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.

$$I = \alpha F + (1 - \alpha) \qquad (1)$$

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.

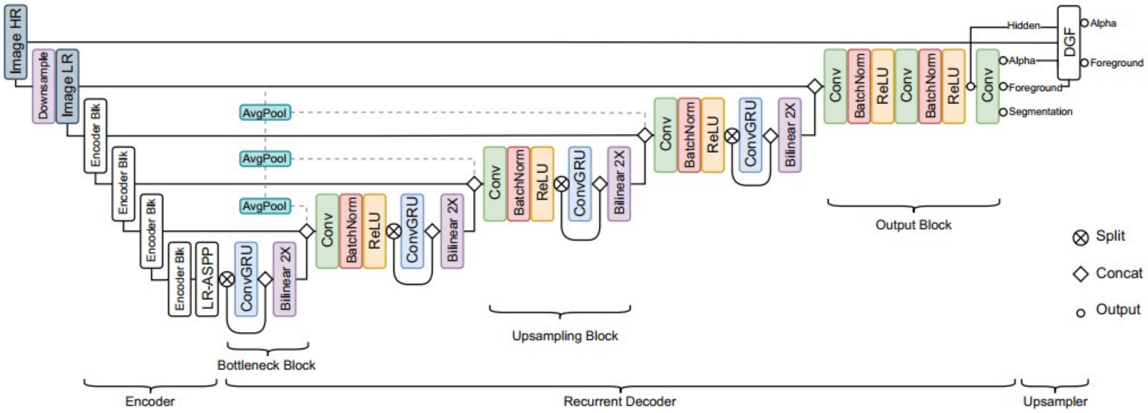

Figure 1. Model architecture

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:

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

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.

2.3. Deep guided filter module

We adopt Deep Guided Filter (DGF) as proposed in 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-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. The entire network is trained end-to-end as described in Section 4. Note that the DGF module is optional and the encoder-decoder network can operate standalone if the video to be processed is low in resolution.

3. Training

We propose to train our network with both matting and semantic segmentation objectives simultaneously.

3.1. Matting Datasets

Our model is trained on VideoMatte240K (VM), Distinctions-646 (D646), and Adobe Image Matting (AIM) datasets. For background, the dataset by [1] provides HD background videos that are suitable for matting composition. The videos include a variety of motions, such as car passing, leaves shaking, and camera movements. We select 3118 clips that does not contain humans and extracts the first 100 frames from every clip. We also crawl 8000 image backgrounds following the approach [2]. The images have more indoor scenes such as offices and living rooms.

We apply motion and temporal augmentations on both foreground and background to increase data variety.

3.2. Segmentation Datasets

We use video segmentation dataset YouTubeVIS and select 2985 clips containing humans. We also use image segmentation datasets COCO and SPD. COCO provides 64,111 images containing humans while SPD provides additional 5711 samples. We apply similar augmentations but without motion, since YouTubeVIS already contains large camera movements and the image segmentation datasets do not require motion augmentation.

3.3. Procedures

Our matting training is pipelined into four stages. They are designed to let our network progressively see longer sequences and higher resolutions to save training time. We use Adam optimizer for training. All stages use batch size B=4 split across 4 Nvidia V100 32G GPU.

Stage 1: We first train on VM at low-resolution without the DGF module for 15 epochs. We set a short sequence length T = 15 frames so that the network can get updated quicker. The MobileNetV3 backbone is initialized with pretrained ImageNet weights and uses 1e −4 learning rate, while the rest of the network uses 2e −4 . We sample the height and width of the input resolution h, w independently between 256 and 512 pixels. This makes our network robust to different resolutions and aspect ratios.

Stage 2: We increase T to 50 frames, reduce the learning rate by half, and keep other settings from stage 1 to train our model for 2 more epochs. This allows our network to see longer sequences and learn long-term dependencies. T = 50 is the longest we can fit on our GPUs.

Stage 3: We attach the DGF module and train on VM with high-resolution samples for 1 epoch. Since high resolution consumes more GPU memory, the sequence length must be set to very short. To avoid our recurrent network overfitting to very short sequences, we train our network on both low-resolution long sequences and high-resolution short sequences. Specifically, the low-resolution pass does not employ DGF and has T = 40 and h, w ∼ (256, 512). The high-resolution pass entails the low-resolution pass and employs DGF with downsample factor s = 0.25, T^ = 6 and h^, w^ ∼ (1024, 2048). We set the learning rate of DGF to 2e −4 and the rest of the network to 1e −5 .

Stage 4: We train on the combined dataset of D646 and AIM for 5 epochs. We increase the decoder learning rate to 5e −5 to let our network adapt and keep other settings from stage 3.

Segmentation: Our segmentation training is interleaved between every matting training iteration. We train the network on image segmentation data after every odd iteration, and on video segmentation data after every even ones. Segmentation training is applied to all stages. For video segmentation data, we use the same B, T, h, w settings following every matting stage. For image segmentation data, we treat them as video sequences of only 1 frame, so T 0 = 1. This gives us room to apply a larger batch size B0 = B ×T. Since the images are feed-forwarded as the first frame, it forces the segmentation to be robust even in the absence of recurrent information.

4. Reference

[1] Yanan Sun, Guanzhi Wang, Qiao Gu, Chi-Keung Tang, and Yu-Wing Tai. Deep video matting via spatio-temporal alignment and aggregation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2021.

[2] Shanchuan Lin, Andrey Ryabtsev, Soumyadip Sengupta, Brian Curless, Steve Seitz, and Ira KemelmacherShlizerman. Real-time high-resolution background matting. In Computer Vision and Pattern Regognition (CVPR), 2021.

[3] https://github.com/PeterL1n/RobustVideoMatting

[4] https://arxiv.org/abs/2108.11515