Real-Time Semantic Segmentation & Applications

Amir Sotoodeh
CS4990: Machine Learning
California Polytechnic University: Pomona

DISCOVERY
College of Science
CAL POLY POMONA

Abstract

Semantic segmentation via Machine Learning models has many practical use cases in countless work areas of today's society. Its applications could be extended into scientific sectors such as autonomous vehicles, terrain mapping, or even be applied in the fashion industry in which allow individuals to apply cosmetic changes to oneself without the necessity of spending money. However, obtaining an accurate mask from a given image poses many issues with respect to the complexity of the desired feature.

With the application of state-of-the art machine learning models and a webcam, real-time semantic segmentation can be achieved for any given image and desired mask. This poster will cover real-time hair and skin segmentation via a computer webcam as well as discuss potential improvements to accuracies obtained with the discussed methods.

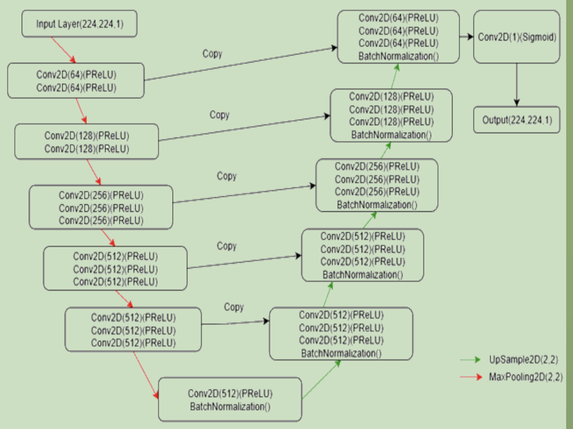
Introduction

The ability to apply computer vision techniques and algorithms to an image of the real-world can have many practical uses. The primary topic of interest for this poster is focused on the applications of semantic segmentation into the fashion/beauty industry. The ability to virtually manipulate hair style or color can be easily accomplished with the help of a basic encoder/decoder machine learning model. However, given that hair can be a very course and complex feature to distinguish, this poses a risk to the accuracy and robustness of a model. Additionally, the model should not be too complex or large in size if we wish to retrieve masks in real-time.

The paper *Real-time deep hair matting on mobile devices* published by the University of Toronto gives a superb explanation of the constraints of mobile devices and web browsers with respect to deep-learning models [1]. Additionally, the paper makes mentions of a hair matting algorithm in which further increases the accuracy of a model as well as enhances the appearance of predicted masks. However, I am primarily focused on the deployment of machine learning models onto web-browsers with the use of webcams. The content of this poster will bill be discussing the means of how to use a live-feed for segmentation, the models used, as well as the issues I ran into during deployment.

Methodology

In order to acquire encoder layers capable of detecting features as complex as hair from an image, I had relied on a model with great performance with respect to feature extraction: VGG16. A barebones U-Net model did not achieve a suffice dice-coefficient; as such, combining the traditional U-Net model with a powerful feature extractor such as VGG16 (imagenet weights) proved to be an adequate model for this task. Given that my target environment would be a web-server and/or web browser, there is much more room for complexity in deep learning models than that of a mobile device. The proposed neural network model found from [2] is as followed:



[2] VGG16-Unet Architecture

By utilizing OpenCV (Open Source Computer Vision Library), capturing images via webcam was a trivial task. The difficulty arises when attempting to format the images into dimensions that are fit for the model. Any preprocessing of the image further adds onto the time complexity of the task, making the application seem less "real-time". This becomes a major issue given that "A single forward passing in VGG16 takes around 100ms even on a powerful GPU" [1]. Thus, there exists a major tradeoff between performance and accuracy for a real-time segmentation application.

Methods to combat this issue will be discussed in the conclusion section and is left as a future project in addition to a k-means cluster hair matting algorithm (applies a "smoothing" effect to the mask).

Data

The dataset utilized to train the VGG16-UNet model is the LFW (Labeled Faces in the Wild) dataset which consists of 2000 training images that are centered, deep funneled, and labeled; this dataset is commonly used for creating advanced facial recognition models.

Little to no preprocessing of data is required given that the dataset is already labeled and modified for training. However, it is important to normalize the training images by dividing each pixel value by 255 (scaling from 0 to 1). Additionally, to augment the dataset, a horizontal flip is applied to each image, resulting in a total of 4,000 training images.

Results

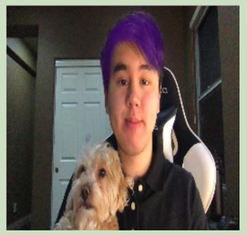
As expected the model had obtained adequate performance with respect to the LFW dataset. The model was trained with the following parameters:

Optimizer: Adadelta	ρ = 0.95
Learning rate: 1.0	ϵ = 1e-7
	Batch Size = 8

Total training time took approximately one hour before early stopping. VGG16-Unet obtained a 0.85 dice coefficient for hair and an 0.955 dice coefficient for skin. With these results, pixel-for-pixel representation in comparison of masks to the ground truth should be acceptable. This held true after applying an alpha mask over a webcam display. To apply an alpha mask using OpenCV, the following equation is required:

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

Where I represents the resulting Image, F is the foreground, B is the background and α is the alpha transparency value. This technique is also known as alpha blending, which applies a transparent overlay to a given image.



Conclusion

In the conclusion of this project, there are still many aspects that I aspire to improve:

- Accuracy of the model
- Application's frame rate
- Refinement of the masks

Each of the aforementioned issues can be amended through a variety of solutions. First, the accuracy of the model can be greatly improved by supplying the model with much more data; LFW (labeled faces in the wild) only consists of 2,000 images which is not enough data to create a robust model without use of heavy data augmentation. With this in mind, it is also import to consider the architecture of the model in order to acquire an adequate frame rate for the webcam-based application. Achieving a better frame rate can likely be solved by converting to a more compact model such as MobileNet. Such an architecture is nearly as accurate as VGG16 and is 32 times smaller and 27 times less computationally expensive [3]. Conversion to a MobileNet Convolutional Neural Network would allow deployment to Mobile Devices for portable applications (i.e Snapchat).

Lastly, in order to refine the masks predicted by the model, there are many algorithms that can solve this problem. The algorithm in question, however, is the K-means clustering algorithm in which gathers clustered predicted points in a mask, producing more refined details to a greater accuracy.

Figure 1
Original Image

Figure 2
Mask

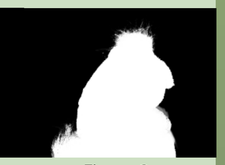


Figure 3
KNN Matting

References & Acknowledgements

[1] Aarabi, Guo, et al., *Real-time deep hair matting on mobile devices.* Retrieved From https://arxiv.org/pdf/1712.07168.pdf
[2] Balakrishna, et al., *Automatic detection of lumen and media in the IVUS images using U-Net with VGG16 Encoder.* Retrieved From https://arxiv.org/pdf/1806.07554.pdf
[3] Howard, Andrew, et al., *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.* Retrieved From https://arxiv.org/pdf/1704.04861.pdf

Thank you Professor Hao Ji for the inspiration to delve into the amazing field of Machine Learning.