Deep Hair Segmentation, Matting and Application

Yawen Sun
yawensun@stanford.edu

Yulian Zhou
zhouyl@stanford.edu

Ziqi Wang
wangziqi@stanford.edu

Abstract

Hair is one of the salient feature in human face region, and hair detection and segmentation have many applications, including face/gender recognition, video surveillance and hair modeling. Recent development in deep learning has demonstrated the great power of Deep Convolutional Neural Networks in semantic segmentation. Yet robust segmentation of hair from portrait images remains a challenging task: hair does not conform to a uniform shape, style or even color, and due to the low resolution of the masks, the edge of hair is poorly recognized.

In this project, we aim to develop a deep learning model for hair segmentation with high resolution. We built our networks based upon DeepLabv3+ network, tested the performace of different backbones, and performed hyperparameter tuning on the baseline model. We experimented with various data argumentation techniques, performed mix-up regularization, and tested different loss functions, especially we combined the regular cross-entropy loss with the gradient consistency loss to increase the resolution. We experimented with alpha matting and fully-connected conditional random field in the post-processing to produce fine-detailed hair mattes. Finally, We built a simple webcam program to perform real-time hair segmentation and coloring on personal computer.

1. Introduction

Hair extraction and modeling are essential in human identification in computer vision. Accurate localization and segmentation of the hair component in face images facilitate face recognition, as well as many beauty applications. Deep Convolutional Neural Networks (DCNNs) have improved the performance of many computer vision tasks, including image classification, object detection and semantic segmentation. DCNNs are trained in a end-to-end manner, and have delivered significantly better results on large datasets than systems relying on hand-crafted features.

Part of our motivation derives from the work by Levinshtein et al.[15], which illustrates the capacity of MobileNets architecture for hair segmentation. Compare to classic VGG-16 network[21], which occupies about 500MB of memory, the modified MobileNet[10] architecture, based on depthwise-separable convolutions, takes only 15MB for this application. Another important aspect of this work is that, the authors have collected crowd-source hair segmentation labels, though noisy and coarse, consisting a much larger dataset (over 9,000 images) than any of the public hair datasets. Moreover, in order to obtain high resolution hair matting, the authors conducted two modifications: 1. They introduced skip connections between layers in the encoder and corresponding layers in the decoder, which benefits the upsampling of low-res layers. 2. They added a second gradient consistency loss function along with the binary cross entropy (BCE) loss, which increases the perceptually accuracy of matting output. With these modifications, the system is able to produce accurate and fine-detailed hair mattes with short computing time on mobile devices.

The success of MobileNet indicates its great potential, yet we want to further push the limit of DCNNs in hair segmentation in three aspects. Firstly, we made use of the DeepLab networks, which have been tuned particularly for the task of semantic image segmentation, and achieved the state-of-art in many segmentation applications. Secondly, since the public LFW hair dataset is small and coarse-grained, we performed data argumentation, such as horizontal reflections, slight rotations, mild scaling, and color jittering. We also processed a private dataset – the CelebHair dataset – to boost the training of the network. Thirdly, the hair boundary is not labeled in details in any of the dataset, which imposes difficulty for the model to properly segment the hair boundary. We tried to overcome this issue with three approaches: 1. We applied the $mixup$ technique, which allows the model to recognize the regions that are labeled partially as hair and partially as background. 2. To obtain high resolution matting, we experimented to add the gradient consistency loss as a second loss function. 3. We tried to integrate hair matting as a post-processing step to generate the matted hair mask.

2. Related Work

In 2016, Chen et al.[2] proposed the **DeepLab** networks, with three major innovations. It uses "atrous" convolution

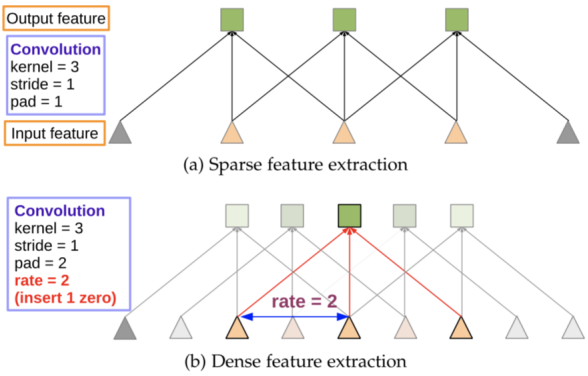

Figure 1: Illustration of atrous convolution in 1-D.

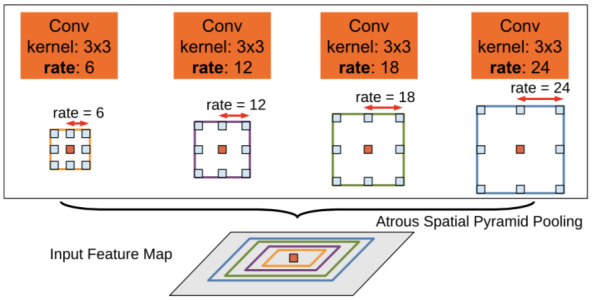

Figure 2: Atrous Spatial Pyramid Pooling (ASPP) employs multi-scale features

for dense feature extraction (Figure 1), which maintains the spatial resolution of the resulting feature maps during upsampling, and is superior to the use 'transposed convolutional' layers, as it doesn't require additional memory or time. Furthermore, it adapts atrous spatial pyramid pooling (ASPP) to represent objects at multiple scales (Figure 2). Finally, though the DCNN score maps can predict the presence and rough position of object, it cannot delineate the border. To address this localization challenge, the Deeplab model couples the recognition capacity of DCNNs and the fine-grained localization accuracy of fully connected conditional random fields (CRFs). Additionally, the authors built their model based upon pre-trained VGG-16[21] and ResNet-101[9], and showed that adopting ResNet-101 instead of VGG-16 significantly improves DeepLab performance.

In 2018, Chen et al.[3] improved the base DeepLab model by combining the spatial pyramid pooling module with encoder-decoder structure (**DeepLabv3+**). The former one captures rich contextual information by pooling features at different resolution, while the latter one is able to obtain sharp object boundaries (Figure 3). The authors also explored the Xception model[4], and applied the depthwise separable convolution to both ASPP and decoder modules, which leads to a faster and stronger encoder-decoder network. This network was successfully adopted by Ma et al.[18] for hair segmentation on time-of-flight images.

Though CRF significantly improves the localization of raw fully convolutional networks (FCN) output, Qin et al. [20] showed that fully-connected CRF itself is not able to produce result good enough for portrait editing, as it fails to recover very thin structures especially for hair, and the use of binary segmentation mask in editing will result in visible boundary effects. To address this issue, they used an image **matting** algorithm to obtain an accurate hair and skin alpha masks. Particularly, an input trimaps is automatically generated from the segmentation label map, and the trimap is fed to an image matting algorithm to calculate an alpha mask. Levinshtein et al.[15] used this algorithm, and integrated matting into their hair segmentation model (HairMatteNet). The HairMatteNet, along with a second **gradient consistency loss** function, produces accurate and sharp hair segmentation output.

Recently, Zhang et al.[25] proposed **mixup**, a simple learning principle to alleviate undesirable issues, such as memorization and sensitivity to adversarial examples, of large deep neural networks. While this approach is more straightforward to be generalized to regression problems than prediction problems such as image segmentation, it has been shown to increase the performance in some medical image segmentation tasks[7]. For hair segmentation, given the mask is rough at the boundary, we hope that mixing up pixels from two images, and converting the part of the image region into partial hair-partial background would benefit the hair edge recognition.

3. Methods

3.1. DeepLabv3+

3.1.1 Network architecture

The model contains three major components: a network backbone, a ASPP pooling layer, and a decoder (Figure 3). We introduce the overall structure in this section, and explain the feature components in details in the next section.
Backbone: The DeepLabv3+ network backbone uses "atrous" (dilated) convolutions. It takes images as input, and outputs both image output as well as feature maps extracted by the atrous convolutions. In this layer, the input is down-sampled, and becomes smaller in size (W and H), and larger in depth (C). Usually, a pre-trained backbone is used to achieve better performance. In this work, we tested pretrained models, including ResNet101[9], MobileNet[10], Xception[4], and DRN (Dilated Residual Networks)[24].
ASPP: This layer contains several ASPP modules, each of which is composed of a atrous convolution layer with varying dilations, a batch normalization layer, and a ReLU layer. Additionally, the ASPP contains a global average pooling

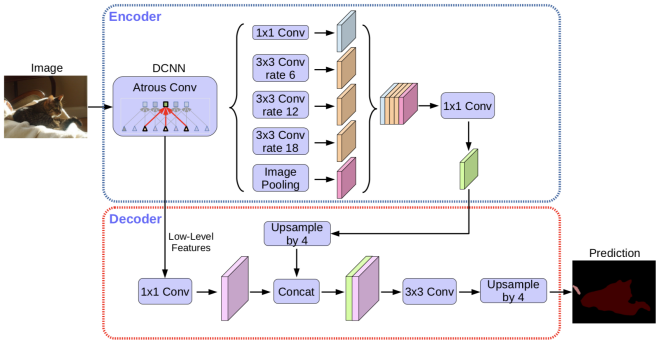

Figure 3: DeepLabv3+ network architecture

layer. The input is passed through all ASPP modules and the pooling layer in parallel, and the output is concatenated, is passed through another Conv-BatchNorm-ReLU layer, before being fed to the decoder.

Decoder: The decoder takes the ASPP layer output as well as extracted low level features from the backbone, and it up-samples the output based upon the low level features. Finally, the up-sampled output and features are concatenated, and are passed through Conv-BatchNorm-ReLU-Conv-BatchNorm-ReLU-Conv to obtain the class probability output. For hair segmentation, the output contains two channels – the hair class channel and the non-hair class channel.

3.1.2 Network features

Atrous convolution: It was originally developed for the efficient computation of the undecimated wavelet transform, and enables computing the responses of any layer at any desirable resolution. Intuitively, the down-sized image is up-sampled filled 'with holes', compared to the full resolution image. Since zero values are introduced, it allows effective filter size increase without extra computation. Mathematically, the output $y[i]$ of atrous convolution of a 1-D input signal $x[i]$ with a filter $w[i]$ of length K is defined as:

$$y[i] = \sum_{k=1}^{K} x[i + r \cdot k]w[k] \tag{1}$$

where the rate parameter r corresponds to the stride to sample the input signal. The standard convolution is a special case for rate $r = 1$. (Figure 1)

Atrous spatial pyramid pooling: Though DCNNs have shown a remarkable ability to implicitly represent scale, simply by being trained on datasets that contain objects of varying size, yet explicitly accounting for object scale can improve the DCNNs ability to successfully handle both large and small objects. The ASPP is inspired by the success of the R-CNN spatial pyramid pooling method, and is implemented as a variant of the scheme using multiple parallel atrous convolutional layers with different sampling rates. The features extracted for each sampling rate are further processed in separate branches and fused to generate the final result. (Figure 2)

Fully-connected conditional random fields: Conditional random fields (CRFs) have been traditionally employed to smooth noisy segmentation maps, and the use of local-range CRFs can potentially improve localization, but misses thin-structures and requires solving an expensive discrete optimization problem. In DeepLab, these limitations are eliminated by integrating the fully connected CRF model. The energy function:

$$E(\boldsymbol{x}) = \sum_{i} \theta_i(x_i) + \sum_{ij} \theta_{ij}(x_i, x_j) \tag{2}$$

where x is the label assignment for pixels. Unary potential $\theta_i(x_i) = -\log P(x_i)$, where $P(x_i)$ is the label assignment probability at pixel i as computed by a DCNN. The pairwise potential:

$$\theta_{ij}(x_i, x_j) = \mu(x_i, x_j)\left[w_1 \exp\left(-\frac{\|p_i - p_j\|^2}{2\sigma_\alpha^2} - \frac{\|I_i - I_j\|^2}{2\sigma_\beta^2}\right) + w_2 \exp\left(-\frac{\|p_i - p_j\|^2}{2\sigma_\gamma^2}\right)\right] \tag{3}$$

where $\mu(x_i, x_j) = 1$ if $x_i \neq x_j$, and 0 otherwise, indicating that only nodes with distinct labels are penalized. The expression uses two Gaussian kernels: the first one depends on both pixel positions (denoted as p) and RGB color (denoted as I), and the second one only depends on pixel positions. The hyper parameters σs control the scale of the kernels.

3.2. Mixup

In a nutshell, $mixup$ constructs virtual training samples:

$$\begin{aligned} \tilde{x} &= \lambda x_i + (1 - \lambda)x_i \\ \tilde{y} &= \lambda y_i + (1 - \lambda)y_i \end{aligned} \tag{4}$$

where (x_i, y_i) and (x_j, y_j) are two feature-target vectors drawn at random from the training data, and $\lambda \sim$ Beta(α, α). The hyper-parameter α controls the strength of interpolation between feature-target pairs, and an empirical value of 0.2 or 0.4 is used on ResNet-101 to alleviate the validation errors on ImageNet-2012[25].

3.3. Loss function

Binary cross-entropy loss: It is a special case for the cross-entropy (CE) loss, which is defined as:

$$\text{CE}(p, y) = \begin{cases} -\log(p), & \text{if } y = 1 \\ -\log(1 - p), & \text{otherwise.} \end{cases} \tag{5}$$

If we define p_t as:

$$p_t = \begin{cases} p, & \text{if } y = 1 \\ 1 - p, & \text{otherwise.} \end{cases} \quad (6)$$

The cross-entropy loss can be rewritten as CE(p,y) = CE(p_t)=-log(p_t). For binary cross-entropy loss, y is the binary indicator, taking the value 0 or 1.

Focal loss: Lin et al. proposed focal loss (FL) for dense object detection[16]. It is derived from cross-entropy loss, while it weights the contribution of each sample to the loss based on the classification error. They first introduce a weighting factor $\alpha \in [0, 1]$ for class 1 and $1 - \alpha$ for class -1, and write the α-balanced CE loss as:

$$\text{CE}(p_t) = -\alpha_t \log(p_t) \quad (7)$$

In practice, α may be set by the inverse of class frequency or treated as a hyperparameter, and it is used to address the class imbalance problem in training. Furthermore, they define the formula for focal loss as:

$$\text{FL}(p_t) = -(1 - p_t)^\gamma \log(p_t) \quad (8)$$

When γ = 0, FL is equivalent to CE, and as γ is increased, it will have more effect on reducing the value of modulating factor (1 - p_t), when $p_t \to 1$. Intuitely, the modulating factor reduces the loss contribution from easy examples, and extends the range in which an example receives low loss. Finally, they combine the two, and use an α-balanced focal loss in practice:

$$\text{FL}(p_t) = -\alpha_t (1 - p_t)^\gamma \log(p_t) \quad (9)$$

Gradient consistency loss: In DeepLab network, CRFs are adopted to capture fine edge details, while Levinshtein et al.[15] proposed a second loss function to generate high-resolution hair matting. Specifically, the mask-image gradient consistency loss is defined as:

$$L_C = \frac{\sum M_{mag}[1 - (I_x M_x + I_y M_y)^2]}{\sum M_{mag}} \quad (10)$$

Where (I_x, I_y) and (M_x, M_y) are the normalized image and mask gradients respectively, and M_{mag} is the mask gradient magnitude. This loss is added to the original BCE loss with a weight w, making the overall loss $L = L_M + wL_C$.

3.4. Matting

Hair segmentation mask is coarse, so is the segmentation output. Therefore, hair matting is needed for perceptual accuracy. Matting aims at extracting a foreground element from a background image by estimating an opacity(Alpha) for each pixel of the foreground element[5]. We calculate the foreground and background colors for all pixels belonging to an image I, along with opacity values. The relationship of the color C, foreground F, and background B is shown in equation:

$$C = \alpha F + (1 - \alpha) B \quad (11)$$

We use Mishima matting[19], which approximates the foreground and background pixels into spheres, and calculates the α distance of unknown pixels to foreground vs. background spheres.

4. Dataset

4.1. LFW parts dataset

Labeled Faces in the Wild (LFW) is a database of face photographs designed for studying the problem of unconstrained face recognition[11]. One of its subset contains labelings of 2,927 250H×250W funneled ("preprocessed" versions) images, which are first segmented into superpixels, and then the superpixels are manually labeled as one of the Hair/Skin/Background classes[14]. The image set is divided as 1,500 in training, 500 in validation, and 927 for testing. In our task, no extra pre-processing step is performed other than transforming RGB mask image to binary hair mask image.

4.2. CelebHair dataset

CelebA is a large-scale face attributes dataset with more than 200,000 218H×178W celebrity images, and the images in this dataset cover large pose variations and background clutter[17]. CelebHair is subset of the CelebA dataset, where 3,556 images are pixel-wise labeled as background, face, or hair[1]. We obtain the segmentation masks, match the masks to their original images, and divide the dataset randomly into 2,556 for train, 500 for validation, and 500 for test.

4.3. Data preprocessing

Basic data argumentation techniques are performed in all of our models, including fix-scale up and random crop, random rotate within (-5, +5) degrees, random horizontal flip (50% probability), and random color jitter (+/-0.05 for brightness, contrast, saturation and hue).

Additionally, we experimented with random scale crop and Guassian blur. Random scale crop resizes the image with its short edge randomly scaled to between [0.5*base_size, 2*base_size], and then pads or randomly crops the resized image to [crop_size, crop_size]. This approach generates training images with different scales from a relatively uniformly scaled dataset, and may implicitly help the network learn multi-scale information. In this work, we use 257 for both base_size and crop_size.

Previous study has show that most of the networks are susceptible to blur and noise distortion[6]. To improve the robustness of our network, we also performed Gaussian blur, and tested its effect on training.

Finally, we normalize the image to the mean and standard deviation of the ImageNet, as all of the pre-trained backbone we used, were trained on ImageNet.

5. Results

5.1. Evaluation metrics

Overall accuracy (Acc): It represents the percent of pixels in the image which were correctly labeled.
Per-class accuracy (Acc_class): It measures the proportion of correctly labelled pixels for each class, and then averages over the classes.
Mean IoU (mIoU): Intersection over Union = Area of Overlap / Area of Union. mIoU reports the avarage IoU for each class.
Frequency-weighted IoU (fwIoU): It is based on mIoU, while it weights each class importance depending on their appearance frequency.
Loss: The total loss value per epoch.

5.2. Networks, backbones and hyperparameters

In the milestone, we reported the metrics of different networks in hair segmentation, and among them, ResNet101 network exhibits the best performance (Appendix A). Yet, we decided to use DeepLab network for two reasons. First, it contains features tuned for semantic segmentation, and achieves the state-or-art in many segmentation tasks. Second, it can adapt pre-trained backbones, including ResNet101.

We first compared the performance of DeepLabv3+ network with different pre-trained backbones on the LFW dataset. The models were trained on Google Cloud with 1 V100 GPU using BCE loss, with Adam optimizer learning_rate = 0.001, weight_decay = 0, batch_size = 32 (for Xception backbone batch_size = 16 due to memory limitation). We fixed the weights of pre-trained backbone, and only trained weights in ASPP and decoder. After training for 20 epoch, we report the model with the best mIoU score on validation set (Table 1; Time/Epoch in second).

Table 1: DeepLabv3+ with different backbones

Backbone	**MobileNet**	**ResNet101**	**DRN**	**Xception**
Acc	95.73%	96.98%	95.93%	93.47%
Acc_class	88.34%	89.92%	87.67%	82.30%
fwIoU	92.28%	94.30%	92.54%	88.74%
mIoU	80.49%	**85.07**%	80.87%	72.43%
Loss	0.05751	0.04279	0.0547	0.3906
Time/Epoch	27.05	44.30	57.15	35.55

We noticed that even though the native implementation of DeepLabv3+ uses Xception module, we obtain better performance with ResNet101. Therefore, we adapted ResNet backbone for the rest of the experiments, and performed hyperparameter tuning with validation mIoU as the criterion to pick the best model.

First, we noticed that Adam optimizer gives more robust training outcome than SGD+momentum, and is more convenient to use. Next we tried to fine tune the backbone weights. Though it is suggested to fix the backbone weights, when the dataset is small, we experimented with fine-tuning backbone weights with $10\times$ or $100\times$ lower learning rates. We found that mIoU is increased from 85.07% to **88.59**% at 20 epoch if we unfroze the backbone, though it leads to overfit soon afterwards (Appendix B). We also experimented with different weight_decay, and found that there is little improvement using weight_decay from 5e-4 to 5e-7 (Appendix B).

5.3. Data argumentation and mixup

Based on these results, we use ResNet backbone in our DeepLab model, and train with Adam optimizer with learning_rate 0.0001 for backbone and 0.001 for non-backbone without weight decay. Unless noted otherwise, we report model with highest mIoU after training over 80 epoch.

We experimented with two data argumentation techniques, random scale crop and Gaussian blur. The baseline model uses fix-resize+random crop, random flip, and color jitter. Since the hair object in our dataset are roughly of the same scale, we want to argument the dataset, and enables it to learn data with different scales using random scale crop (RSC). Indeed, RSC dramatically reduced the overfit problem, and promoted the model performance, when it is trained on the LFW dataset containing only $\sim$1,500 training images (Figure 4 and Table 2).

As networks are generally susceptible to noise and blur, we also introduced Gaussian blur (GB) in our training data, and we showed that it barely affected the validation metrics (Table 2). Still we included GB in the future models, hoping it will make our model more robust to different types of input images other than those seen in the dataset.

Table 2: Effects of data argumentation

	Baseline	**Baseline+RSC**	**Baseline+RSC+GB**
mIoU	89.03%	90.08%	90.15%
Loss	0.03561	0.0324	0.03064

We also performed data $mixup$ on our training data. The idea is that the hair mask is binary (0 or 1), while hair doesn't contain sharp boundary as other types of object. By mixuping images and converting parts of the image to contain both hair and non-hair pixels, would help the model recognize obscure hair boundary. Moreover, the current dataset

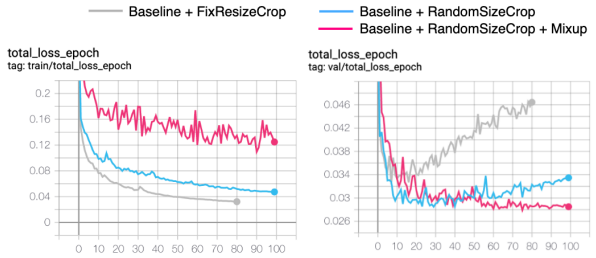

Figure 4: Train and validation loss with random size crop and mixup.

has an intrinsic drawback: the ground-truth only labels hair from one person per image, even if there are multiple person with hair in that image. This biases is problematic as the neural network is good at memorizing data, it might give poor performance on input images with multiple hair regions. Figure 5 shows two examples of $mixup$. (Note: we show mask mixup for visualization purpose, and we don't mixup mask for loss calculation)

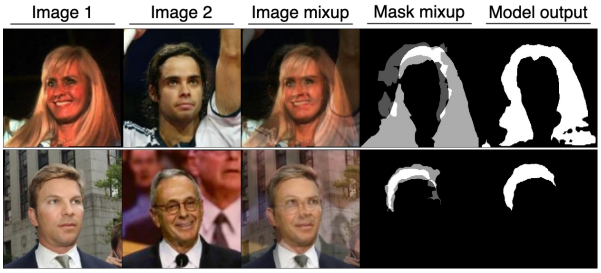

Figure 5: Mixup of training image and mask

We found that mixing-up training data doesn't increase the mIoU significantly, yet it reduces the loss on validation set (Table 3 and Figure 4).

Table 3: Effects of mixup

	No mixup	**Mixup** $\alpha = 0.2$	**Mixup** $\alpha = 0.4$
mIoU	90.15%	90.21%	90.32%
Loss	0.03064	0.02792	0.02855

5.4. Focal Loss and gradient consistency loss

Our baseline model used binary cross-entropy loss, while it has shown that focal loss (FL) would address class imbalance issue, and promote the contribution of incorrectly classified samples. We trained our model with focal loss ($\gamma = 2$ and $\alpha = 0.5$), but the performance is not as good as BCE.

Previous work[15] have shown that using the image-mask gradient consistency (GC) loss enables generating hair matte with high resolution. Figure 6 shows an example of x- and y-gradient of a model output and its corresponding mask. The hair class shows gradient consistent with the mask gradient, while the non-hair class shows gradient opposite to the mask gradient. We combined the BCE loss with gradient loss in training. However, we didn't observe significant improvement on hair segmentation metrics (Table 4) or visual details (data not shown) with GC loss in the absence of hair matting.

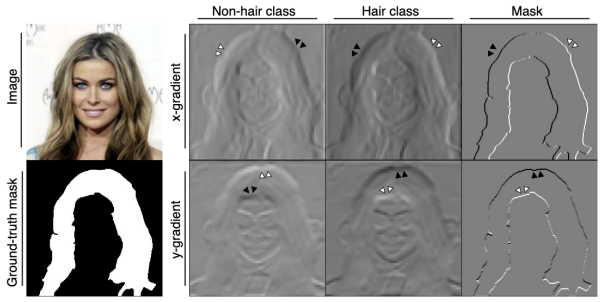

Figure 6: Example of image and mask gradients. White arrowhead indicates high gradient region, and black arrowhead indicates low gradient region.

Table 4: Effects of loss function

Loss	**CE**	**FL**	**CE+GC**
mIoU	90.15%	89.31%	90.00%
Acc	98.02%	97.81%	97.98%

5.5. Error analysis

We did error analysis for our models on LFW test set by visually inspecting the ground-truth label vs. model prediction (Figure 7).

We noticed that there are many cases that our model outputs a better segmentation result than the ground-truth label. On the other hand, there are also failure cases when : 1. The background color is very close to hair color; 2. The model predicts beard hair, while the label of beard is absent, and vice versa; 3. The hair accessories, such as hair band or hat, worsen the prediction in general; 4. There are multiple hair objects in the image; 5. The model predicts bald head with hair, or the model predicts bald head correctly, while the ground-truth is mislabeled as hair. Those prediction errors may be alleviated by using a larger dataset with more training examples in the above category, and with better ground-truth labels.

5.6. Hair matting and conditional random field

We used hair mattting and conditional random fields (CRF) to produce hair matte with rich details and high resolution (Figure 8). We noticed that the effect of matting

Figure 7: Error analysis on LFW dataset

varies depending on different input images. Better visual effect and rich details are achieved, when there is a clear separation between hair and its background.

Though DeepLabv3+ model includes CRF for processing segmentation output, the pytorch re-implementation we adopted doesn't contain the CRF feature. Therefore, we tried to add CRF by ourselves. Surprisingly, adding CRF doesn't promote the visual effect on hair structure, and we would like to investigate into this issue in the future.

Moreover, we also noticed that hair matting reduces model mIoU (Table 5). We think there might be three reasons. 1. The foreground and background color are similar in some input images, and it is more difficult for matting algorithm to resolve the foreground vs. background. 2. The matting assumes dense trimap generated from input image, and it modifies the hair border accordingly, which might not be applicable to all inputs. 3. The ground-truth mask of the dataset are coarse and imperfect, which imposes challenges to the matting.

Table 5: Effects of matting and CRF

	Baseline	**+CRF**	**+Matting**	**+CRF+Matting**
mIoU	90.02%	89.49%	87.01%	86.04%
Acc	98.04%	97.97%	97.40%	97.25%

5.7. Real-time hair segmentation

As we optimized and built our model, we want to put it into application. First, we want to obtain more training data. We discovered a private dataset – CelebHair dataset, and acquired the permission for use in this project. The LFW and CelebHair images are quite similar, and we compared the model performance trained on LFW, CelebHair, and LFW+CelebHair dataset.

Table 6: Quantitative evaluation on different datasets

Dataset		**LFW**	**CelebA**	**LFW+CelebA**
Acc	Val.	98.02%	96.62%	97.32%
	Test	98.04%	96.92%	97.64%
Acc_class	Val.	94.60%	95.87%	95.88%
	Test	94.20%	96.07%	95.69%
fwIoU	Val.	96.24%	93.50%	94.85%
	Test	96.26%	94.09%	95.50%
mIoU	Val.	90.15%	92.21%	92.02%
	Test	90.02%	92.33%	91.62%
Loss	Val.	0.03064	0.04761	0.08247
	Test	0.05463	0.04204	0.09932

Next, we want a smaller backbone and we choose MobileNet, as the model with ResNet101 backbone might be too slow on personal devices. Interestingly, though $mixup$ doesn't increases the evaluation metrics, it consistently makes the model more robust and perform better in this application. We trained a MobileNet-based model on the LFW+CelebA dataset with $mixup$ $\alpha = 0.2$. As we don't need to perform model testing, we combined the train + test set for training. This MobileNet-based DeepLabv3+ model gives **91.6**% mIoU on the validation set, which is comparable to our ResNet-based models.

Finally, we coded up a webcam program for real-time hair segmentation and coloring. We passed individual video frame through the network to generate the hair mask. Instead of coloring the mask to the same extent, we calculate the Softmax probability of the hair class pixels inside of the mask, and apply the color mask proportional to its probability. Figure 9 shows an example of the webcam output (original hair color is black).

6. Conclusion

In this work, we performed hair segmentation with deep convolutional neural networks. Specifically, we adopted DeepLabv3+ network, which achieved the state-of-art performance in many semantic segmentation tasks. We experimented with different backbones, and performed hyperparameter tuning on the baseline model. We also did various data argumentation, and showed that random size crop significantly boosted model performance. We introduced mixup in the semantic segmentation, where its application

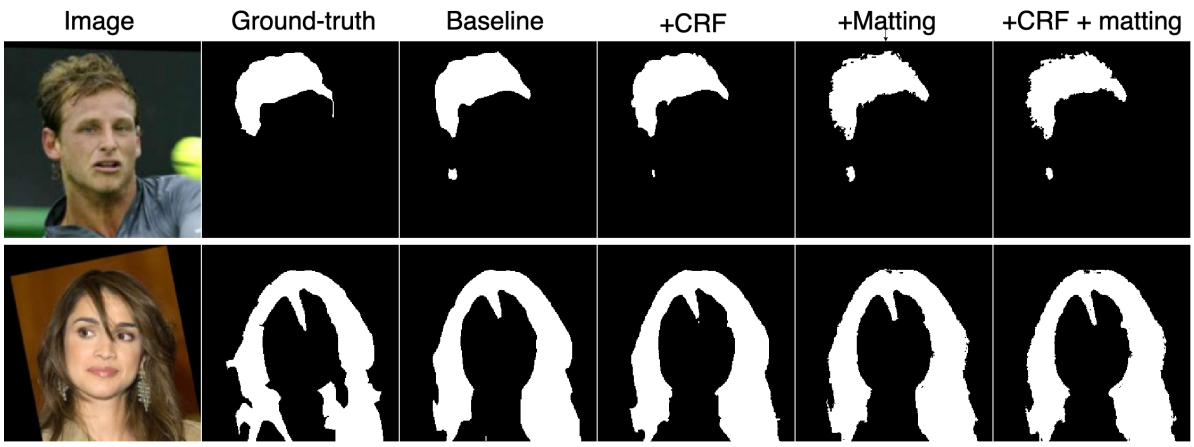

Figure 8: Visualization of alpha-matting and fully-connected CRFs

Figure 9: Real-time hair segmentation and coloring

is largely unexplored. We showed mixup not only decreases the validation loss, but it also makes the model more robust in real-time segmentation application.

To gain better visual effect, we also tried to include alpha matting and fully-connected CRF in the post-processing step. Though it provides richer hair details in some cases, it also compromises the overall segmentation accuracy. For example, using matting sometimes eliminates hair prediction in some small hair region. Moreover, the work by Levinshtein et a. [15] combines the hair matting with gradient consistency loss, while our paper did both but separately. Future work might be set out to integrate matting+gradient and CRF into the training step to obtain better effect.

7. Contribution

Yulian Zhou: Proposed the problem; conducted related work research; performed experiments including hyperparameter tuning (Table 1; Appendix B), data argumentation and mixup (Table 2, 3; Figure 4, 5, 6), adding gradient consistency loss (Table 4), coding up webcam (Figure 9), coding up to perform model testing, as well as preprocessing and integrating celebA dataset (Table 6). Wrote proposal, milestone and final reports.

Yawen Sun and Ziqi Wang: Together conducted related work research; built up baseline by modifying the DeepLabv3 model for hair segmentation; performed experiments including hyperparameter tuning, and preprocessing and intergrating LFW dataset (Table 1; Appendix A; and milestone results); coded and experimented with CRF post-processing and alpha matting (Figure 8; Table 5); coded up and performed error analysis (Figure 7). Contributed to proposal and final report write-up. Ziqi Wang contributed the webcam images (Figure 9).

For data argumentation, code was adapted from [12] and from [23], and modified accordingly.

For $mixup$, code was adapted from [8], and integrated into the network training process.

For gradient consistency loss, the reference code[23] is buggy. It was modify from two aspects: 1. Ignore index is included since image might be padded, 2. x- and y- gradient is normalized to unit vector, which follows the definition in the original paper (Discussed with Bosen Ding).

For webcam, code skeleton was adapted from [13] with three major modifications: 1. The original code does fix-resize, which resizes image (H, W) directly to (224, 224), while we do crop resize with fixed image scale. 2. We normalize the video frame with ImageNet mean and standard deviation before feeding into the network, which is essential for good performance. 3. We add the feature of various

hair coloring options, and the color intensity is proportional to the hair class Softmax probability, which eliminates the sharp segmentation boundary.

For CRF post-processing and alpha matting, the reference code is from [22] and [26].

Our code is publicly available at: `https://github.com/YawenSun9/deep-hair-segmentation`, and the pre-trained model for webcam is under yznew branch (Non-commercial use only).

8. Acknowledgements

The authors would like to thank all CS231N course staff for instructions and guidance on this project, and we especially thank Apoorva Dornadula, Winnie Lin, Haoye Cai, and Kaidi Cao for giving insightful inputs, Bosen Ding for helping debug the gradient consistency loss as well as helpful discussion, Prof. Adrian Sergiu Darabant for the CelebHair mask dataset.

A. Appendix: Networks comparison

Table A1: Performance comparison between different models

Method	Set	Acc	mIoU	Loss
MobileNet	Train	97.8%	80.6%	0.057
	Dev.	97.3%	74.9%	0.080
SqueezeNet	Train	97.5%	78.0%	0.062
	Dev.	97.1%	73.5%	0.076
ResNet101	Train	98.5%	86.6%	0.036
	Dev.	97.9%	79.9%	0.061
DeepLabv3+	Train	97.3%	76.1%	0.065
	Dev.	96.8%	70.8%	0.090

B. Appendix: Hyperparameter tuning

Table B1: Effect of different learning_rate and backbone fine-tune

learning_rate		Best model at epoch #			
		20		80	
backbone	**other**	**mIoU**	**loss**	**mIoU**	**loss**
0	0.001	85.07%	0.04279	85.95%	0.04318
0	0.0005	85.28%	0.04187	86.02%	0.04039
0	0.0001	83.77%	0.04717	85.23%	0.04439
0.0001	0.001	**88.59**%	0.03267	**89.03**%	0.03561
0.00001	0.0001	87.20%	0.0399	87.95%	0.03782
0.00001	0.001	87.60%	0.03584	87.99%	0.03676

Table B2: Dev. mIoU with different weight_decay (best model at 30 epoch)

learning_rate		weight_decay			
backbone	**other**	**5e-4**	**5e-5**	**5e-6**	**5e-7**
0	0.001	78.57%	83.32%	84.44%	85.10%
0.0001	0.001	-	88.12%	88.70%	-

References

[1] D. Borza, T. Ileni, and A. Darabant. A deep learning approach to hair segmentation and color extraction from facial images. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, pages 438–449. Springer, 2018.

[2] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2018.

[3] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 801–818, 2018.

[4] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[5] Y.-Y. Chuang, B. Curless, D. H. Salesin, and R. Szeliski. A bayesian approach to digital matting. In *CVPR (2)*, pages 264–271, 2001.

[6] S. Dodge and L. Karam. Understanding how image quality affects deep neural networks. In *2016 eighth international conference on quality of multimedia experience (QoMEX)*, pages 1–6. IEEE, 2016.

[7] Z. Eaton-Rosen, F. Bragman, S. Ourselin, and M. J. Cardoso. Improving data augmentation for medical image segmentation. 2018.

[8] facebookresearch. mixup: Beyond empirical risk minimization. `https://github.com/facebookresearch/mixup-cifar10`, 2018.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[11] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[12] jfzhang95. Deeplab v3+ model in pytorch. `https://github.com/jfzhang95/pytorch-deeplab-xception`, 2018.

[13] jtiger958. hair-segmentation-pytorch. `https://github.com/jtiger958/hair-segmentation-pytorch`, 2019.

[14] A. Kae, K. Sohn, H. Lee, and E. Learned-Miller. Augmenting CRFs with Boltzmann machine shape priors for image labeling. 2013.

[15] A. Levinshtein, C. Chang, E. Phung, I. Kezele, W. Guo, and P. Aarabi. Real-time deep hair matting on mobile devices. In *2018 15th Conference on Computer and Robot Vision (CRV)*, pages 1–7. IEEE, 2018.

[16] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[17] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[18] Y. Ma, C. Wan, G. Zhang, Q. Jiang, S. Li, and J. Yu. Hair segmentation on time-of-flight rgbd images. *arXiv preprint arXiv:1903.02775*, 2019.

[19] Y. Mishima. Soft edge chroma-key generation based upon hexoctahedral color space, Oct. 11 1994. US Patent 5,355,174.

[20] S. Qin, S. Kim, and R. Manduchi. Automatic skin and hair masking using fully convolutional networks. In *2017 IEEE International Conference on Multimedia and Expo (ICME)*, pages 103–108. IEEE, 2017.

[21] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

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

[23] YBIGTA. pytorch-hair-segmentation. `https://github.com/YBIGTA/pytorch-hair-segmentation`, 2019.

[24] F. Yu, V. Koltun, and T. Funkhouser. Dilated residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 472–480, 2017.

[25] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[26] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1529–1537, 2015.