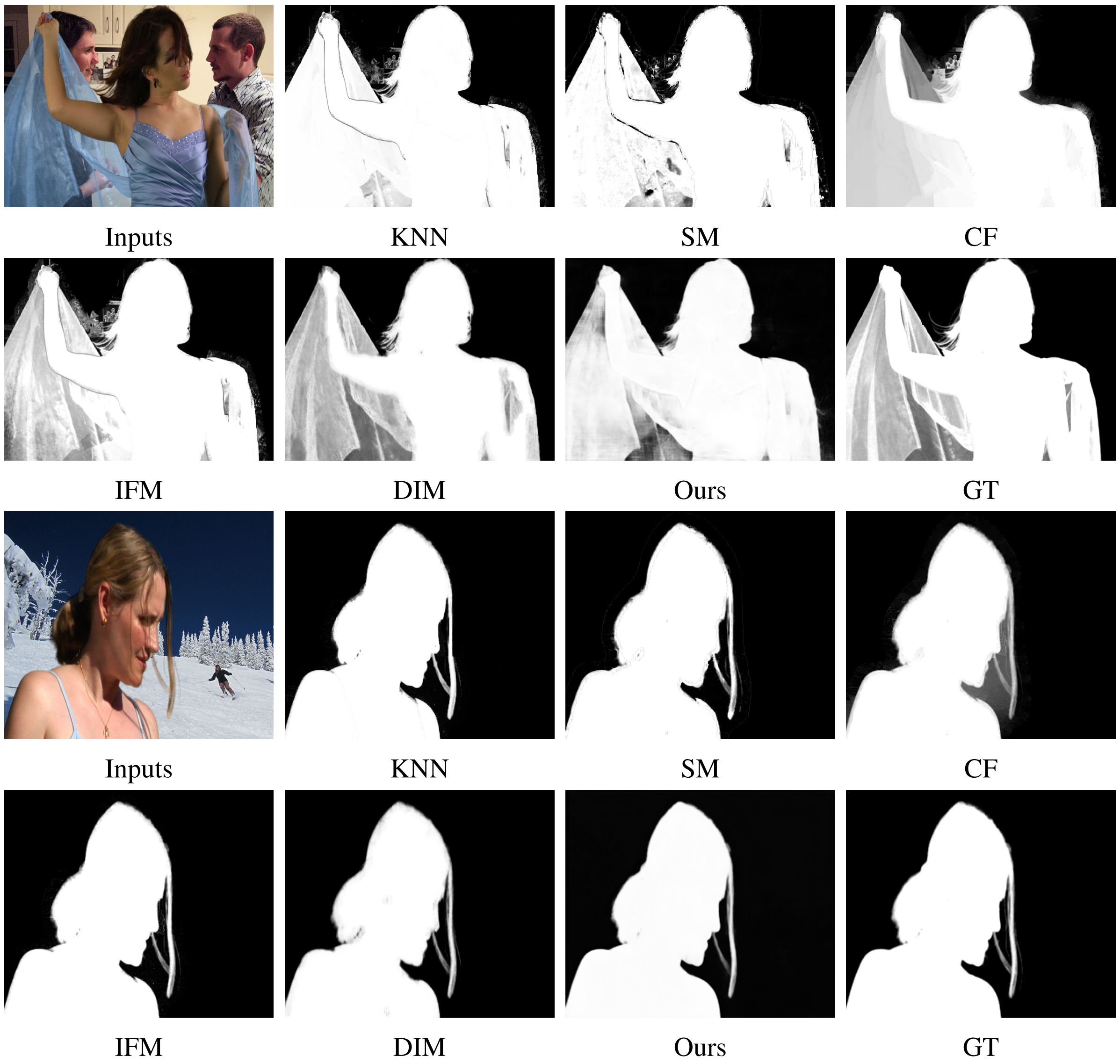

(l) GT

Figure 5.2: The visual comparison results on the DAPM Dataset between two automatic matting methods (SHM and ours).


Inputs
KNN
SM
CF
IFM
DIM
Ours
GT
Inputs
KNN
SM
CF
IFM
DIM
Ours
GT


Inputs
KNN
SM
CF
IFM
DIM
Ours
GT
Inputs
KNN
SM
CF
IFM
DIM
Ours
GT

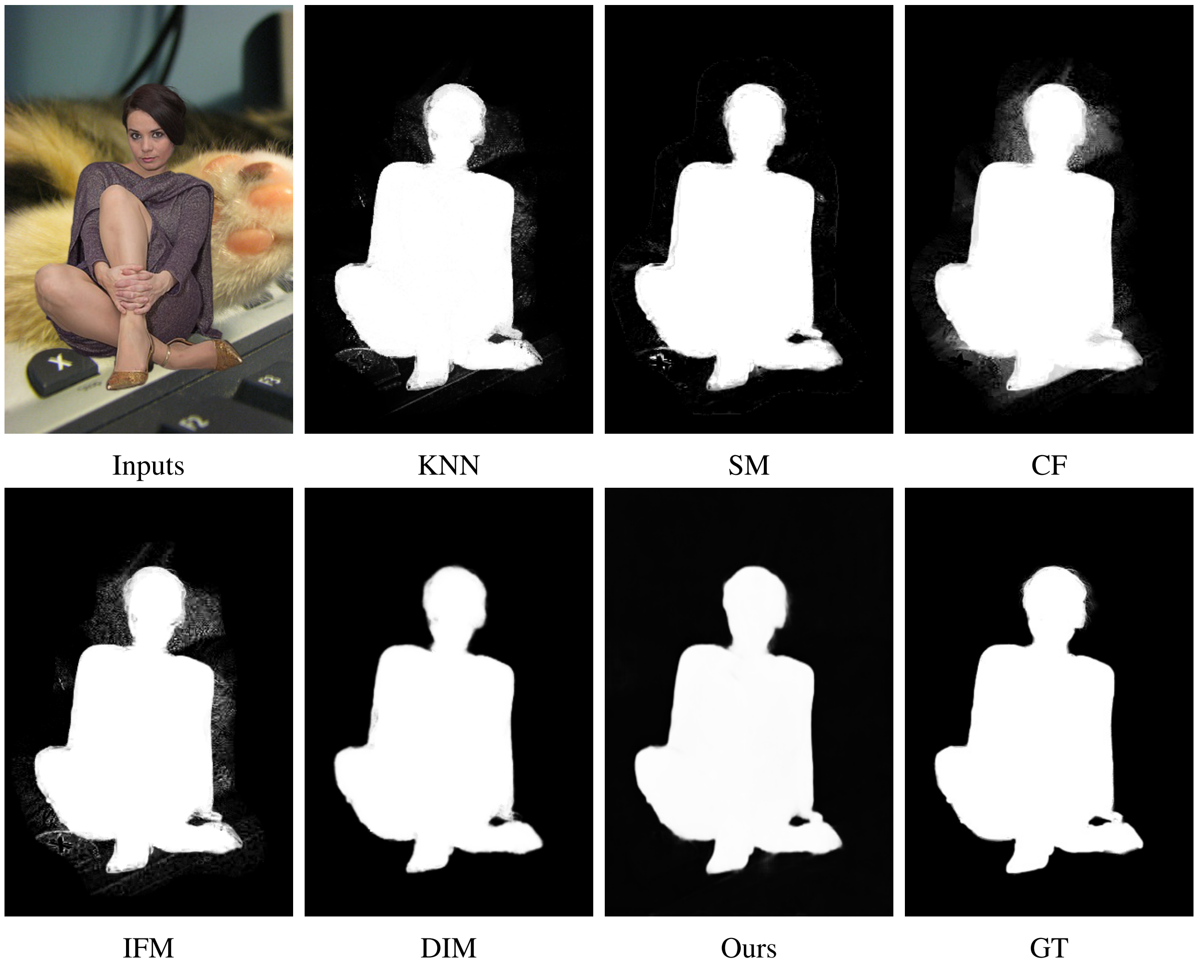


Figure 5.3: Visual comparison of some interactive methods test on DIM Dataset.

as d increases; in particular, the whole trimap is fully covered by the unknown region when $d = 600$.

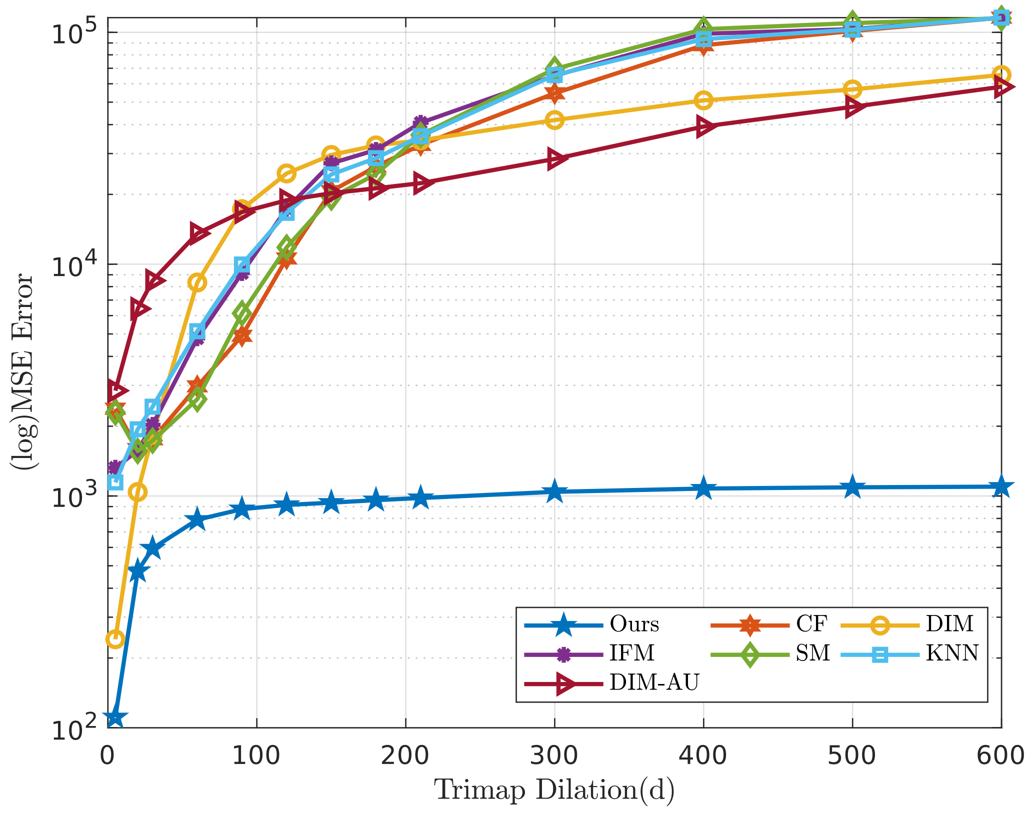


Figure 5.4: Figure of trimaps' side-effect.

It can be seen from Fig. 5.4 that the performance of DIM is quite competitive when $d \leq 20$ but quickly degenerates as d becomes large. DIM-AU is a variant of DIM, trained with all-unknown trimaps. It is more robust than DIM, but its overall performance is far from satisfactory, which suggests that the DIM architecture is inherently unsuitable for automated matting. In contrast, though slighlty inferior to DIM when d is small, our method outperforms all interactive methods under consideration by a wide margin for moderate or large d (it is worth mentioning that our method is trimap-free; its performance is d-dependent simply because the evaluation is performed on the unknown region, which varies with d). One can make similar observations based on the visual comparisons shown in Fig. 5.5.

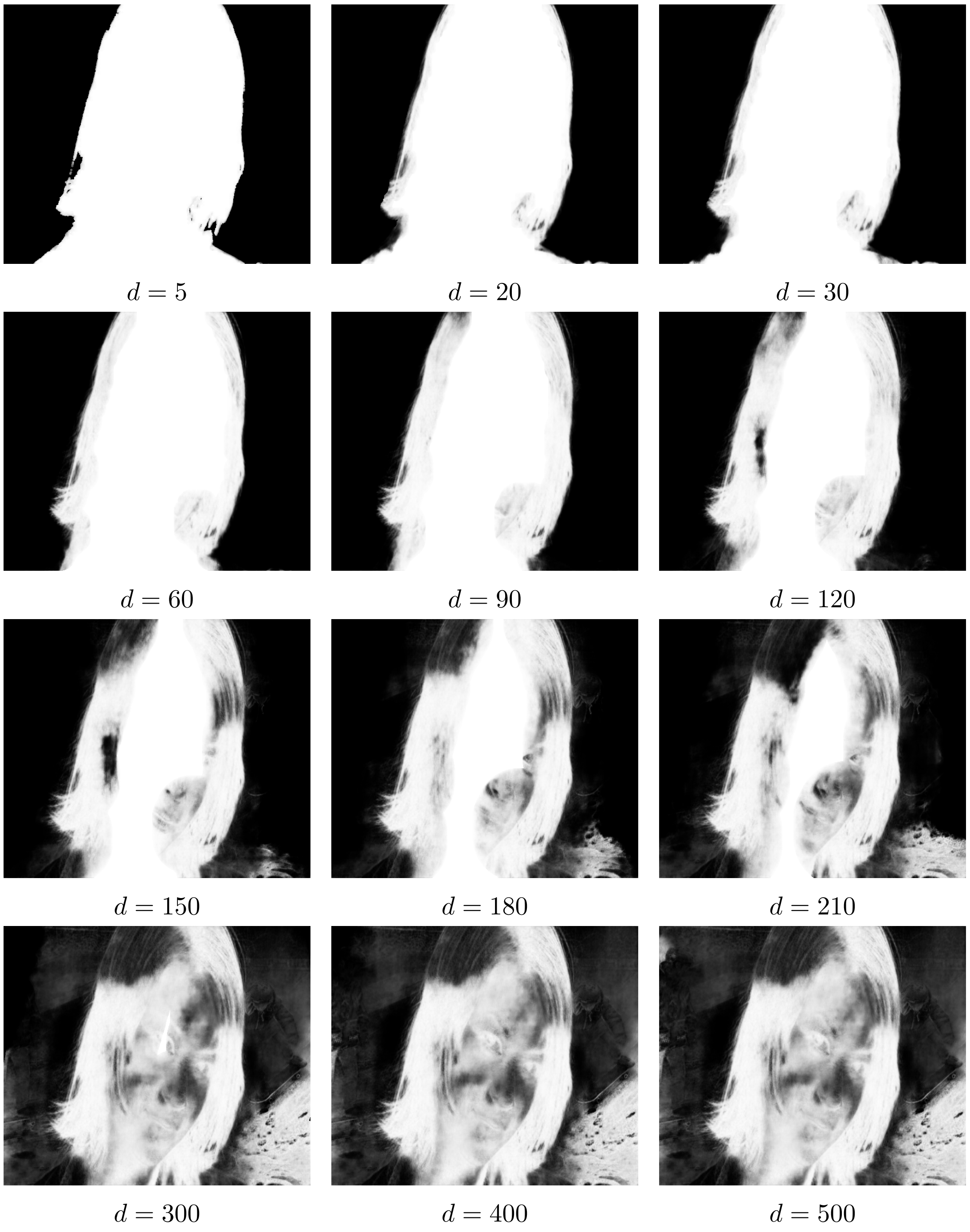
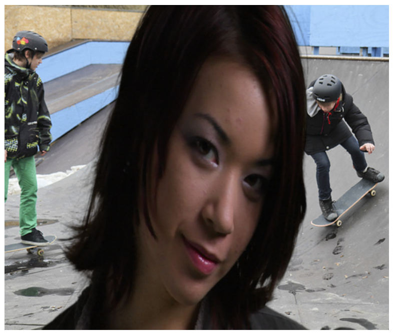
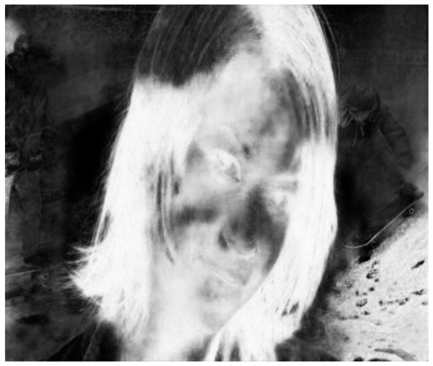
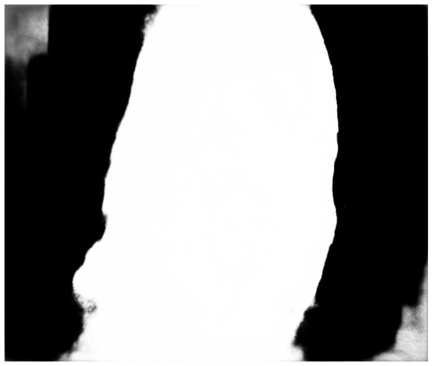
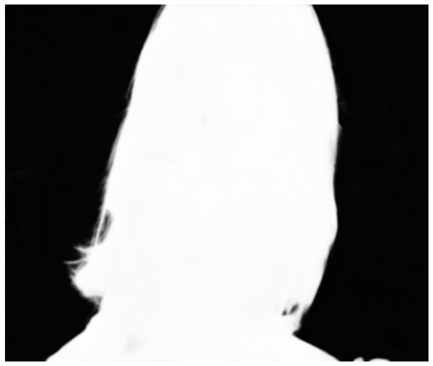


Figure 5.5: Trimap dilation result that represent the results trend with respect to dilation parameter d variation.


(a) Input


(b) $d = 600$


(c) DIM-AU


(d) Ours

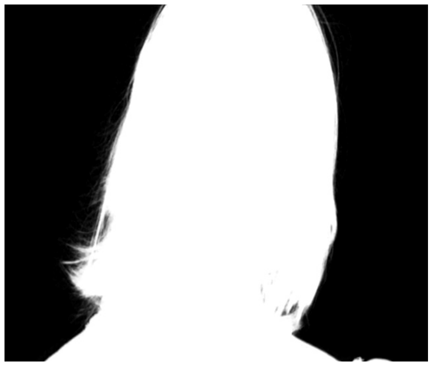

(e) GT

Figure 5.6: This figure shows methods that take an all-unknown trimap as input. (b) is derived from normal trained DIM network, while (c) is obtained from a network that trained on all-unknown trimaps.

Chapter 6

Conclusion and Analysis

In this work we have proposed a fully automated general-purpose image matting method, which can capitalize on both high- and low-level features for matte prediction. Specifically, a multi-scale structure is used in conjunction with a certain attention mechanism to identify the relevant global semantic information and local high-frequency details, which are then integrated by a fusion network to produce the final alpha matte. Experimental results indicate that the performance of the proposed method is competitive against the state-of-the art interactive/special-purpose matting methods.

As I mentioned in Chapter 1, alpha matting algorithm can be useful in several real world applications such as photography and e-commerce advertisement. It is also important when it comes to large-scale implementation. AMMNet, a tri-map free matting method can save time consumed to generate trimap and simplify the procedure to generate alpha matte. Furthermore, the Pytorch framework we used to train our network is a flexible and robust system that can be a good fit in real-time deployment. With high performance GPUs, it is quick and easy to fine-tune the model to new dataset in order to be adapted to other applications.

However, AMMNet is a little hard to train as it is trained from scratch and content loss is employed. I test a method from ShapeVision Inc. that uses PCD data in Chapter 7 to improve convergence time consumption. AMMNet uses attention mechanism to constraint the network learn features in key-area like boundary. PCD data is derived by persistence contour density, which takes a role of enhancing attention map. Therefore, with the help of PCD data we can save a lot training time.

Moreover, due to the idea of pixel-wise attention and fuse features with a sub-network, it could lay the groundwork for further research on attention mechanism and fusion methods. We expect that through training, our network can be transferred to other problems that need to use low- and high-level features together. To conclude, the present work not only proposed a novel approach of solving alpha matting problem, but also represents an idea of designing attention and fusion parts in neural network.

Chapter 7

Appendix

7.1 Overview

From an $n \times m$ source image we derive *persistence contour density (PCD) data* having the form of an $n \times m$ grayscale image. This data format provides easy integration with AMSMNet. In this section we describe the nature and derivation of the persistence contour density data.

PCD data for a colour or grayscale source image is computed by the following steps, each explained in subsequent sections.

PCD algorithm:

1. Convert source image to smaller grayscale image.
2. Construct Reeb graph from continuous interpolation of grayscale image.
3. Identify *cutpoints* in the Reeb graph using persistence analysis.
4. Derive from each cutpoint one or more *persistence contours* in the image plane.
5. Represent the spatial density of the persistence contours as a grayscale image at source image size.

7.1.1 Step 1: Convert source image to smaller grayscale image.

Size and grayscale conversion are standard image processing operations. The choice of conversion size affects the level of detail of the final result, with smaller size resulting in a smaller Reeb graph in step 2, fewer cutpoints in step 3, and therefore fewer persistence contours in step 4.

7.1.2 Step 2: Construct Reeb graph from continuous interpolation of grayscale image.

Let $n \times m$ be the pixel dimensions of the grayscale image that resulted from step 1. We interpret the pixel values as point data samples located at the vertices of an $n \times m$ grid of straight lines that lie on an $(n-1) \times (m-1)$ rectangular continuum, I. The $n \times m$ grid of lines delimit a grid of $(n-1) \times (m-1)$ unit squares. A continuous function within each grid square is defined by bilinear interpolation of the four corner samples; together, the interpolated grid squares define a continuous function f on the entire image plane I.

This interpolation is followed by construction of f's Reeb graph Reeb (1946). Denoting the Reeb graph as M, we let $\mu : I \to M$ denote the mapping of f's contours onto M, and we let $\lambda : M \to \Re$ assign to each point of M to the corresponding contour's f value.

The Reeb graph is combinatorial graph comprised of edges and vertices, having the topology of a graph continuum Jr. (1992). Each vertex of M is a *critical point*; all points along the edges are *regular*.

7.1.3 Step 3: Identify Reeb graph cutpoints using persistence analysis.

The notion of persistence Edelsbrunner and Harer (2010) is analogous to the mountaineering concept of *prominence* Wikipedia (2019), which measures the vertical distance between a mountain peak and the highest elevation to which one must descend from that peak in order to then ascend to an even higher peak.

In the Reeb graph, each maximum of f is represented by a vertex v having only arcs such that λ decreases as one moves away from v along the arc, and similarly for minima. The notion of persistence extends to extrema in the Reeb graph by pairing critical points Agarwal *et al.* (2006).

Let q_1, q_2 be any two points in Reeb graph M with $\lambda(q_1) > \lambda(q_2)$, and let P be the unique path in M connecting them. We say that path P *lies between* q_1, q_2 when, for any other point q along P, $\lambda(q_1) > \lambda(q) > \lambda(q_2)$.

Now let vertex v be a non-global maximum of M; then we can always find a vertex u with $\lambda(u) < \lambda(v)$, and a greater-valued maximum w, i.e. $\lambda(w) > \lambda(v)$, such that there exists a path P_1 lying between v and u, and a path P_2 lying between u and w. Choosing u so that $\lambda(v) - \lambda(u)$ is minimal, we get v's persistence $Per(v) = \lambda(v) - \lambda(u)$. Persistence is similar for local minima.

Given a local maximum v of Reeb graph M, we define v's *cutpoints* as v together with all points $u_1 \ldots u_n$ of M, including critical and regular points, such that $\lambda(v) - \lambda(u) = Per(v)$, and such that the path from v to each u_i lies between v and u_i. Cutpoints for local minima are similar. The cutpoints of Reeb graph M is the union of all cutpoints for all local extrema.

7.1.4 Step 4: Derive two or more persistence countours from each cutpoint.

Let p be any point in the Reeb graph M; then $\mu^{-1}(p)$ is a contour of f in the image plane. We know only that $\mu^{-1}(p)$ is closed and connected.

It is a fact that if $U \subset M$ is a connected open set containing p, then the set $U_p = U \setminus p$ is open, and has open connected components $U_p^1 \ldots U_p^n$. Choose U such that U_p contains no critical points, and choose any component U_p^i; then the set $V_p^i = \mu^{-1} U_p^i$ is open and connected in the image plane. It follows that the boundary, ∂V_p^i, has two connected components, each of which is either a Jordan curve or a curve connecting two points of ∂I, the boundary of the image plane. One of these two curves is a subset of $\mu^{-1}(p)$; we call it a *persistence contour*, denoted as C_p^i.

When U_p has $n > 1$ connected components, then we get n not necessarily distinct persistence contours $C_p^1 \ldots C_p^n$. Distinct persistence contours C_p^i and C_p^j may be disjoint, or may intersect at discrete points and/or along portions of the curves.

We now apply the above construction to every cutpoint v of M, collecting all distinct Jordan curves for each. These are the source images *persistence contours*.

7.1.5 Step 5: Represent the spatial density of the persistence contours as a grayscale image at source image size.

Recalling that the image plane comprises an $(n-1) \times (m-1)$ grid of squares, we can create an $(n-1) \times (m-1)$ grayscale image by counting the persistence contours that intersect each square, and then rescaling these counts to $[0255]$.

By resizing this image to the size of the source image, we get the desired persistence contour density data.

Bibliography

Agarwal, P. K., Edelsbrunner, H., Harer, J., and Wang, Y. (2006). Extreme elevation on a 2-manifold. *Discrete & Computational Geometry*.

Aksoy, Y. (2017). Affinity-based matting toolbox. `https://github.com/yaksoy/AffinityBasedMattingToolbox`.

Aksoy, Y., Aydin, T. O., Pollefeys, M., and Zürich, E. (2017a). Designing effective inter-pixel information flow for natural image matting. *Computer and Pattern Recognition (CVPR)*.

Aksoy, Y., Ozan Aydin, T., and Pollefeys, M. (2017b). Designing effective inter-pixel information flow for natural image matting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 29–37.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *Computer Science*.

Chen, Q., Li, D., and Tang, C.-K. (2013). Knn matting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **35**(9), 2175–2188.

Chen, Q., Ge, T., Xu, Y., Zhang, Z., Yang, X., and Gai, K. (2018). Semantic human matting. In *ACM Multimedia*.

Cho, D., Tai, Y.-W., and Kweon, I. (2016). Natural image matting using deep convolutional neural networks. *European Conference on Computer Vision*, pages 626–643.

Chorowski, J., Bahdanau, D., and Serdyuk, D. (2015). Attention-based models for speech recognition. *Computer Science*.

Chuang, Y.-Y., Curless, B., Salesin, D. H., and Szeliski, R. (2001). A bayesian approach to digital matting. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, **2**.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Edelsbrunner, H. and Harer, J. (2010). *Computational Topology - an Introduction.* American Mathematical Society.

Gastal, E. S. and Oliveira, M. M. (2001). Shared sampling for real-time alpha matting. *Computer Graphics Forum*, **29**, 575–584.

Gastal, E. S. L. and Oliveira, M. M. (2010). Shared sampling for real-time alpha matting. *Computer Graphics Forum*, **29**(2), 575–584. Proceedings of Eurographics.

Grady, L., Schiwietz, T., Aharon, S., and Westermann, R. (2005). Random walks for interactive alpha-matting. *Proceedings of VIIP*, pages 423–429.

He, K., Rhemann, C., Rother, C., Tang, X., and Sun, J. (2011). A global sampling method for alpha matting. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2049–2056.

He, K., Zhang, X., Ren, S., and Sun, J. (2016a). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

He, K., Zhang, X., Ren, S., and Sun, J. (2016b). Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017a). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708.

Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. Q. (2017b). Densely connected convolutional networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Johnson, J., Alahi, A., and Fei-Fei, L. (2016). Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*.

Jr., S. B. N. (1992). *Continuum Theory, An Introduction*. Marcel Dekker, Inc., New York.

Levin, A., Lischinski, D., and Weiss, Y. (2008). A closed-form solution to natural image matting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **30**(2), 228–242.

Li, G., Xie, Y., and Lin, L. (2018). Weakly supervised salient object detection using image labels. *arXiv preprint*.

Lim, B., Son, S., Kim, H., Nah, S., and Mu Lee, K. (2017). Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 136–144.

Liu, Y., Wang, Y., Li, N., Cheng, X., Zhang, Y., Huang, Y., and Lu, G. (2018). An attention-based approach for single image super resolution. pages 2777–2784.

Long, J., Shelhamer, E., and Darrel, T. (2015). Fully convolutional networks for semantic segmentation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440.

Reeb, G. (1946). Sur les points singuliers d'une forme de Pfaff complètement intégrable ou d'une fonction numérique. *C. R. Acad. Sci. Paris*, **222**, 847–849.

Rhemann, C., Rother, C., Wang, J., Gelautz, M., Kohli, P., and Rott, P. (2009). A perceptually motivated online benchmark for image matting. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1826–1833. IEEE.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer.

Sankaran, B., Mi, H., and Alonaizan, Y. (2016). Temporal attention model for neural machine translation. *arXiv*.

Shahrian, E., Rajan, D., Price, B., and Cohen, S. (2013). Improving image matting using comprehensive sampling sets. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 636–643.

Shen, X., Tao, X., Gao, H., Zhou, C., and Jia, J. (2016a). Deep automatic portrait matting. *European Conference on Computer Vision*, pages 92–107.

Shen, X., Tao, X., Gao, H., Zhou, C., and Jia, J. (2016b). Deep automatic portrait matting. In *European Conference on Computer Vision*, pages 92–107. Springer.

Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A. P., Bishop, R., Rueckert, D., and Wang, Z. (2016a). Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883.

Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A. P., Bishop, R., Rueckert, D., and Wang, Z. (2016b). Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1874–1883.

Sun, J., Jia, J., Tang, C.-K., and Shum, H.-Y. (2004). Poisson matting. *ACM Transacstions on Graphics(ToG)*, **23**, 315–321.

Wang, J. and Cohen, M. F. (2007). Optimized color sampling for robust matting. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

Wikipedia (2019). *Topographic prominence*. https://en.wikipedia.org/wiki/Topographic_prominence.

Xu, N., Price, B. L., Cohen, S., and Huang, T. S. (2017). Deep image matting. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 311–320.

Yu, F. and Koltun, V. (2015). Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*.

Zhang, P., Liu, W., and Lu, H. (2018). Salient object detection by lossless feature reflection. *arXiv preprint*.

Zhao, H., Shi, J., Qi, X., Wang, X., and Jia, J. (2015). Pyramid scene parsing network. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2881–2890.