Robust High Resolution Video Matting

1. Introduction

Video matting is a technique for separating the video into two or more layers, usually foreground (F) and background (B), and generating alpha mattes (α) which determine blending of the layers.

$$I = \alpha F + (1 - \alpha)B \quad (1)$$

By extracting α and F, we can separate the foreground objects from the background and achieve the background replacement effect.

Some applications of background replacement are video conferencing and entertainment video creation, etc.

This project focuses on using the neural models to improve the matting quality and robustness for such applications.

Most existing methods are designed for video applications, processing individual frames as independent images. Those approaches neglect the most widely available feature in videos: temporal information. What is temporal information? Temporal is a time example, video consists of image frame sequence. With respect to time the frames are changed in the video. This is called temporal information. Reasons why we can improve the video matting performance are: First, it allows the prediction of more coherent results, as the model can see multiple frames and its own predictions. This significantly reduces flicker and improves perceptual quality; Second, temporal information can improve matting robustness. In the cases where an individual frame might be ambiguous, etc. the foreground color becomes similar to a passing object in the background, the model can better guess the boundary by referring to the previous frames; Third, temporal information allows the model to learn more about the background over time. When the camera moves, the background behind the subjects is revealed due to the perspective change. Even if the camera is fixed, the occluded background still often reveals due to the subject's movements. Having a better understanding of the background simplifies the matting task. Therefore, we can use the recurrent architecture to exploit the temporal information. This helps to improve the matting quality and temporal coherence. Apart from that, the training strategy is also a factor that enforces our model on both matting and semantic segmentation objectives simultaneously. Because the mating tasks are similar to human segmentation tasks, simultaneously training with a segmentation objective can effectively regulate our model without additional adaptation steps.

2. Model architecture

The model consists of 3 parts: an encoder, a recurrent decoder and a Deep guided filter module. The first part (encoder) is used to extract individual frame's features. The second part (recurrent decoder) is used to aggregate temporal information. The last one is used for high-resolution upsampling. The input is first downsampled for the encoder-decoder network, then DGF is used to upsample the result.

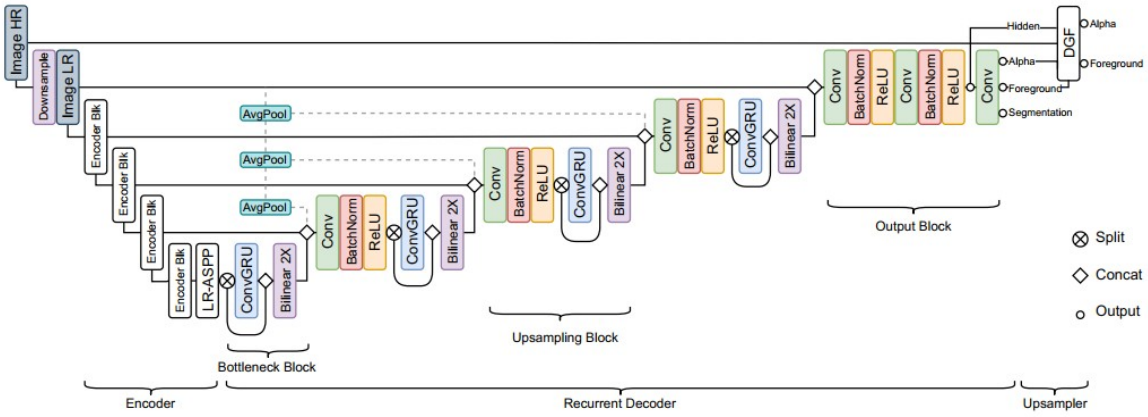

Figure 1. Model architecture

2.1. Feature-Extraction Encoder

In this first part, we use MobileNetV3-Large as the efficient backbone followed by the LR-ASPP module. The last block of MobileNetV3 used dilated- convolutions without downsampling stride. The encoder module operates on individual frames and extracts features at 1/2, 1/4, 1/8, 1/16 scales for recurrent decoder.

2.2. Recurrent Decoder

The reasons why we use recurrent architecture: First, it can learn what information to keep and forget by itself on a continuous stream of video. The ability to adaptively keep both long-term and short-term temporal information makes recurrent mechanisms more suitable for this task.

Our decoder adopts ConvGRU at multiple scales to aggregate temporal information. ConvGRU is defined as:

$$
\begin{aligned}
z_t &= \sigma(w_{zx} * x_t + w_{zh} * h_{t-1} + b_z) \\
r_t &= \sigma(w_{rx} * x_t + w_{rh} * h_{t-1} + b_r) \\
o_t &= tanh(w_{ox} * x_t + w_{oh} * (r_t \odot h_{t-1}) + b_o) \\
h_t &= z_t \odot h_{t-1} + (1 - z_t) \odot o_t
\end{aligned}
\tag{2}
$$

As we can see on the figure of our model architecture above, our decoder consists of 3 parts: A bottleneck block, upsampling block and output block.

- Bottleneck block: it operates at the 1/16 feature scale after the LR-ASPP module.
- Upsampling block: it is repeated at 1/8, 1/4, and 1/2 scale. First, it concatenates the bilinearly upsampled output from the previous block, the feature map of the corresponding scale from the encoder, and the input image downsampled by repeated 2 x 2 average pooling. Then, a convolution followed by Batch Normalization and ReLU activation is applied to perform feature merging and channel reduction. Finally, a ConvGRU is applied to half ò the channels by split and concatenation.

- Output block: In this part, we use regular convolutions to refine the results. It first concatenates the input image and the bilinearly upsampled output from the previous block. Then it employs 2 repeated convolution, Batch Normalization and ReLU stacks to project to outputs, including 1-channel alpha prediction, 3 channel foreground prediction and 1-channel segmentation prediction.

We adopt Deep Guided Filter (DGF) for high-resolution prediction. When processing high-resolution videos such as 4K and HD, we downsample the input frame by a factor s before passing through the encoder and decoder network. Then the low-resolution alpha, foreground, final hidden features, as well as the high-resolution input frame are provided to the DGF module to produce high-resolution alpha and foreground.