REPORT FOR AI6126: ADVANCED COMPUTER VISION

# Project 1: CelebAMask Face Parsing

ZENG ZHENG

G211941J

zzeng006@e.ntu.edu.sg

School of Computer Science and Engineering

March 22, 2022

# 1 Project Description

Face parsing assigns pixel-wise labels for each semantic components, e.g., eyes, nose, mouth. The goal of this project is to design and train a face parsing network.

**Dataset** The data used for training comes from the CelebAMask-HQ dataset[1][2]. A small dataset (5,000 trainning and 1,000 validation samples with $512 \times 512$ resolution) will be used to train and evaluate algorithms of face parsing.

**Evaluation** The performance of the model will be evaluated based on the mIoU between the predicted masks and the ground truth of the test set.

# 2 Model

I used ResNet-50 network[1] as the backbone of the model, which consists of 5 stages, each with a convolution and an identity block. Each convolutional block has 3 convolutional layers, and each identity block also has 3 convolutional layers. The ResNet-50 network has over 23 million trainable parameters. K-net was also tried in the model training, but the experimental results proved that it performed poorly in this project.

DepthwiseSeparableASPPHead and FCNHead are used as the decode head and the auxiliary head, respectively. As shown in Table 1.

Table 1: Model structure

| **Components** | **Model name** |
|---|---|
| backbone | ResNet-50 |
| decode head | DepthwiseSeparableASPPHead |
| auxiliary head | FCNHead |

# 3 Loss Function

This is an extremely unbalanced multiclassification problem because some categories such as hair and face have many more pixels than eyes and necklaces. Therefore the experiment uses a **weighted cross-entropy loss function** to calculate the categorization loss. We assign weights based on the total number of pixels in each category; the fewer the pixels in a category, the larger its weight, which should contribute more to the loss value in the case of misclassification. The predicted probability of these nineteen categories can be denoted as $p_{o,c}$. Then the loss function is expressed as follows.

$$Loss = -w_c \times \sum_{c=1}^{M} y_{o,c} log(p_{o,c}) \quad (1)$$

[1]CelebAMask-HQ Dataset `http://mmlab.ie.cuhk.edu.hk/projects/CelebA/CelebAMask_HQ.html`

where $M$ represents the number of classes (skin, nose, and more), $w_c$ represents the weight of class $c$, the log is the natural log calculation, $y$ is the binary indicator (0 or 1) if class label $c$ is the correct classification for observation $o$, and lastly, $p$ represents the predicted result of the observation $o$ is of class $c$.

I counted the number of pixel points of all categories in the training dataset (5000 images) and assigned each class a weight. The total number of pixels and weight assigned for the classes are as Table 1[2].

Table 2: The total number of pixels and weight assigned for every class

| **Classes** | **Num. of pixels** | **Weigh** | **Classes** | **Num. of pixels** | **Weigh** |
|---|---|---|---|---|---|
| 0: background | 375099645 | 1.805 | 10: mouth | 3896683 | 8.3939 |
| 1: skin | 333172471 | 1.976 | 11: upper lip | 5417555 | 7.9185 |
| 2: nose | 27015905 | 5.6004 | 12: lower lip | 8898431 | 7.2026 |
| 3: eye glasses | 3467864 | 8.5621 | 13: hair | 411175974 | 1.6725 |
| 4:left eye | 2937773 | 8.8014 | 14: hat | 12651045 | 6.695 |
| 5: right eye | 2928618 | 8.8059 | 15: earring | 3102372 | 8.7228 |
| 6: left brow | 5572580 | 7.8778 | 16: necklace | 196952 | 12.7002 |
| 7: right brow | 5433404 | 7.9143 | 17: neck | 54031542 | 4.6004 |
| 8: left ear | 6164110 | 7.7323 | 18: cloth | 44507381 | 4.8802 |
| 9: right ear | 5049695 | 8.0199 | | | |

# 4 Parameters

The number of parameters of my model is **43589878**.

# 5 Environment Specifications

The specifications of my trainning machine are listed in Table 3. I used the Google Colab IDE to conduct my experiments, using one Tesla T4 GPU to train my models. In addition, the CUDA version is 11.2 and Pytorch version is 1.6.0.

Table 3: Specifications of my training machine

| **Specifications** | **Parameters** |
|---|---|
| IDE | Google Colab |
| number of GPUs | 1 |
| GPU model | Tesla P100-PCIE-16GB |
| CUDA Version | 11.2 |
| Pytorch version | 1.6.0 |

[2]You can also generate the weights by run *generateClassWeights.py* in my submitted code file.

# 6 Training Curves

After 40,000 thousand iterations, the loss curve and mIOU curves are plotted as Figure 1 and Figure 2, respectively. It seems that the best mIOU is reached around 30,000 iterations (around 0.77), after which the overfitting occurs.

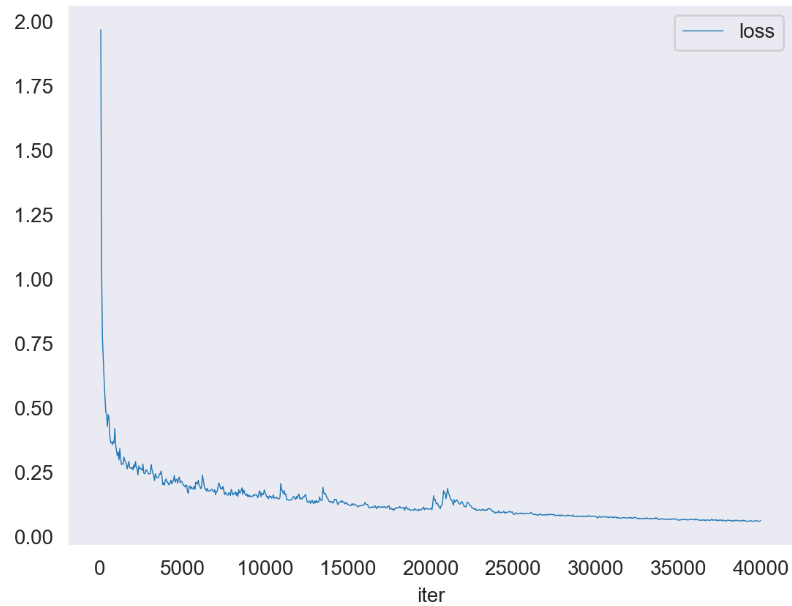


Figure 1: The loss curve for val set

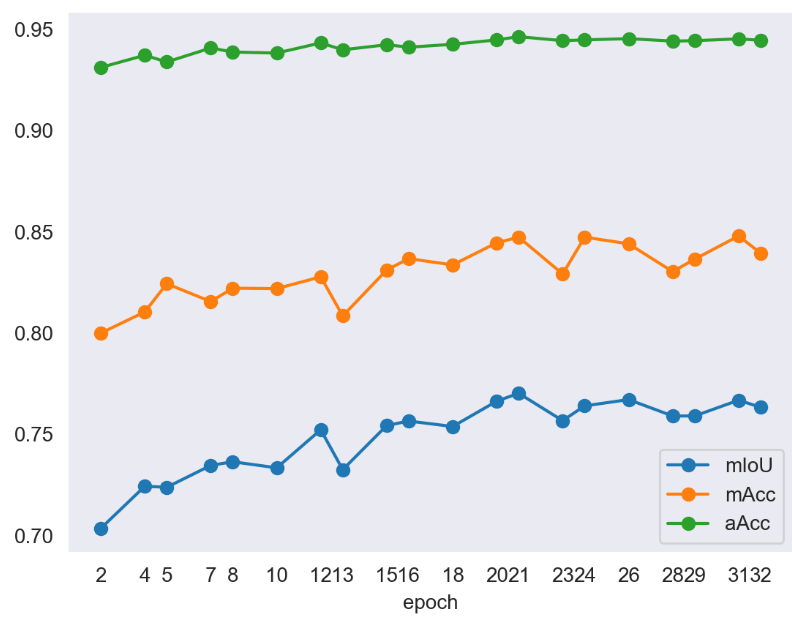


Figure 2: The mIOU curve for val set

# References

[1] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 770–778.

[2] Cheng-Han Lee et al. "MaskGAN: Towards Diverse and Interactive Facial Image Manipulation". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.