Portrait Matting Project Progress Report

Chongyu He

May 2021

1 Description

This report will cover the current progress of the *CVPR 2021 Human-centric video matting* competition from `https://competitions.codalab.org/competitions/30523`.

This project aims to perform efficient and accurate human-centric portrait matting in videos, which can be applied to real video conferencing scenarios such as setting virtual background.

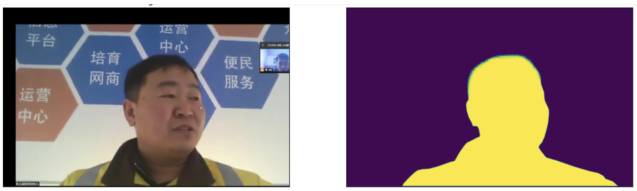

Figure 1: Camera Input with Desired Segmentation Mask

2 Project Setup

1. Download dataset from the competition website or kaggle website
   `https://www.kaggle.com/anguschowchowxu/humancentricvideomatting`
2. Prepare the dataset by running the script
(detailed instruction in Section 4.2)
   ```
   python prepare_dataset.py
   ```
3. Train the model by running the script
   ```
   python fpn_segmentation.py
   ```

3 Package Installed

The section covers the python packages I installed for the project with version.

```
numpy==1.20.3
opencv-python=3.4.2.16
torch==1.8.1
torchvision==0.9.1
segmentation-models-pytorch==0.0.4
albumentations==0.5.2
```

4 Dataset

4.1 Data Source

The source of data are from the competition website `https://competitions.codalab.org/competitions/30523#participate-get-data`. For easy access, we could download it from `https://www.kaggle.com/anguschowchowxu/humancentricvideomatting`. This portrait matting dataset covers a wide range of scenarios from both indoor scenes (e.g., office, bedroom) and outdoor scenes (e.g., park, street). The size of the dataset is around 4GB.

4.2 Data Preparation

After downloading the dataset, we should do some preparations to set up training dataset and validation dataset. The detailed setup instruction is shown below.

Run ‘prepare_dataset.py‘
Parameters:

```
usage: prepare_dataset.py [-h] [-i INPUT] [-o OUTPUT] [-s SIZE] [-l LENGTH]

optional arguments:
  -i INPUT      path to input dataset(default = './train/outdoor')
  -o OUTPUT     path to output directory(default = 'prepared_dataset')
  -s SIZE       number of videos in the desired dataset(default = 10)
  -l LENGTH     number of images in each video(default = 5)
```

Run the script usage:

```
python3 prepare_dataset.py -i path_to_dataset_scenario_dir
```

5 Current Progress

I have already preprocessed the data for training and now using FPN as the backbone of training.

5.1 Training Details

I use pretrained model (imageNet, se_resnext50_32x4d) in training and IoU(intersection over union) as the metric. We can try other pretrained models and other metrics such as Fscores, Accuracy or Recall later. However, due to hardware and time limit, I have not trained a model completely. (The training time for an epoch is now around 15min on my laptop.)

5.2 Different Models

The package segmentation-models-pytorch supports models other than FPN. We can test models based on Unet, Pspnet, Linknet, Manet changing one line of code.

5.3 Benchmark and Visualization

6 Proposal for Next Stage

6.1 Exploring more Models

Instead of using FPN as backbone, we can use other structure such as Resnet, Unet or Pspnet, some of which are also supported by the package segmentation-models-pytorch.

6.2 Setting up Benchmark

We need to set up a profiling tool to test our models. We are going to work on setting up benchmark based on several basic evaluation metrics such as IoU, Fscore.

6.3 Train with GPU

7 What didn't worked but fixed later

7.1 Padding

The size of images from the dataset are not the same. So I padded the image to (1200×2000). However, while training, I found that the training speed is extremely slow and sometimes there might be error about 'image size not match'. I fixed the bug by random crop the image by the size (640).

7.2 Training Speed

The training speed on cpu is extremely slow. If the length of dataset is 200, the training time for 2 epoch will be more than 15min, which is really slow. One important reason for this is that the image size after random crop is now (640×960). I have checked out many examples and found that actually the

training speed is actually reasonable on my laptop. I have downloaded several notebooks, the training time of which is around 12sec per epoch with cuda and 12 num of workers in dataloader on colab. I ran those scripts and found that the training time on my laptop is also around 8min per epoch. So I think it is the hardware that limits the training speed. If we use cuda and more number of workers in dataloader, the training speed will be fast.

8 What didn't worked still to be fixed

8.1 Colab Environment

When I develop the project on Google Colab, there is an issue when I utilize the embedded metric in the *segmentation-models-pytorch* package:

```
code: smp.utils.metrics.IoU(threshold=0.5)
issue: AttributeError: 'smp.utils.metrics' object has no attribute 'iou'
```

I have not fixed the issue yet and I finally used PyCharm as IDE to train the model instead.

9 Possible Extension of Project

This project can be applied to real video conferencing scenarios such as setting virtual background.

10 Resource Used

An interesting package of image augmentation (e.g flip, rotate, resize, random crop, blur etc.)

https://github.com/albumentations-team/albumentations

https://albumentations.ai/docs/api_reference/augmentations/transforms/#albumentations.augmentations.transforms.Blur

https://github.com/qubvel/segmentation_models.pytorch

11 Interesting Resource Saved to be Checked Out Later

https://github.com/nearthlab/image-segmentation

https://github.com/facebookresearch/Detectron/tree/8170b25b425967f8f1c7d715bea3c5b8d9536c/references

https://pytorch.org/vision/stable/datasets.html#voc

https://github.com/facebookresearch/maskrcnn-benchmark/blob/master/demo/panoptic_segmentation_shapes_dataset_demo.ipynb