Proposal of developing Deep Learning Model with U-NET[1] Neural Network on Carvana Image Masking Dataset[2] using Amazon Sagemaker

Proposed
By
Lu. Htin Aung
For
Udacity's AWS Machine Learning Engineer Course
On Capstone Project

[1] U-Net Neural Network - https://en.wikipedia.org/wiki/U-Net
[2] Carvana Image Masking Dataset from Kaggle - https://www.kaggle.com/c/carvana-image-masking-challenge/data

Table of Contents

Submitted
By
Htin Aung Lu

Project’s Domain Background

This project is based on carvana competition on the kaggle platform. The image segmentation model is performed on the given data without manual image cutting. This project is solely a part of computer vision field. The given data (photos) will be segmented by using a U-Net Network and given a mask to the desired area. This can be used in further implementation of checking the car’s background and prices to more complex model like using to identify cars and people, in production of self-driving vehicles.

In digital image processing and computer vision, image segmentation is the process of partitioning a digital image into multiple segments (sets of pixels, also known as image objects). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics.[3]

CNN became the most used in image processing models since 2010-2012, after that, in 2014, there came the R-CNN(Regions With CNNs) for image detection jobs with the use of bounding boxes. Just one year later, the fast R-CNN is develop with ROI(Region of Interest) Pooling and combining all models in single Net. In 2016, there is faster R-CNN with improvement in more accurate bounding box for regions. In 2017, the Mask R-CNN evolved as faster R-CNN for pixel level segmentation. With the improvement of R-CNN, U-Net NN was first introduced in 2015 to be used mostly in biochemical field and it keeps improving with R-CNN timeline.

Problem Statement

As with any big purchase, full information and transparency are key. While most everyone describes buying a used car as frustrating, it’s just as annoying to sell one, especially online. Shoppers want to know everything about the car but they must rely on often blurry pictures and little information, keeping used car sales a largely inefficient, local industry.

An interesting part of their innovation is a custom rotating photo studio that automatically captures and processes 16 standard images of each vehicle in their inventory. While Carvana takes high quality photos, bright reflections and cars with similar colors as the background cause automation errors, which requires a skilled photo editor to change.[4]

In this project, we’re about to develop an algorithm using the best practices of Sagemaker that automatically removes the photo studio background.

[3] Image Segmentation - http://en.wikipedia.org/wiki/Image_segmentation
[4] Carvana Image Masking challenge from Kaggle - https://www.kaggle.com/c/carvana-image-masking-challenge/

Datasets and Inputs

Dataset is get from the Kaggle challenge page, to reduce the storage usage, we will only use the non-HD version of the datasets. The dataset consists of a large number of car images (as .jpg files). Each car has exactly 16 images, each one taken at different angles. Each car has a unique id and images are named according to id_01.jpg, id_02.jpg ... id_16.jpg. In addition to the images, it also provides with some basic metadata about the car make, model, year, and trim.In the training set, a .gif file that contains the manually cutout mask for each image.

We will then upload these data into the S3 bucket to use effectively with Sagemaker. The images in the train dataset and train_mask dataset will be feed to the Model for training purpose, while the test dataset will be used for validation during the training.

Solution Statement

We will use U-Net algorithm to extract the features on each image, eliminate the unwanted part and rebuild the image to get the masking image of the car or the expected output. We will use sagemaker notebook to implement the solution, first we will try hyper parameter tuning and search for the best hyperparameters to use with the algorithm, after that we will train the model using the debugger and profiler enabled to be able to log the situation and loss condition during the process. After that, deploy the model to a sagemaker endpoint and create lambda functions to start utilizing it on the image of the any type of car.

To define the success of the outcome, first we have our implemented mean dice coefficient to check the accuracy of the result. Moreover, we can see an output is good or not by our eyes also as shown below.

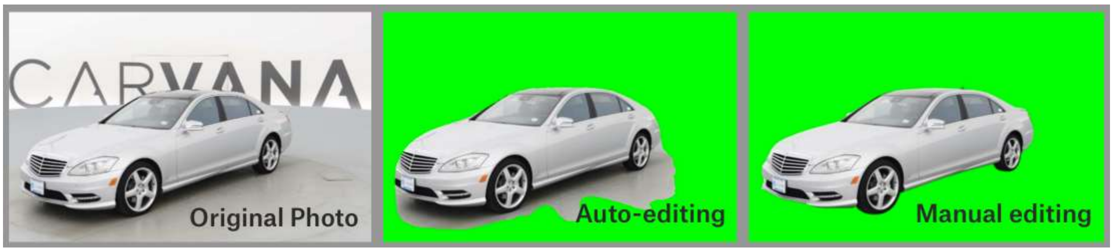

Fig. The Result Example[5]

We can see if the result is as good as the one shown in Manual editing box or not completely masked with model error.

[5] Photo taken from Carvana Kaggle Page - https://storage.googleapis.com/kaggle-competitions/kaggle/3333/media/carvana_graphics.png

Benchmark Model

The benchmark model is taken with permission from a github repo[6]. The model's output is in the output folder. I have also trained that way and record some of the metrics during the training run as follow.

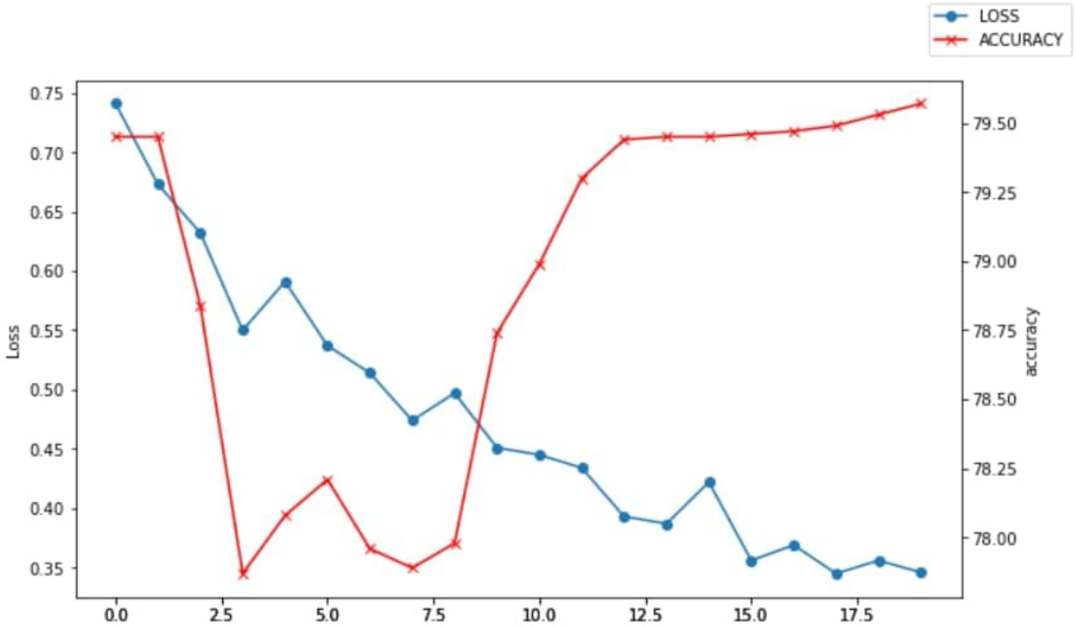

Fig. The Plot Showing the Benchmark Model's Accuracy and Loss

As we can see, the benchmark model is somewhat doing fine, but we might want to reduce the training time, search for the best hyperparameters (number of epochs, batchsize, the learning rate) to get more accurate and smoother model. I will improve the model and operationalizing it using sagemaker's best practices and AWS services.

[6] GitHub Repo of neirinzaralwin - https://github.com/neirinzaralwin/Deep-Learning/tree/main/Neural%20Networks/U-NET%20(Image%20Segmentation)

Evaluation Metrics

We will use the mean Dice coefficient. The Dice coefficient can be used to compare the pixel-wise agreement between a predicted segmentation and its corresponding ground truth. The formula is given by:

$$\frac{2 \times |\mathrm{X} \cap \mathrm{Y}|}{|X| + |Y|}$$

where X is the predicted set of pixels and Y is the ground truth. The Dice coefficient is defined to be 1 when both X and Y are empty. The Metrics is the mean of the Dice coefficients for each image in the test set.

Project Design

The Project design is straightforward, first we will create a sagemaker notebook instance with decent and fair performance. Download the dataset to the instance's environment and after that upload to S3 bucket to use while training. EDA is not needed in this case because dataset is clean and also include manifest files. After that by using sagemaker tuner, we will tune the model with best possible hyperparameters and when it is obtained, we will train the model using sagemaker estimator on a GPU enabled instance. We will then deploy the model to a sagemaker instance and create lambdas functions with appropriate permission to easily utilize the model.

Thank You