# Enabling Trimap-Free Image Matting via Multitask Learning

# Enabling Trimap-Free Image Matting via Multitask Learning

By Chengqi Li,

*A Thesis Submitted to the Department of Electrical & Computer Engineering and the School of Graduate Studies in the Partial Fulfillment of the Requirements for the Degree Master of Applied Science*

McMaster University

Master of Applied Science (2021)

Hamilton, Ontario (Department of Electrical and Computer Engineering)

TITLE: Enabling Trimap-Free Image Matting via Multitask Learning

AUTHOR: Chengqi Li (McMaster University)

SUPERVISOR: Dr. Jun Chen

NUMBER OF PAGES: x, 55

*To my dear family and encouraging supervisor*

# Abstract

Trimap-free natural image matting problem is an important computer vision task in which we extract foreground objects from given images without extra trimap input. Compared with trimap-based matting algorithms, trimap-free algorithms are easier to make false detection when the foreground object is not well defined. To solve the problem, we design a novel structure (SegMatting) to handle foreground segmentation and alpha matte prediction simultaneously, which is able to produce high-quality mattes based on RGB inputs alone. This entangled structure enables information exchange between the binary segmentation task and the alpha matte prediction task interactively, and we further design a hybrid loss to adaptively balance two tasks during the multitask learning process. Additionally, we adopt a salient object detection dataset to pretrain our network so that we could obtain a more accurate foreground segment before our training process. Experiments indicate that the proposed SegMatting qualitatively and quantitatively outperforms most previous trimap-free models with a significant margin, while remains competitive among trimap-based methods.

# *Acknowledgements*

To start with, I would like to express my sincere gratitude to my supervisor, Dr. Jun Chen, for his kind support in this thesis work. In addition, I want to thank him for his constant encouragement, guidance, and assistance throughout my master's program, especially during my doctoral application.

It is my honor to have Dr. Xiaolin Wu and Dr. Shirani as my committee members, thank you for your reviewing and suggestions on my thesis.

My appreciation to Linhui Dai and Xiang Song for giving me inspiration every time I meet some trouble as well as for backing me up in those tedious experiments during the matting project.

Finally, I would like to thank Zhihao Shi, Xiaohong Liu and Minghan Fu for providing helpful advice and assistance throughout most of my projects.

# Contents

# List of Figures

# List of Tables

# Chapter 1

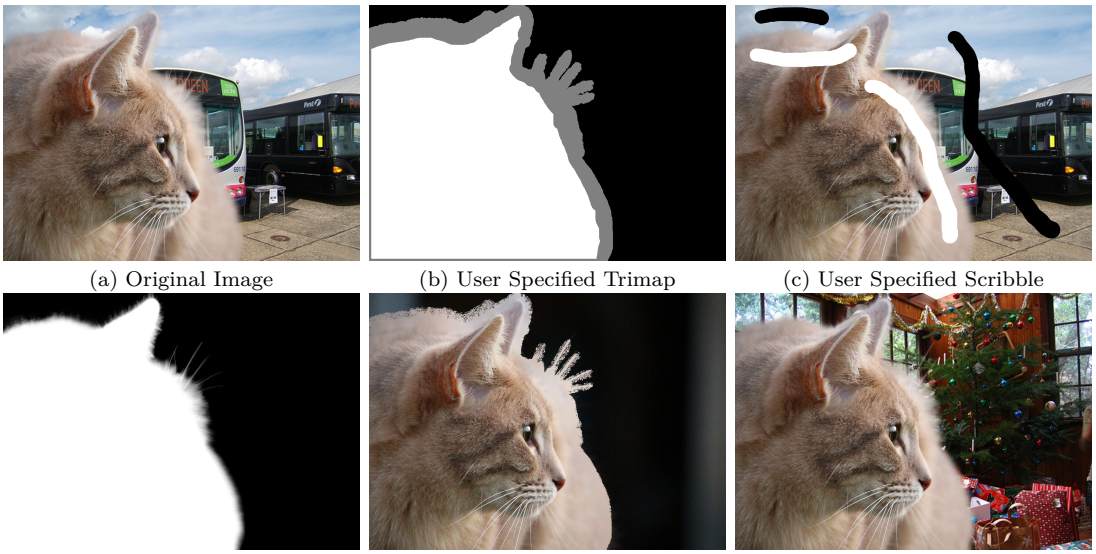
# Introduction and Problem Statement

## 1.1 Introduction

In natural image matting, our goal is to extract the target foreground object out from an image with background using the predicted alpha matte, and the whole process is shown in Fig. 1.1. There are a large amount of industrial natural image matting applications, including image synthesis, video editing, and film production. More precisely, we could formulate the problem as the following equation:

$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i, \quad \alpha_i \in [0, 1], \tag{1.1}$$

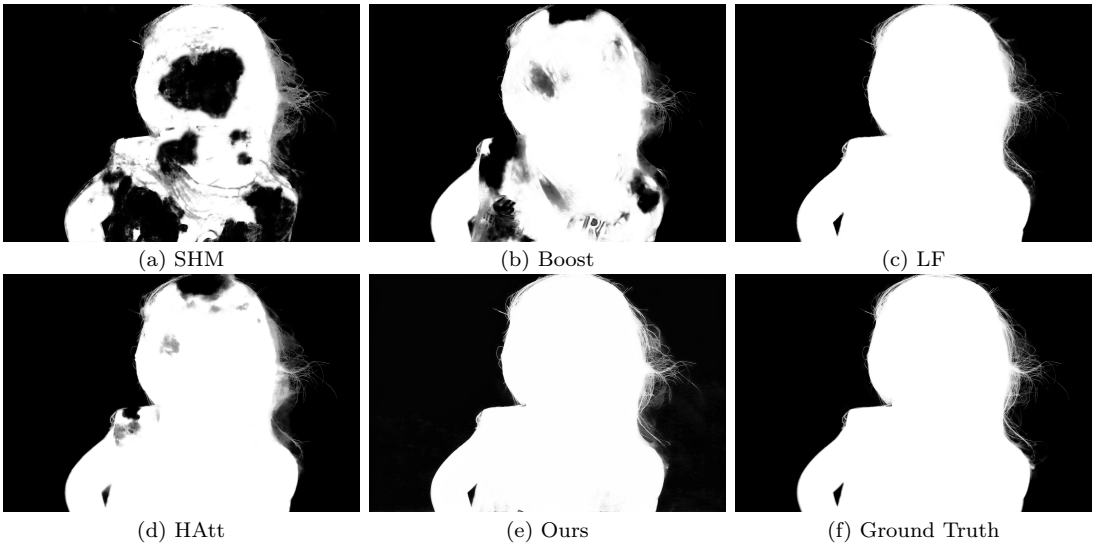
where $I_i$ represents the composite image at pixel $i$, on the right side of the equation, $F_i$ is the foreground (FG) image, $B_i$ is the background (BG) image, and $\alpha_i$ is the alpha estimation at pixel $i$. Since $F_i$ and $B_i$ are unknown, the matting problem is mathematically ill-posed.



(a) Original Image (b) User Specified Trimap (c) User Specified Scribble

(d) Alpha Matte (e) Estimated Foreground Object (f) New Composite

FIGURE 1.1: A matting example. We use predicted alpha matte to cut the FG object out and composite to a new BG.



(a) SHM (b) Boost (c) LF

(d) HAtt (e) Ours (f) Ground Truth

FIGURE 1.2: Artifacts in trimap-free algorithms.

To make Eq. 1.1 solvable, early works on image matting (Smith and Blinn 1996) resort to use solid color in the BG to constraint alpha value, yet such approaches are strictly limited by the object color and transparency in practice. Following this, a number of hand-crafted algorithms were proposed that attempted to learn the alpha value from the correlation between pixels within a local region in the image. In these affinity-based algorithms (Aksoy et al. 2017; Grady et al. n.d.; Levin et al. 2007; Sun et al. 2004), for instance, alpha mattes are derived on the assumption that $F$ and $B$ values are locally smooth. To estimate the alpha value of an unknown region, sampling-based methods (Chen et al. 2013; Feng et al. 2016; He et al. 2011; Shahrian and Rajan 2012; Wang and Cohen 2007) use information from the FG and BG patches. Unfortunately, these handcrafted methods, both affinity-based and sampling-based, suffer from performance degradation when applied to images with complicated BGs.

A series of data-driven matting algorithms (Cho et al. 2016; Xu et al. 2017; Li and Lu 2020; Lu et al. 2019; Hou and Liu 2019) have been introduced in recent years that are capable of generating high-quality alpha mattes with trimaps as user input. Nevertheless, creating user inputs is rather time-consuming and not feasible for real-time applications like video matting. Therefore, there has been an increasing interest in trimap-free approaches (Zhang et al. 2019; Chen et al. 2018; Qiao et al. 2020; Liu et al. 2020; Sengupta et al. 2020) that can determine alpha matte based exclusively on origin images. However, in most cases, trimap-free matting methods (Chen et al. 2018; Liu et al. 2020; Sengupta et al. 2020) are custom-designed for one certain type of object (e.g., humans). Moreover, even if trimap-free algorithms produce good mating results for certain natural images

with complex FGs, there is no reason to believe that they can generalize well in the real world because of the absent user input and the missing definition of FG objects associated with it (see Fig. 1.2).

In this thesis, we introduce SegMatting, an entangled network to handle the trimap-free matting problem. There are a few ideas behind our work design. First, since salient object detection (binary segmentation) and matting (alpha prediction) are highly correlated in our setting, we tackle these two tasks via multitask learning and let them reinforce each other through the communication mechanism integrated inside our network. Second, we argue that it is essential to adaptively balance two correlated tasks during different stages of the training process. Generally, in multitask learning, different tasks are uniformly weighted, but we design a hybrid loss which contains learnable parameters to adjust the focus throughout the training process. Lastly, we believe that extra prior knowledge on object saliency would be beneficial, since the problem is highly ill-posed and lacks location clue. Pre-training on a salient object detection dataset would enable our model to pinpoint the FG object.

In summary, the main contributions of this paper are:

- By utilizing the entangled decoder and task switcher module of the network structure, we employ multitask learning to tackle both alpha prediction and binary segmentation tasks concurrently and optimize the information flow in-between the tasks.
- A hybrid loss function is designed that allows adaptive multitask learning in terms of binary segmentation and alpha prediction on a multi-scale basis.

- Using salient object detection data as a pre-training dataset, we exploit the correlation between trimap-free matting and salient object detection to refine our model for trimap-free natural image matting.

In addition to all the features mentioned above, our network performs better qualitatively and quantitatively than most existing trimap-free algorithms while still being competitive with trimap-based mating algorithms.

## 1.2 Thesis Structure

First, we will review several alpha matting algorithms and discuss their connections with our work in chapter 2. In the same chapter, there is a short derivation of our hybrid multitask loss. Then, chapter 3 will describe the proposed method in detail, including the encoder, the tangle decoder, the task switcher, the refinement module, the loss function, and the pretrain strategy. In chapter 4, qualitative and quantitative results are provided to demonstrate the superiority of our network as well as the effectiveness of network modules and training strategies. Finally, in chapter 5, there will be a conclusion about the proposed method, its performance, and a potential future solution to problems found in experiments.

# Chapter 2

# Background and Previous Work

## 2.1 Natural Image Matting

Natural Image matting aims to make a pixel-wise prediction for object opacity. It has been a popular computer vision task for few decades. The natural image matting algorithms could be divided into two major categories, the probability model based traditional methods and the convolutional neural network (CNN) based methods.

Tradition matting algorithms branch into two categories, namely, affinity-based methods and sampling-based methods. Affinity-based methods make assumptions on the smooth color distribution on the local area, and propagate this clue to solve an optimization system in the closed form (Aksoy et al. 2017; Grady et al. n.d.; Levin et al. 2007; Sun et al. 2004). In the closed-form matting (Levin et al. 2007), authors first established a linear relation between the alpha value and the RGB value from Equation 1.1, and then an objective function is derived based on the local smoothness assumptions inside a small kernel. Starting from pixels given by

scribbles, the alpha values are solved by minimizing the objective function, and the process is carried out gradually throughout the entire image. On the other hand, sampling based methods (Chuang et al. 2001; Wang and Cohen 2007; He et al. 2011; Shahrian and Rajan 2012; Chen et al. 2013; Feng et al. 2016) collect samples (image patches) from FG and BG areas to find the given pixel's FG and BG color and use those to solve alpha value based on Equation 1.1. For example, with user markups, KNN matting (Chen et al. 2013) uses FLANN to find $K$ nearest neighbors for each pixel in feature space, and then apply the nonlocal principle to obtain a closed-form solution. Though different sampling methods are used in these works, they are all performed in certain areas located by trimaps.

The field of computer vision has witnessed the rapid development of deep learning in recent years. Learning-based matting algorithms dominate the natural image matting tasks because they could deliver better results even if they don't require prior assumptions adopted by traditional algorithms. DCNN (Cho et al. 2016) was the first work that introduced deep learning methodology into natural image matting, but it only used a deep neural network to refine alpha mattes generated by traditional methods. Utilizing the newly introduced Adobe Composition-1k dataset, DIM 2.1 proposed an encoder-decoder network to generate the alpha matte from scratch, and its novelty also includes the hybrid loss (weighted sum of the alpha loss and the composition loss). Based on the encoder-decoder structure, (Lu et al. 2019; Hou and Liu 2019; Li and Lu 2020) try to improve the matting performance by indexing the pooling process, adding attention mechanism, and even duplicating the structure respectively. Similar to most matting algorithms, our model utilizes encoder-decoder structure with modification in the decoder, and

it can also be trained in an end-to-end manner.

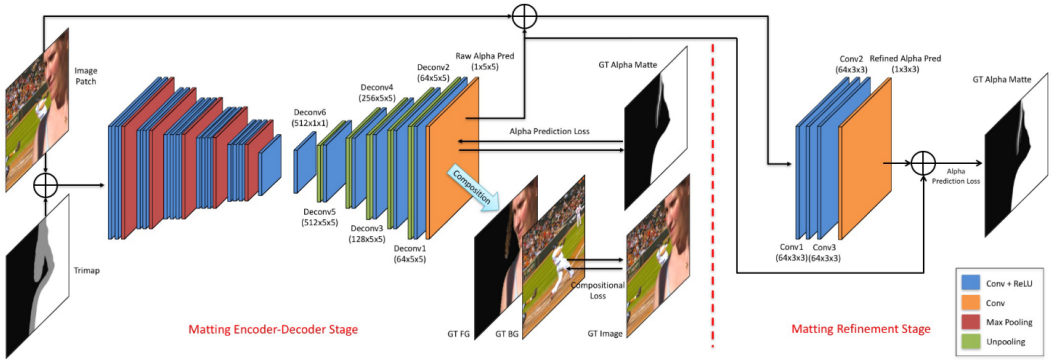



FIGURE 2.1: The architecture of DIM. (Image originally used in Xu et al. 2017)

## 2.2 Trimap-free Matting

Despite the different underlying assumptions and methodologies, the aforementioned algorithms all rely on user inputs (e.g. trimaps or scribbles) to solve the extremely ill-posed problem in which we need to compute seven unknown values with only three given ones.

Fig. 1.1 shows examples of user input trimap and scribble. The trimap is a single channel mask in which FG, BG, and unknown region pixels are labeled 0, 255, and 128 respectively. Since a precise trimap usually requires hours of human effort, the real time implementation of trimap-based algorithms is impossible. Compared with trimaps, scribbles are hand-drawn lines that indicate approximate locations of the FG and the BG, and they are much easier to generate. Even though it takes only a few seconds to create input scribbles, it is unrealistic to allow people to do so when implementing massive image matting (millions of images). Because

scribbles offer limited information, algorithms that utilize them frequently perform worse than those that use trimaps.

Due to the inconvenient process of generating user inputs in certain applications (e.g. real-time video matting), people resort to trimap-free matting which using only the RGB image to predict the opacity. Since matting without the user input is extremely difficult, some works start to replace the precise-labeled trimap with other user inputs that could be obtained more easily. Boosting matting (Liu et al. 2020) proposed a series of networks which takes coarse annotated trimaps in and refines them with the quality unification module before re-joining the normal matting backbone. Though coarse annotated trimaps are easier to get, it is impossible to do so in real-time. A further step is taken by background matting (Sengupta et al. 2020), it utilizes multiple priors to replace the trimap, but it still requires some effort in gathering such priors (pre-obtained background images and automatically calculated soft segmentation).

Different from the aforementioned algorithms which partially get rid of trimaps, a few works (Chen et al. 2018; Zhang et al. 2019; Qiao et al. 2020) attempt to compute the opacity map by embedding the localization logic into the network architecture so that user input could be avoided. Semantic human matting(Chen et al. 2018) leverages CNN to segment the image content first, and then uses the segmentation information to guide the alpha prediction. Furthermore, late fusion matting (Zhang et al. 2019) branches out two segmentation decoders, one for locating the foreground and the other for the background, and then fuses foreground and background information to get an accurate alpha matte. HAttMatting (Qiao et al. 2020) designs a hierarchical attention module to guide the alpha prediction.

Nevertheless, the lack of object location clues can introduce false detection when extracting foreground from complex background. To solve the problem, we reinforce the location information by an entangled network structure in which the segmentation task and alpha matting task are interacting with each other at every sampling level, and also by pretraining our network on a salient object detection dataset.

## 2.3 Attention Mechanisms

In a variety of deep learning models, attention mechanisms have been used to enhance the performance of various tasks, including machine translation (Bahdanau et al. 2016; Vaswani et al. 2017) and computer vision (Wang et al. 2018; Hu et al. 2018; Woo et al. 2018; Hu et al. 2019).

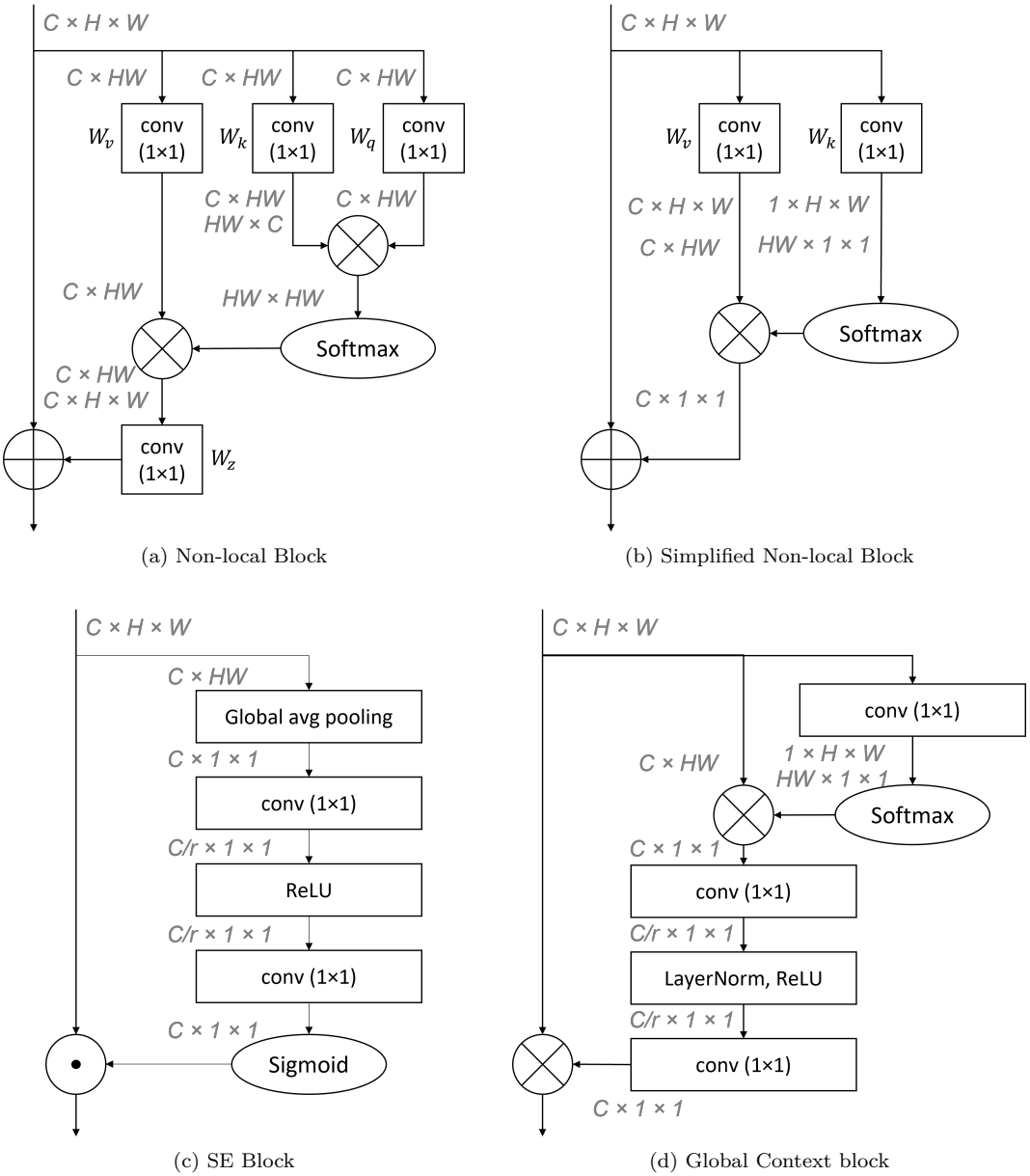
In (Wang et al. 2018), authors try to establish a non-local dependency using the pixel-wise correlation matrix, since convolutional and recurrent operations could only utilize the local information. Such dependency is further developed into an attention mechanism that helps to guide the contextual information aggregation spatially. The feature information also distributes over channels, so SEnet (Hu et al. 2018) proposed a totally different channel-wise attention to weighing the importance of feature maps in each channel. Later, a series of papers explored different implementing strategies for attention mechanism. For example, (Woo et al. 2018) introduced a combination of spatial and channel-wise attention, (Fu et al. 2019) used dual attention branches, and (Huang et al. 2019) applied attention sequentially.

In image matting, (Li and Lu 2020) introduces the idea of guided contextual attention that uses low-level features from different areas of the RGB image to guide the propagation of the alpha feature. Though we have a pretrained entangled network for our trimap-free matting task, we could further leverage the attention mechanism to provide extra information. Therefore, we adopt the global contextual block (Cao et al. 2019), which is a simpler and more efficient variation of non-local attention block, and the SE block for spatial and channel-wise attention respectively. Illustrations for several attention blocks are given in Fig. 2.2.

## 2.4 Multi-Task Learning

In a multi-task learning setting, multiple related learning objectives are carried out simultaneously. Since those objectives share the same representation in the latent feature space, multi-task learning enables them to reinforce each other, and thus increases the accuracy of a model while efficiently learning multiple objectives (Caruana 1997).

In the computer vision area, multi-task learning has been implemented in many works (Liao et al. 2015; Teichmann et al. 2018; Eigen and Fergus 2015; Uhrig et al. 2016; Zhang et al. 2018b). Many of them focus on the segmentation task since the segmentation task is a prerequisite of or closely related to many other CV tasks, such as classification, object detection, depth estimation, and alpha matting. The connection between classification and segmentation is utilized in (Liao et al. 2015). Using the shared representation extracted through encoder network, MultiNet (Teichmann et al. 2018) is able to accomplish detection, classification, and semantic




(a) Non-local Block

(b) Simplified Non-local Block

(c) SE Block

(d) Global Context block

FIGURE 2.2: Architecture of the attention blocks. $C \times H \times W$ represents a feature map with channel number $C$, height $H$ and width $W$. $\otimes$ denotes matrix multiplication, $\oplus$ denotes element-wise addition, and $\odot$ denotes broadcast element-wise multiplication.

segmentation. (Eigen and Fergus 2015) attempts to gain an accurate depth estimation based on the information provided from the jointly trained segmentation and surface normal. However, in these methods, weights of losses are either set uniformly or be treated as hyperparameters that are fine-tuned manually, such a naive setting deteriorates the performance of the main task and requires tedious tuning.

In (Kendall et al. 2018), the author suggests that the performance of a multi-task learning model is strongly dependent on the relative weighting between each task's loss, and he proposed a learned multi-task loss base on the homoscedastic uncertainty of each task.

In matting task, (Cai et al. 2019) leverages multi-task learning to predict the trimap and the alpha matte simultaneously. However, we argue that these two tasks are only loosely connected via the loss at the final output level, and the main alpha matte prediction branch does not make full use of the task correlation. More importantly, (Cai et al. 2019) requires a trimap input during the inference process. Inspired by (Zhang et al. 2018b), we proposed an entangled structure that facilitates the joint task learning between alpha matte prediction and binary segmentation and maximizes the information flow between two tasks throughout the decoder network. Meanwhile, the learned multi-task loss (Kendall et al. 2018) is adopted in our work, and it is modified to give dynamic supervision on the alpha prediction task at the binary segmentation task at every sampling level.

Here starts the derivation of the learned multi-task loss for our alpha matting prediction (regression) and binary segmentation (classification) combination.

Basically, it is to weigh two maximum likelihood objectives and to sum them afterward. Assume $f^{\mathbf{W}}(\mathbf{x})$ is the output of a neural network with parameter $\mathbf{W}$ when the input is $\mathbf{x}$. For regression task (alpha matte prediction), it follows a normal distribution, so we define our likelihood function with mean given by the model output and variance $\sigma^2$:

$$p(\mathbf{y} \mid f^{\mathbf{W}}(\mathbf{x}), \sigma) = \mathcal{N}(f^{\mathbf{W}}(\mathbf{x}), \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{\left\|\mathbf{y} - f^{\mathbf{W}}(\mathbf{x})\right\|^2}{2\sigma^2}). \tag{2.1}$$

With all ground truth $\mathbf{y}$ known, we want to maximize the likelihood function so that the prediction $f^{\mathbf{W}}(\mathbf{x})$ would be more accurate and general. For simplicity, we rewrite the above equation in a log-likelihood form and remove scalars:

$$\log p(\mathbf{y} \mid f^{\mathbf{W}}(\mathbf{x}), \sigma) \propto -\frac{1}{2\sigma^2} \left\|\mathbf{y} - f^{\mathbf{W}}(\mathbf{x})\right\|^2 - \log \sigma. \tag{2.2}$$

For classification (segmentation), a softmax function is often used to normalized the network output. In our case, there are only two classes, the FG and the BG, and the likelihood function could be formulated as follow:

$$p(\mathbf{y} \mid f^{\mathbf{W}}(\mathbf{x})) = Softmax(f^{\mathbf{W}}(\mathbf{x})) = \frac{\exp(f^{\mathbf{W}}_{\mathbf{y}}(\mathbf{x}))}{\exp(f^{\mathbf{W}}_{FG}(\mathbf{x})) + \exp(f^{\mathbf{W}}_{BG}(\mathbf{x}))}. \tag{2.3}$$

A scalar $\sigma$ is added in the classification likelihood function with the regression likelihood function.

$$p(\mathbf{y} \mid f^{\mathbf{W}}(\mathbf{x}), \sigma) = Softmax(\frac{1}{\sigma^2} f^{\mathbf{W}}(\mathbf{x})) = \frac{\exp(\frac{1}{\sigma^2} f^{\mathbf{W}}_{\mathbf{y}}(\mathbf{x}))}{\exp(\frac{1}{\sigma^2} f^{\mathbf{W}}_{FG}(\mathbf{x})) + \exp(\frac{1}{\sigma^2} f^{\mathbf{W}}_{BG}(\mathbf{x}))}. \tag{2.4}$$

Such adaptation would affect the output of classification output because it changes

the distribution to the Gibbs distribution, but the influence could be minimized by making $\sigma$ learnable. The log-likelihood for this output can then be written as:

$$\log p(\mathbf{y} \mid f^{\mathbf{W}}(\mathbf{x}), \sigma) = \frac{1}{\sigma^2} f_{\mathbf{y}}^{\mathbf{W}}(\mathbf{x}) - \log[\exp(\frac{1}{\sigma^2} f_{FG}^{\mathbf{W}}(\mathbf{x})) + \exp(\frac{1}{\sigma^2} f_{BG}^{\mathbf{W}}(\mathbf{x}))]. \quad (2.5)$$

In the case of multiple outputs, for example, a combination of a continuous output $\mathbf{y}_1$ and a discrete output $\mathbf{y}_2$, we will have a joint likelihood function. Let $f^{\mathbf{W}}(\mathbf{x})$ be a sufficient statistics, the joint likelihood function could be factorized:

$$p(\mathbf{y}_1, \mathbf{y}_2 \mid f^{\mathbf{W}}(\mathbf{x}), \sigma_1, \sigma_2) = p(\mathbf{y}_1 \mid f^{\mathbf{W}}(\mathbf{x}), \sigma_1, \sigma_2) \cdot p(\mathbf{y}_2 \mid f^{\mathbf{W}}(\mathbf{x}), \sigma_1, \sigma_2). \quad (2.6)$$

The joint loss in a form of log likelihood $\mathcal{L}(\mathbf{W}, \sigma_1, \sigma_2)$ is straight forward:

$$\begin{aligned}
\mathcal{L} &= -\log p\left(\mathbf{y}_1, \mathbf{y}_2 \mid f^{\mathbf{W}}(\mathbf{x}), \sigma_1, \sigma_2\right) \\
&= -\log \mathcal{N}\left(\mathbf{y}_1 \mid f^{\mathbf{W}}(\mathbf{x}), \sigma_1\right) \cdot p\left(\mathbf{y}_2 \mid f^{\mathbf{W}}(\mathbf{x}), \sigma_2\right) \\
&= \frac{1}{2\sigma_1^2} \left\|\mathbf{y}_1 - f^{\mathbf{W}}(\mathbf{x})\right\|^2 + \log \sigma_1 - \log Softmax\left(\mathbf{y}_2 \mid f^{\mathbf{W}}(\mathbf{x}), \sigma_2\right) \\
&= \frac{1}{2\sigma_1^2} \mathcal{L}_1(\mathbf{W}) + \frac{1}{\sigma_2^2} \mathcal{L}_2(\mathbf{W}) + \log \sigma_1 \\
&\quad + \log \frac{\exp(\frac{1}{\sigma_2^2} f_{FG}^{\mathbf{W}}(\mathbf{x})) + \exp(\frac{1}{\sigma_2^2} f_{BG}^{\mathbf{W}}(\mathbf{x}))}{[\exp(f_{FG}^{\mathbf{W}}(\mathbf{x})) + \exp(f_{BG}^{\mathbf{W}}(\mathbf{x}))]^{\frac{1}{\sigma_2^2}}} \\
&\approx \frac{1}{2\sigma_1^2} \mathcal{L}_1(\mathbf{W}) + \frac{1}{\sigma_2^2} \mathcal{L}_2(\mathbf{W}) + \log \sigma_1 + \log \sigma_2,
\end{aligned} \quad (2.7)$$

here we let $\mathcal{L}_1(\mathbf{W}) = \left\|\mathbf{y}_1 - f^{\mathbf{W}}(\mathbf{x})\right\|^2$, and $\mathcal{L}_1(\mathbf{W}) = -\log Softmax\left(\mathbf{y}_2 \mid f^{\mathbf{W}}(\mathbf{x}), \sigma_2\right)$. For simplicity, we make an approximation,

$$\frac{1}{\sigma_2}\left[\exp(\frac{1}{\sigma_2^2} f_{FG}^{\mathbf{W}}(\mathbf{x})) + \exp(\frac{1}{\sigma_2^2} f_{BG}^{\mathbf{W}}(\mathbf{x}))\right] \approx \left[\exp(f_{FG}^{\mathbf{W}}(\mathbf{x})) + \exp(f_{BG}^{\mathbf{W}}(\mathbf{x}))\right]^{\frac{1}{\sigma_2^2}}. \quad (2.8)$$

Though such approximation is oversimplified and loose, the disadvantage could be compensated by a learnable $\sigma_2$. According to this loss function, it would be possible to optimize the multi-task model with respect to the network on the parameters $\mathbf{W}$ and task weights $\sigma_1$ and $\sigma_2$.

## 2.5 Salient Object Detection

Salient object detection aims to localize the most attention-grabbing FG object in an image and then make pixel-wise classification to extract the FG segments (Itti et al. 1998; Pang et al. 2020; Qin et al. 2019; Zhao et al. 2019). Since the matting problem is extremely difficult to solve, we believe the prior knowledge of object saliency could be beneficial in fixing the fraud of falsely localizing the target FG object when it is placed remotely or among other objects. Therefore, we pretrain our network on a salient object detection dataset (DUTS). In the dataset, there are some images in which the salient object appears small because objects are located in a relatively remote position. Also, in other images, the salient object is placed among distracting objects.

# Chapter 3

# Proposed Method

## 3.1 Overall structure

Our model could be divided into three parts, the encoder, the entangled decoder, and the refinement module. For the encoder, we use a ResNet (He et al. 2015) to extract the feature of the input image. Based on the extracted feature information, segmenting and matting maps are decoded interactively in the entangled structure. Finally, we will put our coarse alpha matte prediction into a Unet-like (Ronneberger et al. 2015) refinement network for further improvement. All sub-outputs (e.g. segmentation outputs, coarse alpha matte predictions, and refined alpha matte prediction) of the model are supervised by our hybrid loss. Since the matting problem is highly ill-posed, we pretrain our model on a salient object detection dataset. The overall structure of our model is shown in Fig. 3.1, more details will be described in the following sections.

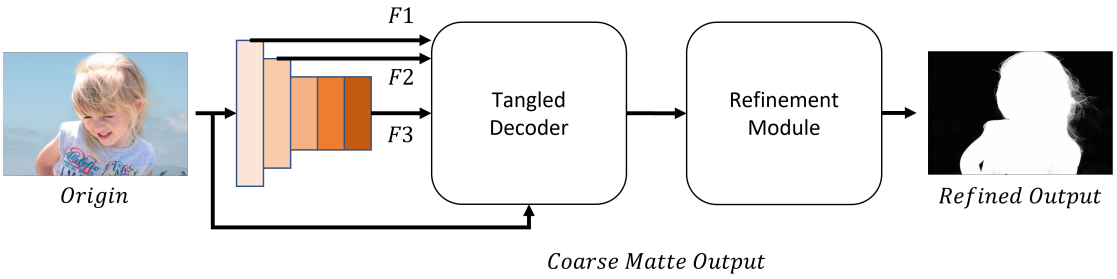



FIGURE 3.1: Overall network structure.

## 3.2 Network Modules

### 3.2.1 Encoder

The representational power of the feature extracted by ResNet encoder is promising since such deep residual architectures are widely studied and adopted by most matting algorithms (Lu et al. 2019; Hou and Liu 2019; Li and Lu 2020; Chen et al. 2018; Zhang et al. 2019; Qiao et al. 2020). As shown in Fig. 3.2, we adopt a modified version of ResNet-50 that integrates SE blocks into all stages of residual blocks as suggested in (Hu et al. 2018). Specifically, there are 50 layers in the encoder network, includes the starting convolutional layer (kernel size $7 \times 7$), the normal convolutional layer (kernel size $3 \times 3$), the bottleneck convolutional layer (kernel size $1 \times 1$), the downsampling convolutional layer (kernel size $3 \times 3$, stride 2) and the fully connected layer. All layers are followed by batch normalization and ReLU activation. With SE blocks added to each residual block, the representational power is further strengthened by adaptively recalibrating channel-wise feature attention maps at each layer. The difference between normal residual block and SE residual block is shown in Fig. 3.2. The encoder module transforms input images into downsampled feature maps in different sizes, and each feature map

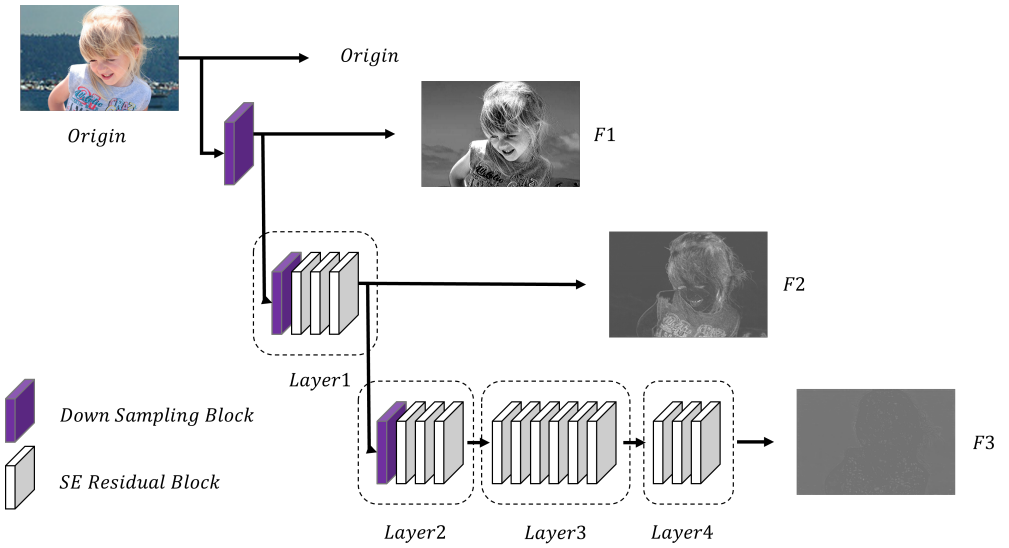
size stands for a sampling level. We will keep all feature maps and pass them into the decoder module.

### 3.2.2 Entangled Decoder

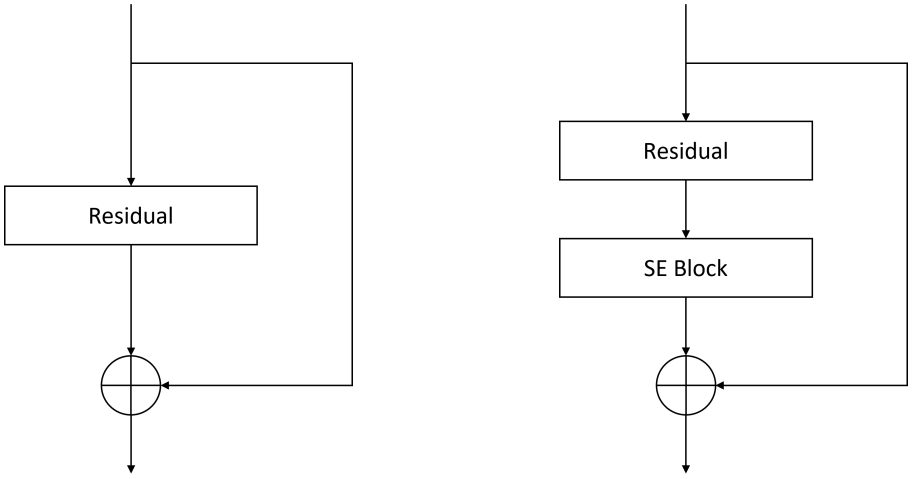
Matting natural images without trimaps is an extremely challenging problem, since the algorithm needs to find the location of a foreground object while extracting fine details from the same object. To pinpoint the target object, most of the trimap-free methods have a two-stage model, the first stage will explicitly generate trimaps and the second stage includes the normal matting model which requires trimap input. Different from them, we embed the trimap generation logic into the decoder. More specifically, we designed an entangled decoder in which feature output from the encoder will be fed into segmentation decoders and matting decoders. Outputs from every decoder would be supervised by the corresponding segmentation and matting ground truth, and there is a multitask loss which regulates both decoders at the end. To reinforce the feature exchange, a task switcher is adopted to create a communication path between the segmentation and the matting decoder at every sampling level. Fig. 3.3 shows the structure of our entangled decoder.

### 3.2.3 Basic Block

The basic block serves as a strong feature processor in the segmentation decoder, the matting decoder, and the task switcher. Inspired by the residual structure from (Zhang et al. 2018a), we use residual dense blocks (RDBs) followed by channel and spatial attention blocks as our main processing units. Inside the RDB, each layer is directly connected to the previous layers, leading to a dense feature input into




(a) Encoder




(b) Original Residual Block

(c) SE Residual Block

FIGURE 3.2: The detail structure of the SE-ResNet encoder (a). The schema of the original residual block (b) and the SE residual block(c).

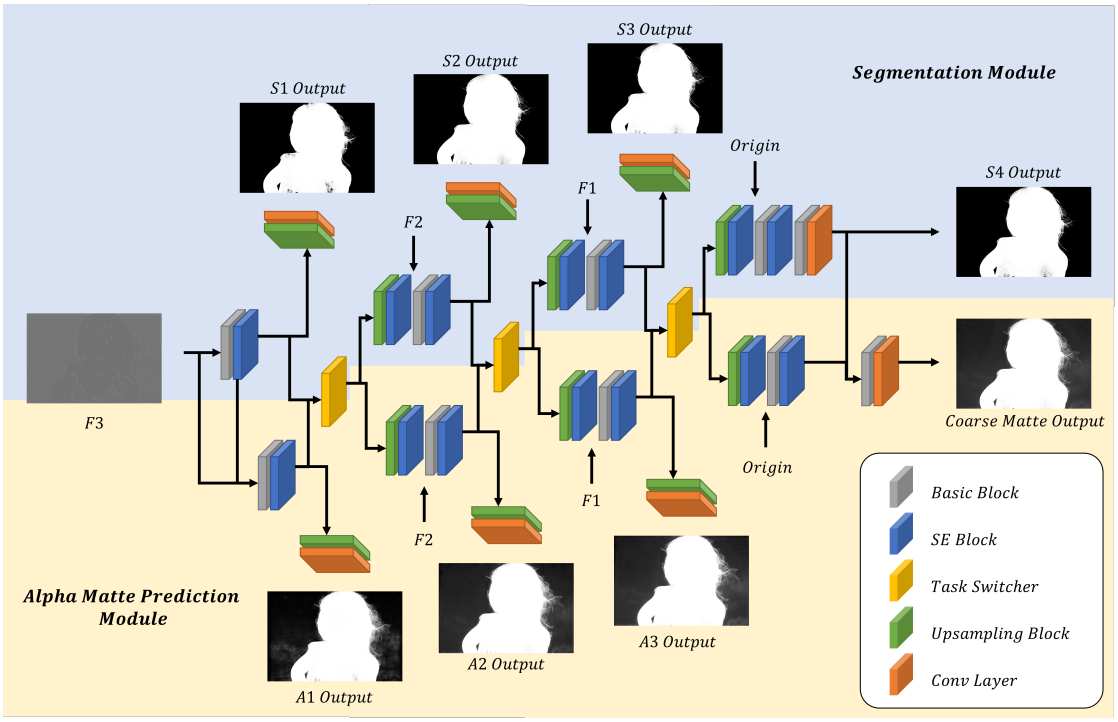



FIGURE 3.3: The detail structure of the entangled decoder.

the current layer. After dense layers, we use concatenation and $1 \times 1$ convolutional layer to fuse the local dense feature and then sum the fused dense feature and the input feature to give a residual feature output. With such architecture, connections are further established between RDBs, which means the current layer is able to leverage the information extracted from every other layer even if they are in other preceding RDBs.

### 3.2.4 Task Switcher

In image matting, alpha matte can be treated as an extension of binary segmentation map as it contains alpha values along with the solid FG information. Thus, we use the task switch module to exploit the correlation between binary segmentation

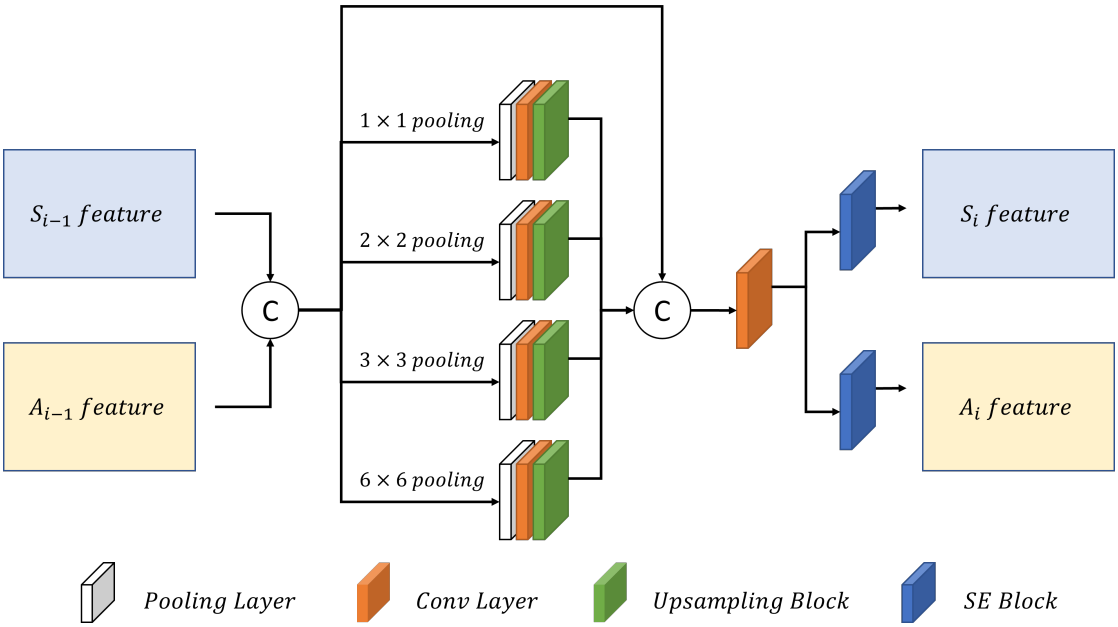



FIGURE 3.4: The Task Switcher.

and alpha prediction to improve the alpha matte quality. As shown in Fig. 3.4, inside the task switcher, we first use the pyramid pooling operation to construct and aggregate the features under different kernel sizes, since local features from small receptive fields are useful in generating an accurate alpha matte while perceptual information from larger fields is beneficial in classification. Later, the decoded information will be branched into segmentation steam and matting steam where spatial and channel-wise attentions are added to locate the focus of the feature maps for each steam.

### 3.2.5 Refinement Module

Although we could obtain a coarse alpha matte prediction from the encoder-decoder structure mentioned above, such prediction is far from satisfactory since it may have holes and an over-smoothed boundary. Therefore, we attach a UNet

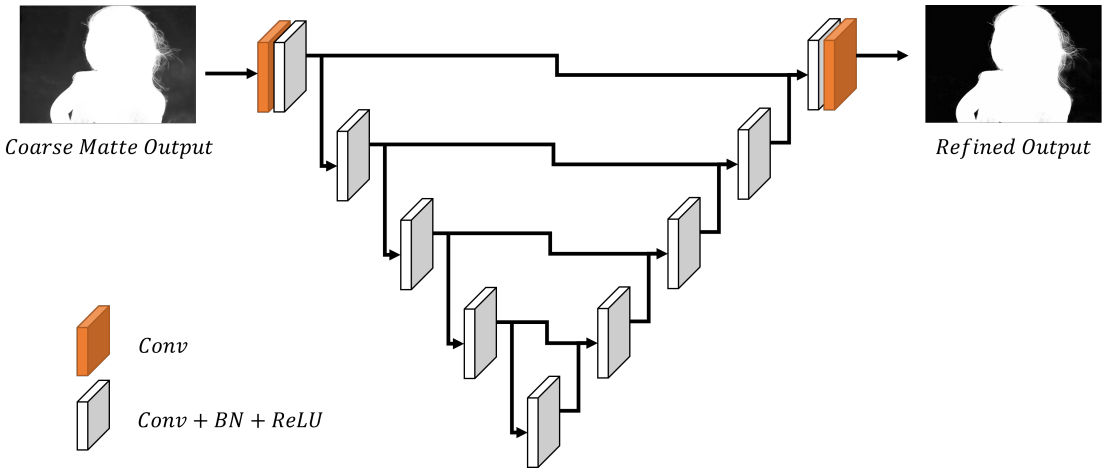



FIGURE 3.5: The detail structure of the refinement module.

as a residual refinement module at the end of our network to enhance the result. The refinement module is effective, and refined alpha matte predictions usually have filled holes and sharpened edges when compared to the direct outcome of the encoder-decoder network. During the training stage, the refinement module is jointly trained with all other modules. However, strong supervisions are added to both coarse and refined opacity prediction with respect to the alpha matte ground truth, so that we could restrict the refinement module to make small-scale changes to the coarse output. Details of our refinement module are illustrated in Fig. 3.5, and further discussions on the refinement module are in the ablation study section.

## 3.3 Loss function and Pretraining Strategy

Since our network has an entangled structure in which the binary segmentation decoder and the alpha matte decoder interact with each other, we designed a loss function to supervise each task depending on the distribution of that task, and then combine losses from all sampling levels to form a hybrid loss.

For the segmentation task, we adopt the loss combination from (Qin et al. 2019) due to its promising performance on the salient object detection task. The segmentation loss $l^{seg}$ is defined as:

$$\mathcal{L}^{seg} = \ell^{bce} + \ell^{ssim} + \ell^{iou} \tag{3.1}$$

where $\ell^{bce}$, $\ell^{ssim}$, and $\ell^{iou}$ indicate the Binary Cross-Entropy (BCE) loss (De Boer et al. 2005), the Structural Similarity (SSIM) loss (Wang et al. 2003), and the Intersection over Union (IoU) loss (Máttyus et al. 2017) respectively.

The BCE loss is the most popular target function among binary classification and segmentation tasks. In our work, it could be formulated as:

$$\ell^{bce} = \frac{1}{n}\sum_{i=1}^{n}[S_i \cdot log(\hat{S}_i) + (1 - S_i) \cdot log(1 - \hat{S}_i)] \tag{3.2}$$

where $\hat{S}_i$ is the predicted probability that pixel $i$ is a foreground pixel, and $S_i$ denotes the corresponding ground truth.

Generally, the SSIM loss is a measurement frequently used in low-level computer vision tasks such as super-resolution, dehaze, and derain. The SSIM uses the pixel correlation between ground truth and prediction to represent the similarity in structure. Assume there is a patch of predicted segmentation mask $\mathbf{x} = \{x_i \mid i = 1, 2, 3, ..., n\}$, and its corresponding ground-truth $\mathbf{y} = \{y_i \mid i = 1, 2, 3, ..., n\}$, the SSIM and the SSIM loss are defined as follow:

$$SSIM(\mathbf{x}, \mathbf{y}) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \tag{3.3}$$

$$\ell^{ssim} = 1 - SSIM(\mathbf{x}, \mathbf{y}), \tag{3.4}$$

where $\mu_x$, $\mu_y$ are means, $\sigma_x$, $\sigma_y$ are standard deviations, and $\sigma_{xy}$ is the covariance between the prediction and the ground truth. $C_1$ and $C_2$ are small values (e.g. 0.0001 and 0.0003) added to avoid dividing by zero. Note that the SSIM is 1 when the prediction is exactly the same as the ground truth (covariance equals self variance), and simple shifting or rotation could also lead to a high value.

The IoU loss is introduced to supervise the foreground segmentation task at an image level since it is usually utilized in high-level computer vision tasks such as object detection and segmentation:

$$\ell^{iou} = 1 - \frac{\sum_{i=1}^{n} S_i \cdot \hat{S}_i}{\sum_{i=1}^{n} (S_i + \hat{S}_i - S_i \cdot \hat{S}_i)}, \tag{3.5}$$

where $\hat{S}_i$ is the prediction, and $S_i$ denotes the corresponding ground truth at pixel $i$. Our hybrid loss $\mathcal{L}^{seg}$ is a comprehensive target function for the task, since it carries out supervision on the pixel, patch, and image-level (Qin et al. 2019).

For the alpha matte prediction task, the matting loss $\mathcal{L}^{m}$ contains two parts, the alpha-prediction loss $\ell^{\alpha}$ and the composition loss $\ell^{comp}$:

$$\mathcal{L}^{m} = \ell^{\alpha} + \ell^{comp}. \tag{3.6}$$

The alpha-prediction loss proposed in (Xu et al. 2017) uses $\sqrt{(\alpha_i - \hat{\alpha}_i)^2 + \epsilon^2}$ to replace the absolute difference which is non-differentiable:

$$\ell^{\alpha} = \sum_{i=1}^{n} \sqrt{(\alpha_i - \hat{\alpha}_i)^2 + \epsilon^2}, \tag{3.7}$$

where $\hat{\alpha}_i$ is the predicted alpha value at pixel $i$, and $\alpha_i$ is the ground truth value at the same location. With $\epsilon$ set a fixed value, the above equation is differentiable, and its derivative with respect to $\hat{\alpha}_i$ could be explicitly written as:

$$\frac{\partial \ell^{\alpha}}{\partial \hat{\alpha}_i} = \sum_{i=1}^{n} \frac{\alpha_i - \hat{\alpha}_i}{\sqrt{(\alpha_i - \hat{\alpha}_i)^2 + \epsilon^2}}. \tag{3.8}$$

Let $F$ be a foreground image associated with the ground truth alpha matte $\alpha$ and the predicted alpha matte $\hat{\alpha}$. According to equation 1.1, we could add the foreground image onto a random background image $B$ using ground truth and predicted alpha matte:

$$c = \alpha \odot F + (1 - \alpha) \odot B, \tag{3.9}$$

$$\hat{c} = \hat{\alpha} \odot F + (1 - \hat{\alpha}) \odot B, \tag{3.10}$$

where $\odot$ denotes the pixel-wise multiple calculations. Then the composition loss could be obtained by computing the absolute difference between $c$ and $\hat{c}$:

$$\ell^{comp} = \sum_{i=1}^{n} \|c_i - \hat{c}_i\|. \tag{3.11}$$

Since our network have an entangled architecture, we carry out supervision on different downsampling level $j$ (different feature map scale), and thus the total loss is defined as follow:

$$\mathcal{L}^{Total} = \sum_{j=1}^{k} \mathcal{L}_j^{seg} + \sum_{j=1}^{k} \mathcal{L}_j^{m}. \tag{3.12}$$

To further balance the two tasks, we use the strategy proposed in (Kendall et al. 2018), and derive a specific loss function for our segmentation and matting combination in Chapter 2. Our loss function will supervise the network in a multi-scale way. Suppose we have $k$ scales of outputs, the final loss function is defined as:

$$\mathcal{L}(\sigma_1, \sigma_2) = \sum_{j=1}^{k} \left[ \frac{1}{2\sigma_1^2} \mathcal{L}_j^{seg} + \frac{1}{\sigma_2^2} \mathcal{L}_j^{m} + \log \sigma_1 + \log \sigma_2 \right] \tag{3.13}$$

where $\sigma_1$ and $\sigma_2$ are learnable task priorities which weights the importance of each task. Here, we make a simplification that all levels of the network share one pair of $\sigma_1$ and $\sigma_2$. To avoid division by zero, we define $\gamma = log(\sigma^2)$ so that the total loss can be revised as:

$$\mathcal{L}(\gamma_1, \gamma_2) = \sum_{j=1}^{k} \left[ \frac{1}{2} exp(-\gamma_1) \mathcal{L}_j^{seg} + exp(-\gamma_2) \mathcal{L}_j^{m} + \frac{1}{2}\gamma_1 + \frac{1}{2}\gamma_2 \right]. \tag{3.14}$$

Such loss function could balance weights of tasks, and thus accelerate the training process.

To further address the problem of how to locate the $FG$ object, we let the network acquire some prior knowledge before proceeding to the matting task by pre-training our network on the DUTS dataset (Wang et al. 2017). This pre-training enables the network to identify the salient objects and further facilitates

the alpha prediction task in the later stage. More implementation details could be found in the subsequent section.

# Chapter 4

# Implementation and Experimental Results

## 4.1 Implementation details

Our model is trained on three Nvidia GTX 1080 Ti GPUs using PyTorch 1.2.0 deep learning framework.

In the pretraining stage, the ImageNet pretrained weight is loaded into our backbone encoder module to initiate the training. Then, we train our network on DUTS salient object detection dataset where input images are resized to $256 \times 256$ and then randomly cropped to $224 \times 224$. During the pre-training stage, 50k iterations of training are carried out.

In the formal training stage, our network is trained on the Adobe Composition-1k dataset, and input images would be directly resized or randomly cropped. Our random cropping process begins by arbitrarily selecting a cropping window size

from $320 \times 320$, $480 \times 480$, $640 \times 640$, $720 \times 720$, and $940 \times 940$ as smaller patches reveal more fine details and larger ones have more semantic information. The window would be located along the unknown regions(semitransparent areas), and we resize all cropped images to $320 \times 320$. During training process, we train our network for 100 epochs.

To further avoid overfitting, both stages use vertical, horizontal, and diagonal flipping as an additional augmentation strategy. Also, we apply Adam optimizer (Kingma and Ba 2017) with $\beta_1 = 0.9, \beta_2 = 0.999$, and polynomial learning rate decay, where $lr = lr_{start} \times (1 - \frac{iter}{totaliter})^p$ (starting learning rate $lr_{start}$ and decay rate $p$ are set to $1e^{-4}$ and 0.9, respectively in our case).

## 4.2 Datasets

### 4.2.1 DUTS dataset

The DUTS dataset is adopted in our pretraining stage. It is a salient object detection dataset with 10,553 training images and 5,019 test images, and most of its images are collected from the ImageNet dataset.

### 4.2.2 Adobe Composition-1k dataset

The Adobe Composition-1k Dataset from (Xu et al. 2017) is adopted in our main training stage. The training dataset consists of 431 foreground images and their corresponding alpha mattes ground truth. We combine each foreground image with 100 unique randomly selected background images from the MS COCO dataset (Lin et al. 2014), and thus we have 43100 different training images in total. As for

testing, another 50 foreground images and their corresponding alpha mattes would be given. Then we combine 20 arbitrary background images from the PASCAL VOC dataset (Everingham et al. 2010) with each foreground image to form a test set with 1000 images.

## 4.3 Evaluation metrics

Following (Rhemann et al. 2009), we use four quantitative metrics to evaluate the performance of our matting algorithm: the Mean Squared Error (MSE), the Sum of Absolute Differences (SAD), the Gradient Error (Grad), and the Connectivity Error (Conn).

In deep learning and computer vision, the **Mean Squared Error (MSE)** is the most popular metric, and it is defined as follow:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\alpha_i - \hat{\alpha}_i)^2 \tag{4.1}$$

where $\alpha_i$ and $\hat{\alpha}_i$ are the ground truth and the predicted alpha matte value at pixel $i$ respectively. Statistically, the MSE represents the variance of the unbiased estimator, meaning it represents the quality of the estimation. Note that the MSE is the average error among all pixel points, thus is not dependent on the size of the output.

Another frequently used metric is the **Sum of Absolute Differences (SAD)**, which calculates the accumulated absolute difference between the ground truth and

the prediction:

$$SAD = \sum_{i=1}^{n} |\alpha_i - \hat{\alpha}_i|. \tag{4.2}$$

Note that the SAD value is related to the size of the prediction (e.g. larger output image would usually have a larger SAD value), and thus SAD values are comparable only if they are obtained under the same test condition.

Though MSE and SAD are widely used in pixel-level performance evaluation, they are not always consistent with human perceptions (e.g. over smoothing and big inconsistency in a small region), and thus the gradient and the connectivity are introduced to establish perceptual evaluations (Rhemann et al. 2009).

To avoid over smoothing, we introduce the **Gradient Error (Grad)** metric which could be formulated as follow:

$$Grad = \sum_{i=1}^{n} (\nabla\alpha_i - \nabla\hat{\alpha}_i)^2 \tag{4.3}$$

where $\nabla$ denotes the Gaussian gradient operator, and the gradient error is the sum of squared gradient difference. Notice that the gradient error is also a measurement dependent on the size of the prediction.

On the other hand, the **Connectivity Error (Conn)** is adopted to reflect the small sudden pixel-level inconsistencies which are perceptually catastrophic. First, we need to define the connectivity. Let $\varphi(\alpha_i, \Omega)$ denotes the degree of connectivity at pixel $i$ to a source region $\Omega$. Here we define the largest area, which is composed of the absolute foreground pixels (with alpha value 1), as the source region. Assume there is a maximum threshold $l_i$ at pixel $i$ which determines whether the pixel is

connected to the source region, and the threshold is no larger than the alpha value $\alpha_i$ at pixel $i$.

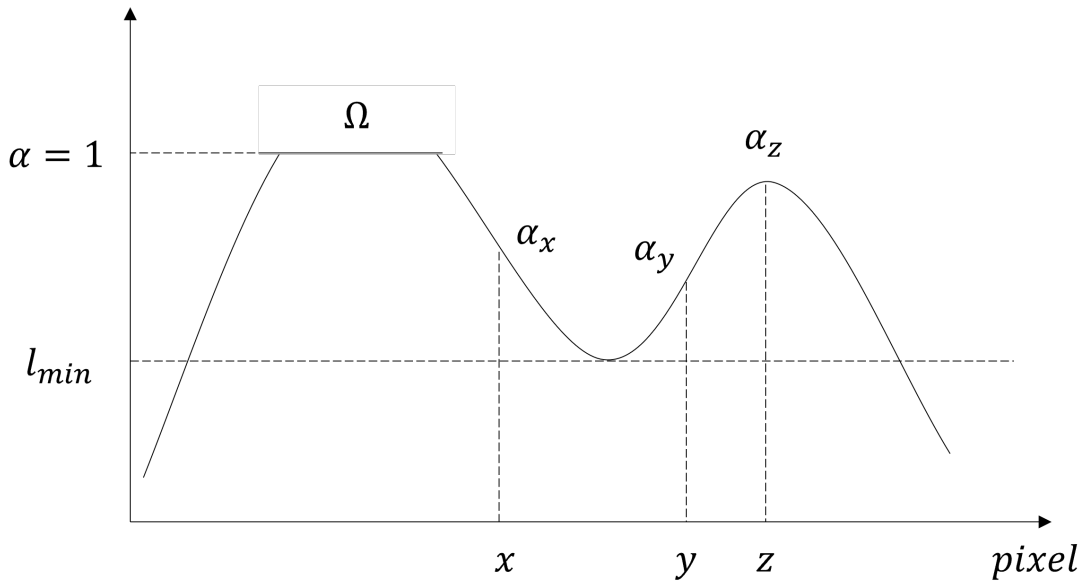



FIGURE 4.1: Connectivity

For example, in Fig. 4.1, there is one row of pixels from a picture. Starting from the source region $\Omega$ and going right, we update the threshold $l_i$ with the minimum alpha value that we passed. Therefore, we will have $\alpha_x = l_x$ at pixel $x$ and a full connection from the pixel to the source region. Otherwise, like in pixel $y$ and $z$, the threshold $l_y = l_z = l_{min}$, and we will have $\alpha_y > l_y$ and $\alpha_z > l_z$. In these cases, those pixels are not connected to the source region, and the connectivity decreases when the gap between $\alpha_i$ and $l_i$ widens. Thus we could formulate the degree of connectivity as:

$$\varphi(\alpha_i, \Omega) = 1 - [\lambda_i \cdot \delta(\alpha_i - l_i \geq \theta) \cdot (\alpha_i - l_i)]. \tag{4.4}$$

There is a full connection when $\varphi = 1$, and a completely disconnection when $\varphi = 0$.

The measurement becomes robust to small variants with the introduction of the $\delta$ function, and distance-related scalar $\lambda_i$ is added base on the intuition that distant disconnection is more distracting visually. As the connectivity is defined, we could easily compute the connectivity error as:

$$Conn = \sum_{i=1}^{n} [\varphi(\alpha_i, \Omega) - \varphi(\hat{\alpha}_i, \Omega)]^2. \tag{4.5}$$

Note that all four metrics are computed among transition areas indicated by the trimap. Therefore, even though our method is trimap-free, we would still generate the trimap for calculating the above metrics along unknown region.

## 4.4 Qualitative and quantitative results

**Adobe Composition-1k dataset.** Since trimaps are commonly used in matting algorithms and they often provide a more accurate solution, six trimap-based matting algorithms are selected for comparison, namely, KNN (Chen et al. 2013), Closed (Levin et al. 2007), DIM (Xu et al. 2017), IndexNet (Lu et al. 2019), GCA (Li and Lu 2020), CA (Hou and Liu 2019). Also, we compare our SegMatting with five trimap-free methods (SHM (Chen et al. 2018), LF (Zhang et al. 2019), BGM (Sengupta et al. 2020), Boosting (Liu et al. 2020), HAtt (Qiao et al. 2020)). More precisely, SHM, LF, Boosting, HAtt, and our method only need RGB inputs; BGM requires the BG image as an additional input; and others need both RGB and trimap images, produced according to the procedure described in (Xu et al. 2017). Note that Boosting, BGM, and SHM are designed specifically for human

| Methods | MSE ($10^{-2}$) | SAD | Grad | Conn |
|---|---|---|---|---|
| KNN (Chen et al. 2013) | 9.44 | 170.3 | 128.4 | 67.5 |
| Closed (Levin et al. 2007) | 7.66 | 150.6 | 111.2 | 61.4 |
| BGM (Sengupta et al. 2020) | 2.17 | 56.8 | 74.8 | 56.2 |
| DIM (Xu et al. 2017) | 1.31 | 47.1 | 25.4 | 46.1 |
| IndexNet (Lu et al. 2019) | 1.05 | 39.5 | 17.3 | 36.2 |
| GCA (Li and Lu 2020) | 0.79 | 32.9 | 15.3 | 29.6 |
| CA (Hou and Liu 2019) | 0.67 | 32.6 | 13.9 | 28.9 |
| SHM† (Chen et al. 2018) | 5.38 | 102.8 | 56.1 | 111.6 |
| Boosting† (Liu et al. 2020) | 3.44 | 85.8 | 78.9 | 90.3 |
| Late Fusion† (Zhang et al. 2019) | 2.34 | 60.9 | 77.6 | 61.5 |
| **HAtt**† (Qiao et al. 2020) | **1.46** | **51.2** | **30.7** | **51.1** |
| Ours† | 2.19 | 63.1 | 31.7 | 63.7 |

TABLE 4.1: Numerical results on the Adobe Composition-1k dataset calculated on the unknown region indicated by the trimap. "†" indicates that the corresponding method takes only RGB images as inputs.

matting, but we still involve them in our experiments to compare the model effectiveness. We predict alpha mattes with full-scale image inputs and calculate the aforementioned four metrics along with the whole image as well as the unknown region indicated by the trimap.

From Table 4.1, we can see that the proposed SegMatting outperforms most trimap-free methods. Note that the red number denotes the best result for methods that use additional inputs while the bold number indicates the best result for methods that use only the RGB image. In the table, we only measure errors over the unknown region indicated by the trimap. Under this setting, our method is comparable to trimap-based algorithms considering our method does not rely on any user input and only makes use of the intrinsic image feature for alpha prediction. Also, our method is competitive among trimap-based methods shown in the same table. The qualitative results in Fig. 4.2 and Fig. 4.3 further prove

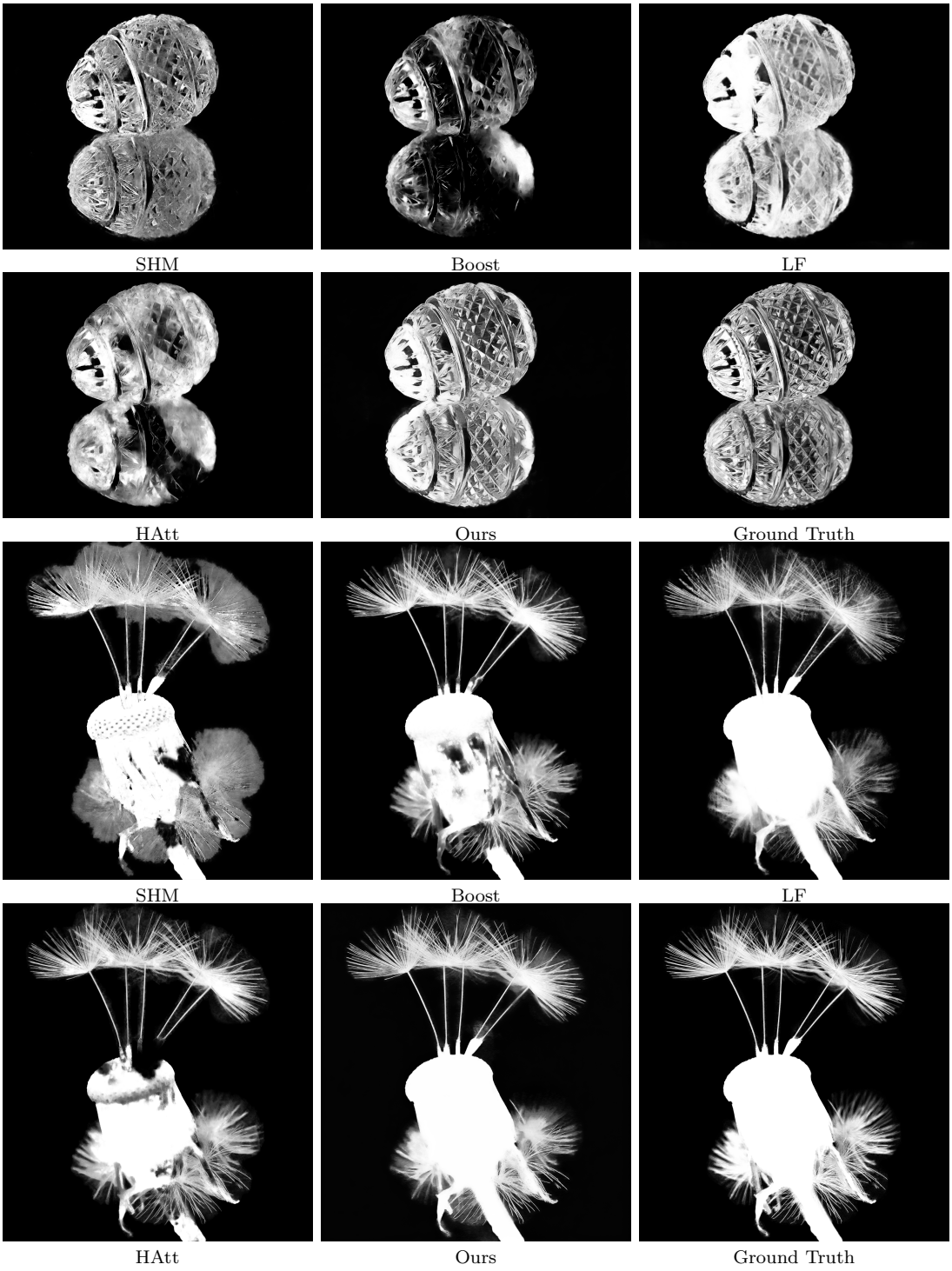



FIGURE 4.2: Qualitative results on the Adobe Composition-1k dataset. (1) Zoom-in for better views.

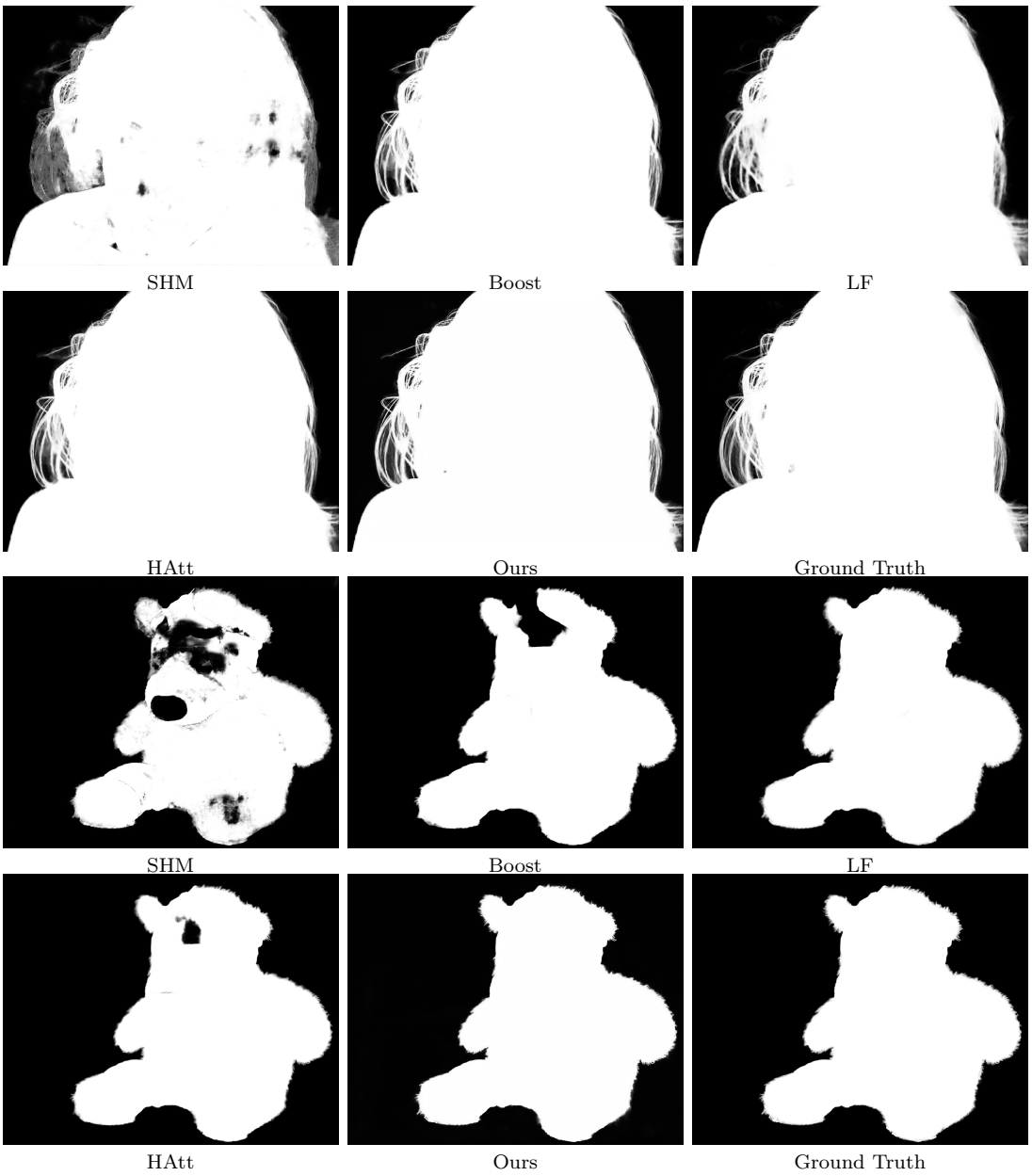



FIGURE 4.3: Qualitative results on the Adobe Composition-1k dataset.(2) Zoom-in for better views.

| Methods | MSE ($10^{-2}$) | SAD | Grad | Conn |
|---|---|---|---|---|
| Single decoder | 5.15 | 98.4 | 45.9 | 104.9 |
| No refinement module | 3.80 | 90.8 | 32.4 | 97.0 |
| Normal loss | 3.14 | 76.4 | 39.9 | 77.8 |
| No pretrain | 2.40 | 68.0 | 41.9 | 68.4 |
| Full model | 2.19 | 63.1 | 31.7 | 63.7 |

TABLE 4.2: Numerical results given by different model setup on the Adobe Composition-1k dataset.

the effectiveness of our method in localizing the foreground object. In Fig. 4.2, other matting methods are confused with foreground and background objects, but our method could locate the foreground object out of a complicated image.

**Real-world image matting.** Though deep learning algorithms perform very well on the toy dataset where the training set and the testing set have a similar distribution, their capabilities tend to deteriorate on real-world images. As shown in Fig. 4.4 and Fig. 4.5, we try to generate the alpha matte from several real-world images from the internet with the weights obtained from previous experiments. Even without any user input (trimaps and scribbles), our SegMatting is able to produce accurate alpha mattes for FG images with complex shapes. This implementation shows the robustness of the model under the real-world situation, and also re-demonstrates the superiority of our model when user inputs are unavailable. However, there are some artifacts in our alpha matte predictions. If the FG color coincides with the BG color (For example, white hair on the doglegs and the white floor in Fig. 4.5 row 1), our model may falsely detect the boundary of the FG object.

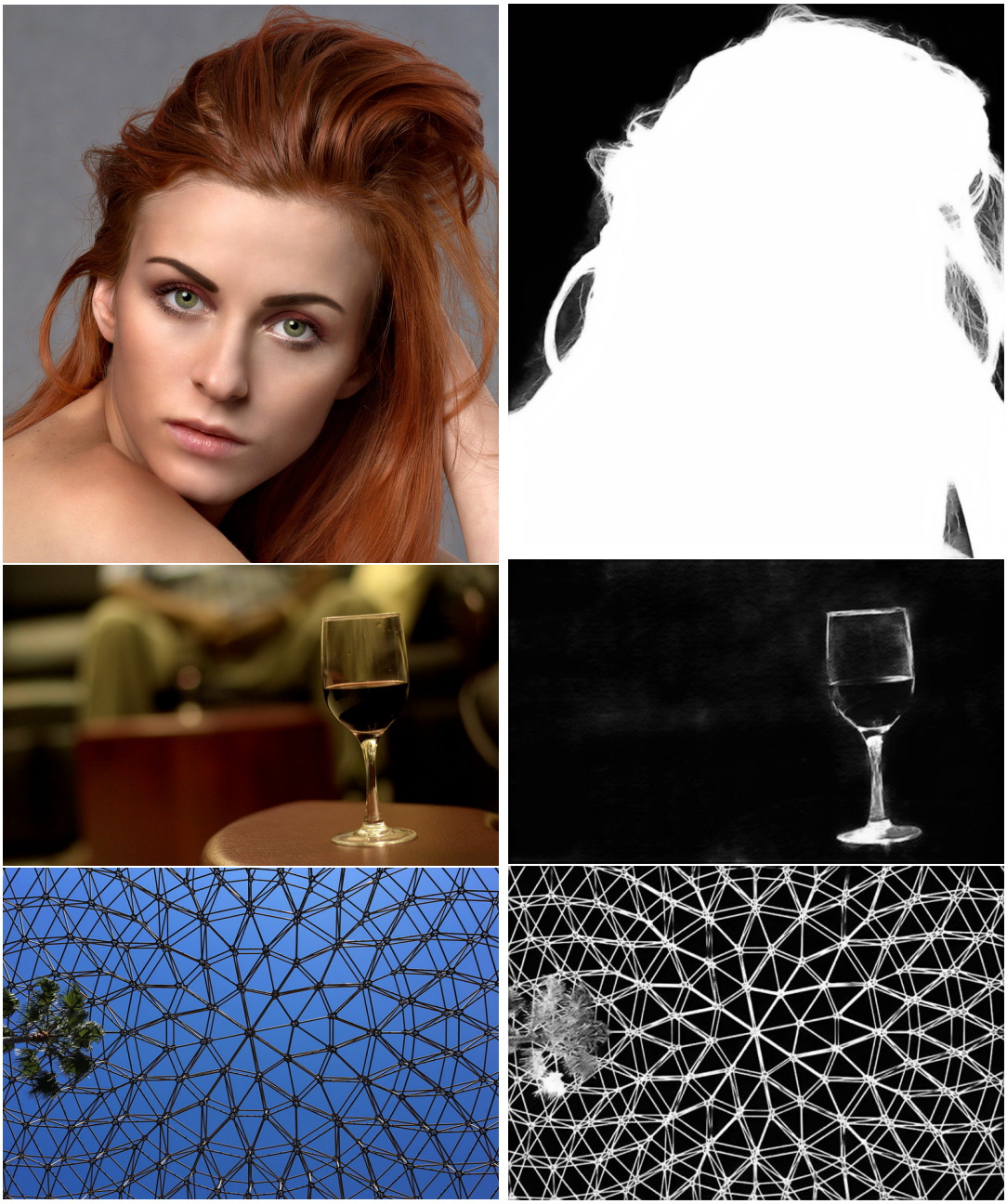



FIGURE 4.4: Results on real-world images.(1) Zoom-in for better views.

Input Predicted Alpha

FIGURE 4.5: Results on real-world images.(2) Zoom-in for better views.

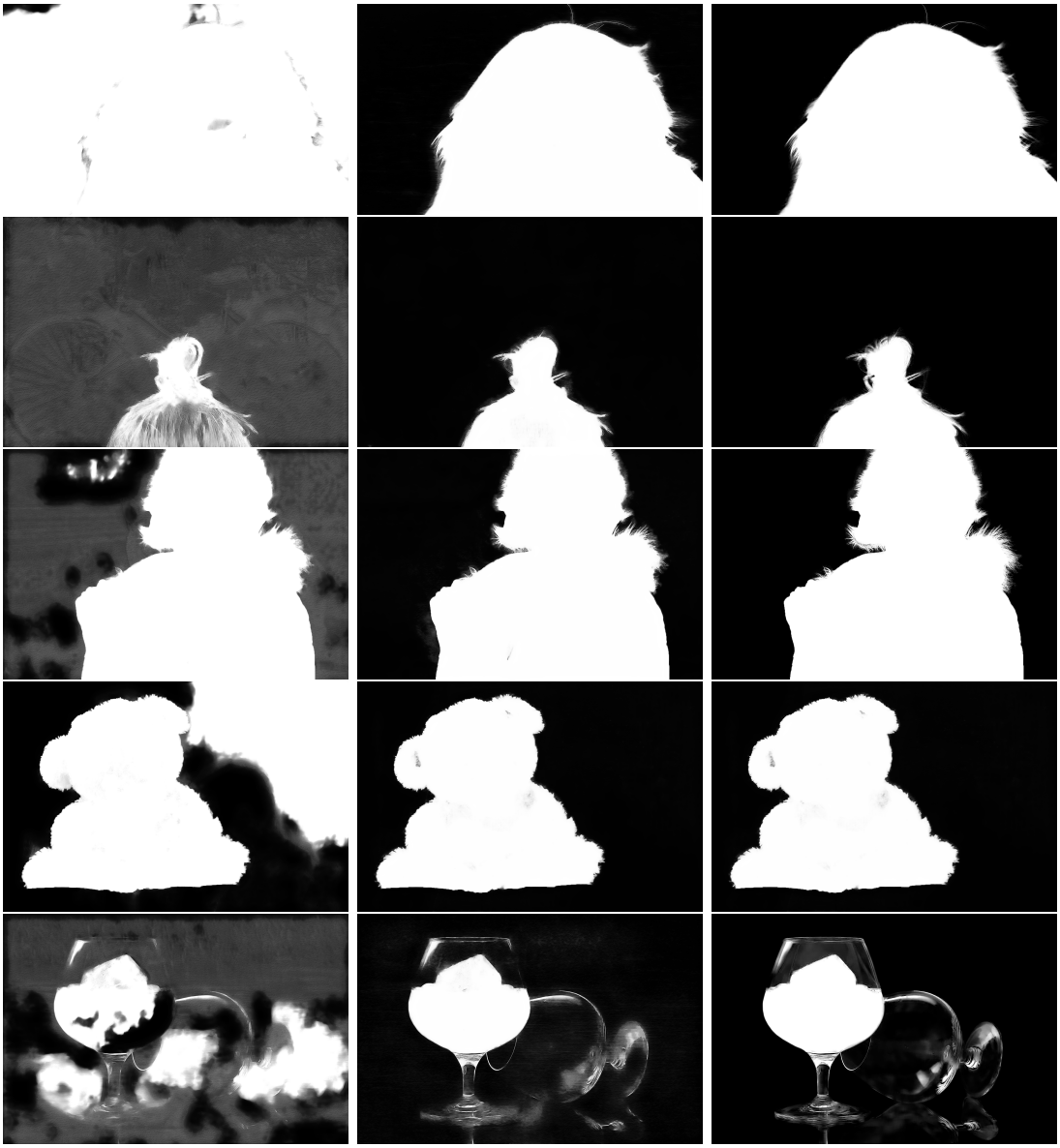
## 4.5 Ablation study

To further demonstrate the effectiveness of our design and training strategy, we test 4 other alternatives: single decoder, no refinement module, no pretrain, and normal loss. In the single decoder variant, we remove the task switcher and the information communication between two tasks; In the no refinement module version, we use the direct coarse outcome of the entangled decoder as final prediction without further refinement; During the no pretrain process, we skip the pretraining on the DUTS dataset; For the normal loss variant, we still supervise whole training process at all sampling levels, but we replace the dynamic weighting on different tasks with fixed parameter.

Results on the Composition-1k testing set are shown in Table 4.2. The entangled decoder is the most important design of our SegMatting since the performance of disentangled structure deteriorates by a large margin. Further qualitative illustrations are provided in Fig. 4.6 showing that the single decoder model is not able to distinguish FG objects from complicated BGs without the entangled decoder, and it also fails to clear up the BG remains.

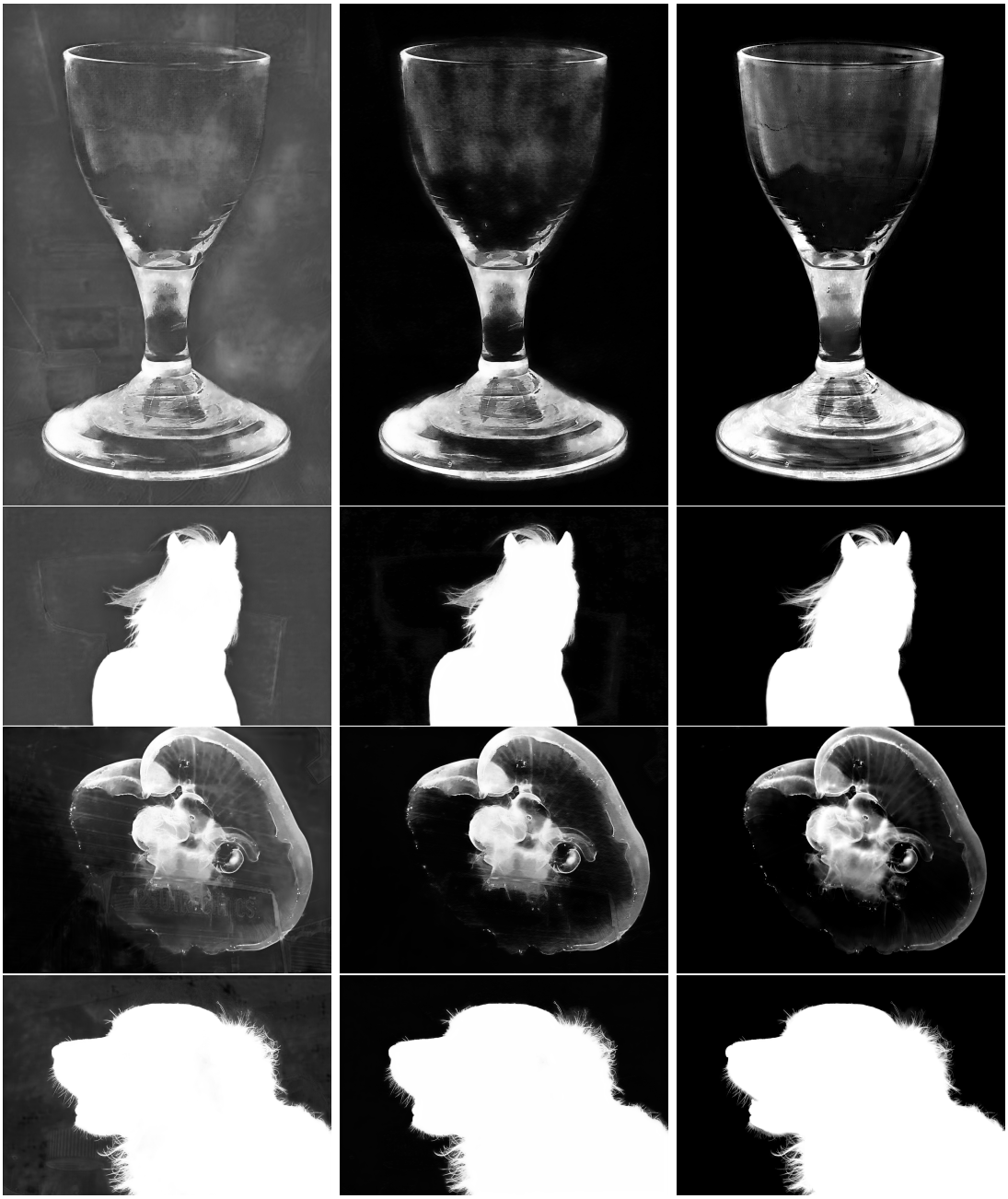
The refinement module does an excellent job of improving coarse alpha matte predictions, but does not reduce the gradient error significantly. As shown in Fig. 4.7, the refinement module only reduces the overall noise level, but it sharpen the boundary. Such modification doesn't affect the gradient error a lot, because the outline of foreground objects is already solidated in coarse predictions.

As shown in Table 4.2, both the pretraining on DUTS and the multitask loss are important training strategies to our model, since they both boost our model.



Disentangled Decoder | Entangled Decoder | Ground Truth

FIGURE 4.6: Disentangled decoder vs Entangled decoder. Zoom-in for better views.



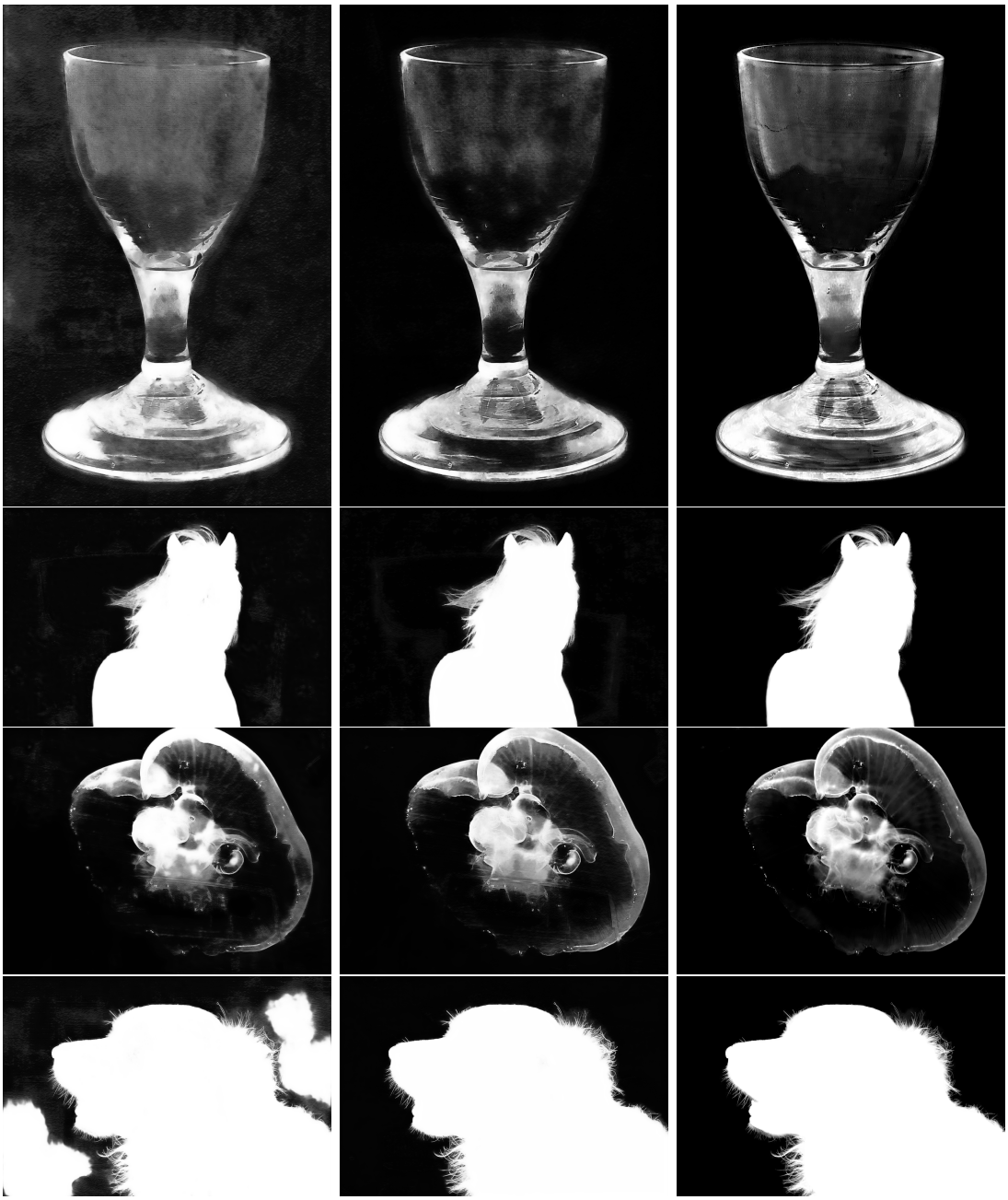
Coarse Prediction | Refined Prediction | Ground Truth

FIGURE 4.7: Refinement module enhance the coarse prediction. Zoom-in for better views.


Pretraining Loss

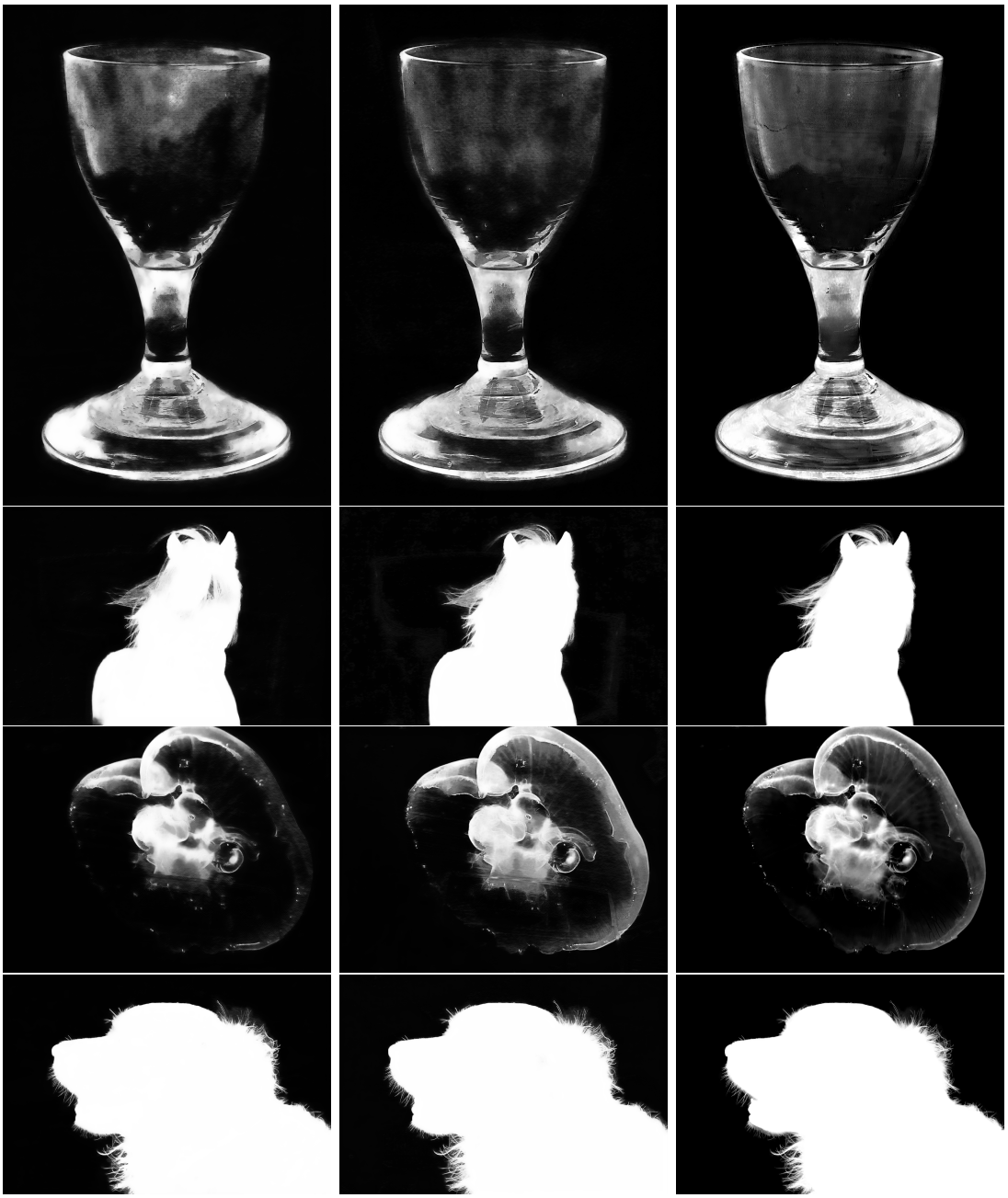
FIGURE 4.8: Convergence of different variant.

In Fig. 4.8, we could observe that pretraining on DUTS will provide a better starting point since the prior knowledge on saliency is utilized. However, the improvement of the model is not very significant when the model converges. On the other hand, our model achieves a higher PSNR using the hybrid loss than using the normal loss, which demonstrates the importance of adaptively balancing multitasks. Qualitative results are shown in Fig. 4.9 and 4.10.



Normal Loss | Hybrid Loss | Ground Truth

FIGURE 4.9: Normal Loss vs Hybrid Loss. Zoom-in for better views.



No Pretraining Pretraining Ground Truth

FIGURE 4.10: No Pretraining vs Pretraining. Zoom-in for better views.

# Chapter 5

# Conclusion and Future Work

In this thesis, we have proposed a trimap-free natural image matting algorithm based on multitask learning. An entangled decoder is designed to combine binary segmentation with alpha matte prediction to eliminate trimap input. In addition, we implement a task switcher and a hybrid multitask loss to establish information flow between two tasks, and adaptively utilize the correlated feature information. To further improve the performance of our model, the channel-wise attention is added in the regular deep residual encoder network, and a U-shape refinement model is also attached to enhance the coarse opacity prediction. Moreover, we find out that prior knowledge on object saliency would be beneficial to our trimap-free image matting task since the task is challenging with limited information provided. Extensive experiments demonstrate that the proposed method outperforms most trimap-free matting algorithms and achieves comparable results against state-of-the-art trimap-based algorithms.

Although our model shows a decent performance, we could still identify some artifacts in our alpha matte prediction, for example, hollows in the supposed FG

object area and detail losses in the boundary region. In the frequency domain, the flat area of the definite FG locates in the low-frequency section while the rapidly changing boundary region has more information in the high-frequency section. Therefore, we expect that further progress can leverage the frequency information to guide the generation of alpha matte.

# Bibliography


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

Bahdanau, D., Cho, K., and Bengio, Y. (2016). *Neural Machine Translation by Jointly Learning to Align and Translate*.

Cai, S., Zhang, X., Fan, H., Huang, H., Liu, J., Liu, J., Liu, J., Wang, J., and Sun, J. (2019). Disentangled image matting. In: *Proceedings of the IEEE International Conference on Computer Vision*, 8819–8828.

Cao, Y., Xu, J., Lin, S., Wei, F., and Hu, H. (2019). Gcnet: Non-local networks meet squeeze-excitation networks and beyond. In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 0–0.

Caruana, R. (1997). Multitask learning. *Machine learning* 28(1), 41–75.

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: *Proceedings of the 26th ACM international conference on Multimedia*, 618–626.

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

Chuang, Y.-Y., Curless, B., Salesin, D. H., and Szeliski, R. (2001). A bayesian approach to digital matting. In: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*. Vol. 2. IEEE, II–II.

De Boer, P.-T., Kroese, D. P., Mannor, S., and Rubinstein, R. Y. (2005). A tutorial on the cross-entropy method. *Annals of operations research* 134(1), 19–67.

Eigen, D. and Fergus, R. (2015). *Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture*.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision* 88(2), 303–338.

Feng, X., Liang, X., and Zhang, Z. (2016). A cluster sampling method for image matting via sparse coding. In: *European Conference on Computer Vision*. Springer, 204–219.

Fu, J., Liu, J., Tian, H., Li, Y., Bao, Y., Fang, Z., and Lu, H. (2019). Dual attention network for scene segmentation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3146–3154.

Grady, L., Schiwietz, T., Aharon, S., and Westermann, R. (n.d.). Random walks for interactive alpha-matting. In:

He, K., Rhemann, C., Rother, C., Tang, X., and Sun, J. (2011). A global sampling method for alpha matting. In: *CVPR 2011*. IEEE, 2049–2056.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). *Deep Residual Learning for Image Recognition.*

Hou, Q. and Liu, F. (2019). Context-aware image matting for simultaneous foreground and alpha estimation. In: *Proceedings of the IEEE International Conference on Computer Vision*, 4130–4139.

Hu, J., Shen, L., Albanie, S., Sun, G., and Vedaldi, A. (2019). *Gather-Excite: Exploiting Feature Context in Convolutional Neural Networks.*

Hu, J., Shen, L., and Sun, G. (2018). Squeeze-and-excitation networks. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7132–7141.

Huang, Z., Wang, X., Huang, L., Huang, C., Wei, Y., and Liu, W. (2019). Ccnet: Criss-cross attention for semantic segmentation. In: *Proceedings of the IEEE International Conference on Computer Vision*, 603–612.

Itti, L., Koch, C., and Niebur, E. (1998). A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20(11), 1254–1259.

Kendall, A., Gal, Y., and Cipolla, R. (2018). Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7482–7491.

Kingma, D. P. and Ba, J. (2017). *Adam: A Method for Stochastic Optimization.*

Levin, A., Lischinski, D., and Weiss, Y. (2007). A closed-form solution to natural image matting. *IEEE transactions on pattern analysis and machine intelligence* 30(2), 228–242.

Li, Y. and Lu, H. (2020). Natural image matting via guided contextual attention. *arXiv preprint arXiv:2001.04069.*

Liao, Y., Kodagoda, S., Wang, Y., Shi, L., and Liu, Y. (2015). *Understand Scene Categories by Objects: A Semantic Regularized Scene Classifier Using Convolutional Neural Networks.*

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In: *European conference on computer vision*. Springer, 740–755.

Liu, J., Yao, Y., Hou, W., Cui, M., Xie, X., Zhang, C., and Hua, X.-s. (2020). Boosting Semantic Human Matting with Coarse Annotations. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8563–8572.

Lu, H., Dai, Y., Shen, C., and Xu, S. (2019). Indices matter: Learning to index for deep image matting. In: *Proceedings of the IEEE International Conference on Computer Vision*, 3266–3275.

Máttyus, G., Luo, W., and Urtasun, R. (2017). Deeproadmapper: Extracting road topology from aerial images. In: *Proceedings of the IEEE International Conference on Computer Vision*, 3438–3446.

Pang, Y., Zhao, X., Zhang, L., and Lu, H. (2020). *Multi-scale Interactive Network for Salient Object Detection.*

Qiao, Y., Liu, Y., Yang, X., Zhou, D., Xu, M., Zhang, Q., and Wei, X. (June 2020). Attention-Guided Hierarchical Structure Aggregation for Image Matting. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).*

Qin, X., Zhang, Z., Huang, C., Gao, C., Dehghan, M., and Jagersand, M. (2019). Basnet: Boundary-aware salient object detection. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7479–7489.

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*. IEEE, 1826–1833.

Ronneberger, O., Fischer, P., and Brox, T. (2015). *U-Net: Convolutional Networks for Biomedical Image Segmentation*.

Sengupta, S., Jayaram, V., Curless, B., Seitz, S. M., and Kemelmacher-Shlizerman, I. (2020). Background Matting: The World is Your Green Screen. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2291–2300.

Shahrian, E. and Rajan, D. (2012). Weighted color and texture sample selection for image matting. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 718–725.

Smith, A. R. and Blinn, J. F. (1996). Blue screen matting. In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 259–268.

Sun, J., Jia, J., Tang, C.-K., and Shum, H.-Y. (2004). Poisson matting. In: *ACM SIGGRAPH 2004 Papers*, 315–321.

Teichmann, M., Weber, M., Zoellner, M., Cipolla, R., and Urtasun, R. (2018). *MultiNet: Real-time Joint Semantic Reasoning for Autonomous Driving*.

Uhrig, J., Cordts, M., Franke, U., and Brox, T. (2016). *Pixel-level Encoding and Depth Layering for Instance-level Semantic Labeling*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In: *Advances in neural information processing systems*, 5998–6008.

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

Wang, L., Lu, H., Wang, Y., Feng, M., Wang, D., Yin, B., and Ruan, X. (2017). Learning to Detect Salient Objects with Image-level Supervision. In: *CVPR*.

Wang, X., Girshick, R., Gupta, A., and He, K. (2018). Non-local neural networks. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7794–7803.

Wang, Z., Simoncelli, E. P., and Bovik, A. C. (2003). Multiscale structural similarity for image quality assessment. In: *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*. Vol. 2. Ieee, 1398–1402.

Woo, S., Park, J., Lee, J.-Y., and Kweon, I. S. (2018). *CBAM: Convolutional Block Attention Module*.

Xu, N., Price, B., Cohen, S., and Huang, T. (2017). Deep image matting. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2970–2979.

Zhang, Y., Tian, Y., Kong, Y., Zhong, B., and Fu, Y. (2018a). Residual dense network for image super-resolution. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2472–2481.

Zhang, Y., Gong, L., Fan, L., Ren, P., Huang, Q., Bao, H., and Xu, W. (2019). A late fusion CNN for digital matting. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7469–7478.

Zhang, Z., Cui, Z., Xu, C., Jie, Z., Li, X., and Yang, J. (2018b). Joint task-recursive learning for semantic segmentation and depth estimation. In: *Proceedings of the European Conference on Computer Vision (ECCV)*, 235–251.

Zhao, J.-X., Liu, J., Fan, D.-P., Cao, Y., Yang, J., and Cheng, M.-M. (2019). *EGNet:Edge Guidance Network for Salient Object Detection.*