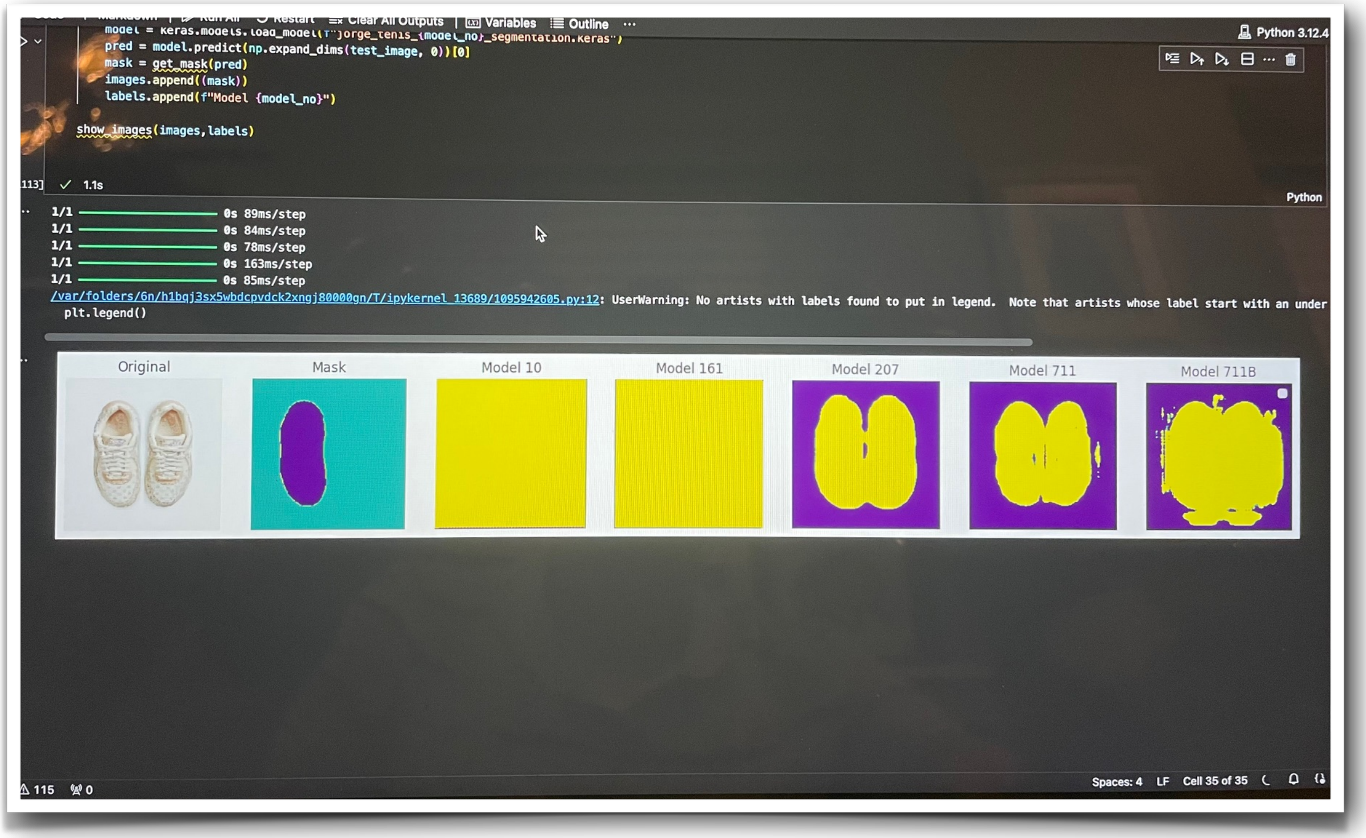
CNN on Image Segmentation

Get to know SAM

Jorge Rodriguez



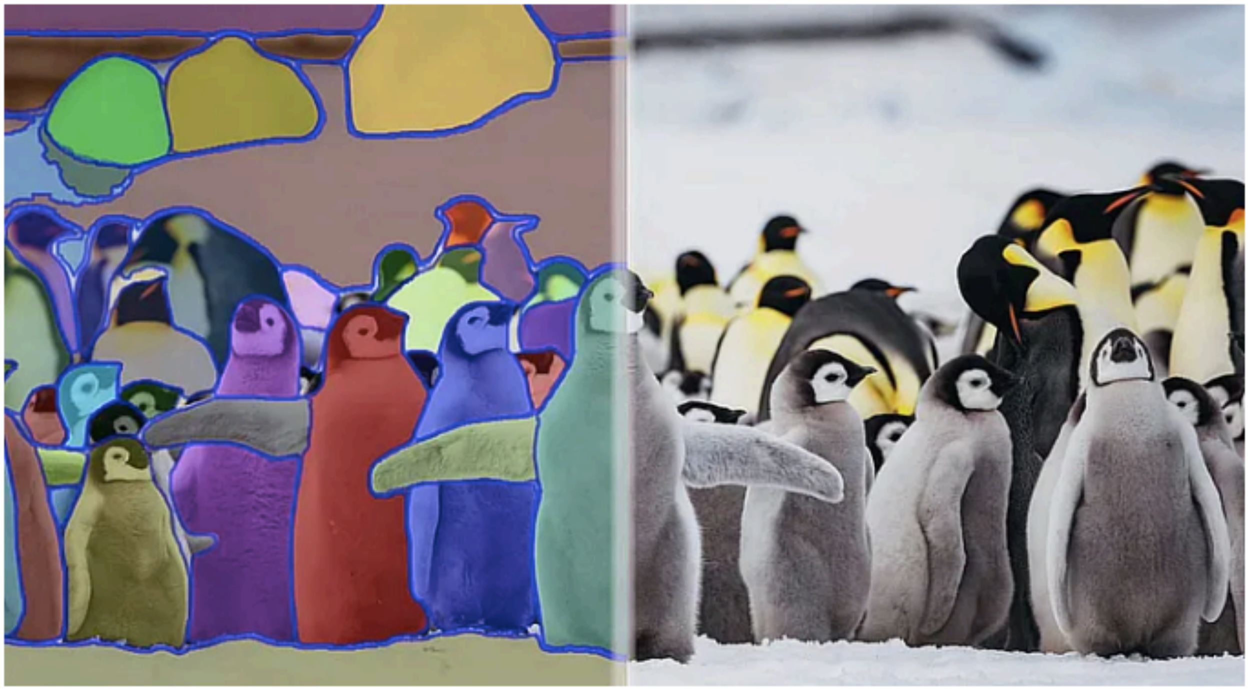
```
        model = keras.models.load_model(f"jorge_tenis_{model_no}_segmentation.keras")
        pred = model.predict(np.expand_dims(test_image, 0))[0]
        mask = get_mask(pred)
        images.append((mask))
        labels.append(f"Model {model_no}")

    show_images(images,labels)
```

```
[113] ✓ 1.1s
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 89ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 84ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 78ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 163ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 85ms/step
/var/folders/6n/h1bqj3sx5wbdcpvdck2xngj80000gn/T/ipykernel_13689/1095942605.py:12: UserWarning: No artists with labels found to put in legend.  Note that artists whose label start with an under
  plt.legend()
```

Comp 4531 - Summer 2024

CNN Image Segmentation

Get to know SAM



https://dataconomy.com/wp-content/uploads/2023/04/SAM-model-Metas-new-Segment-Anything-Model-explained-4.jpg

Introduction:

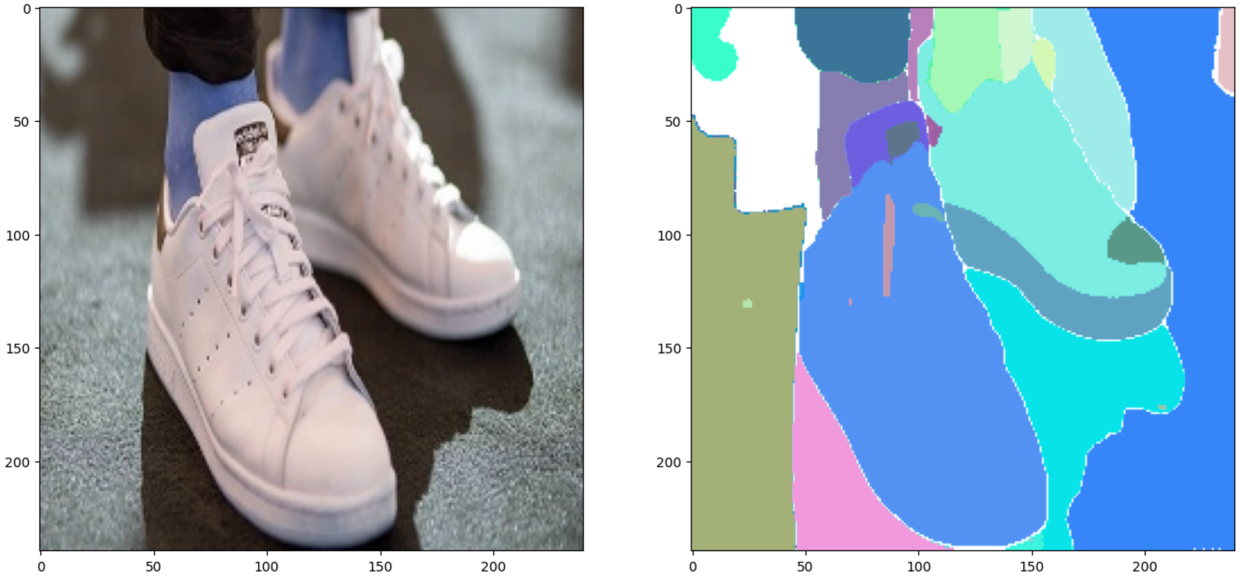
In the field of Computer Vision, the CNN (Convolutional Neural Networks) as an Image Segmentation modeling tool is fascinating and also it is a new field of study on image identification that is constantly evolving. The use of tagging and classification in image segmentation is almost everywhere nowadays, you can see it all they way from self-driving cars to trying to identify potential cancel cells in radiology. Image segmentation is a way of breaking down a digital image into multiple groups known as image segments, which help reduce the image's complexity and simplify processing or analysis. In other words, segmentation involves labeling pixels. Each part of a picture or pixel that falls within the same category is given a unique label. Meta AI developed a very powerful library in Python know as

SAM (Segment Anything Model) that can help us on the Segmentation of Images, but we still need to do a lot of work to making sure the image we are looking for are labeled and identified correctly.

The main challenge lies in getting and prepping the data. Building an image segmentation dataset demands annotating heaps of images to define the labels, which is a massive task and this requires a ton of resources. So, the game changed when the **Segment Anything Model (SAM)** came into the scene. SAM revolutionized this field by enabling anyone to create segmentation masks for their data without relying on labeled data.

Research Questions:

Sample of one of the tennis shoes images, and how SAM segmented it



1.- How can I create an automated process to use **SAM** to identify the most important object in a picture, without telling it which one is, since the pictures will have primary image of tennis shoes, and from there, be able to create the appropriate mask and mask file for it, as well as building the contour around it for proper classification, therefore I can use a CNN model to predict and located the mask of any tennis shoes in any picture.?

2.- How many pictures do I need to have and create masks for, in order for the model to be effective and be able to recognize some of the shoes?

3.- How efficient is SAM to located and identify the most important object in the picture via segmentation.?

Libraries Needed and Pre-Train Models Files:

In order to start working with Image Segmentation, we need to download the pre-trained weights' files from Meta-AI for the image segmentation model. The website to download the PTH file is below:

https://dl.fbaipublicfiles.com/segment_anything/sam_vit_h_4b8939.pth

A PTH file is a machine learning model created using PyTorch, an open-source machine learning library. It contains algorithms used to automatically perform a task, such as upscaling or identifying an image. PTH files can be used in a variety of machine learning and algorithm-related applications, but are most commonly used to upscale images

PyTorch is a machine learning library based on the Torch library, used for applications such as computer vision and natural language processing, originally developed by Meta

OpenCV is a widely acclaimed open-source library that provides an extensive suite of computer vision and image processing functions.

The Segment Anything Model, or SAM, is a cutting-edge image segmentation model that allows for promotable segmentation, providing unparalleled versatility in image analysis tasks. SAM forms the heart of the Segment Anything initiative, a groundbreaking project that introduces a novel model, task, and dataset for image segmentation

a) torch
b) segment_anything
c) opencv-python

Picture Selection and Structure:

The 711 Tennis Shoes pictures used for this project, are saved under the input directory and the 711 mask files generated for the this project, will be saved in the target directory as follow:

input_dir = "./images/"
target_dir = "./annotations/trimaps/"

The 711 images are .jpg format (250,250,3) and the mask files are *.png format (250,250,1).

Mask Generation Process and Creation of PNG (mask) files:

The Segment Anything Model (SAM) produces high-quality object masks from input prompts such as points or boxes, and it can be used to generate masks for all objects in an image. It has been trained on a dataset of 11 million images and 1.1 billion masks and has strong zero-shot performance on a variety of segmentation tasks. The Challenge for me was to select the best mask that was identified by SAM without any human intervention, as it would have defeat the purpose of the project as it was going to take me several days or weeks to go over each of the 711 photos one by one, to make sure SAM or myself selected the correct image-mask pair.

After learning and understanding how SAM library really works, it was not hard to write a python code to read the images one by one, get the "best" masks and then select the top one, which sometimes was the tennis shoe and sometimes was the shoe box and other times it was the tennis shoe and what ever was attached to it, but that was the whole intent to do it this way.

After selecting the best mask, the challenge was how to create a boarder or contour around the mask so the mask file could have the correct data labeling for foreground, background and border (contour).

Once the mask and contour tagging was created, the file needed to be saved as .png format, which required to use the vc2 library to save it with the correct format.

Border/Contour Label Creation:

Border creation or contour was not an easy task to do, since it requires to read pixel by pixel and compare if the previous and next pixels are the same or if there are any changes and then build the label for contour which is this case was "3". So I tried to do all in one step and did not work… so I had to break the task into 3 parts. First, I did a horizontal scan of all the pixels and once that was done, I did a vertical scan of all the pixels, and finally, I had to re-label the background to the correct label number. Since the original mask was a boolean array, I had to convert it to a numerical value first which was not hard.

Model Efficiency and Training the Model:

When I started putting all the pieces together, it became very clear very soon to me that the image mask selection was going to be a challenge, as I sample several of the masks, but to my surprise the majority of them were correct, so I keep moving forward as it was also part of the project to see how well SAM will identify the tennis shoe and to see how well a self-contained program can read images, select the most important image on the picture, create a mask, see it and move on to the next one.

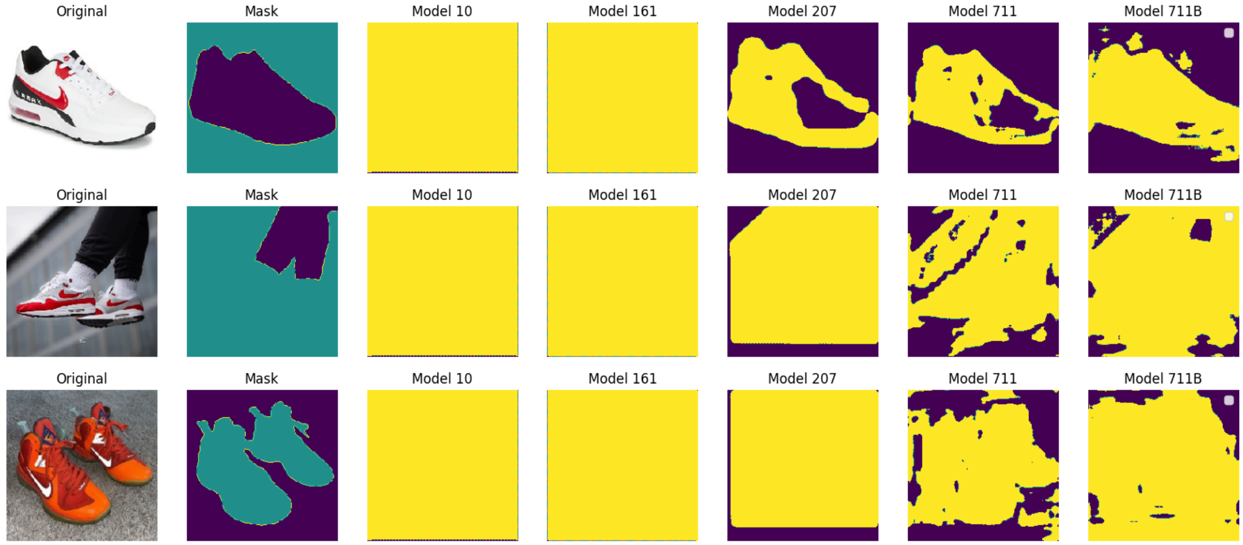
Also, I did realize that the process was taking an average of 1 minute per image to be processed and saved based on a test I ran with 10, 20, 50 and 100 images. So I calculated a total of 711 minutes to read and build 711 masks. Well… that was not the case, as it took over 28 hrs on my Mac-Pro M3 using CPU to do the job. It was a long process so the value to have a GPUs has become more clear to me now.

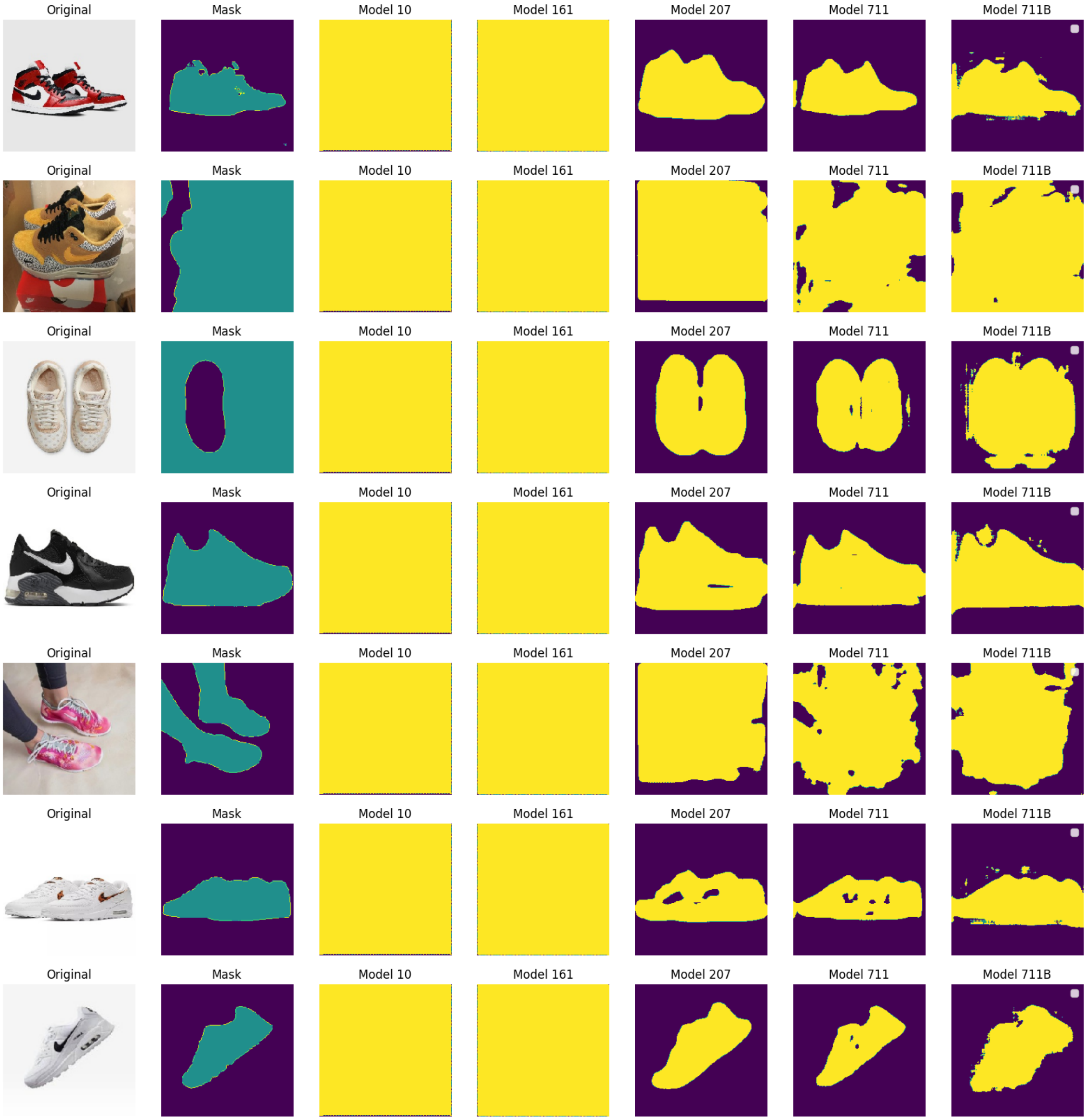
Now with the most important task done (Data Cleaning and Data Collection), which is having the picture and the mask ready to be consumed by the CNN Model; I started to train the model with only 10 images first to see how well it will perform, and to my surprise, I thought for a second that the model and the mask were not working as you can see below on all model 10, the prediction was nothing but just a yellow picture, so I got really concerned about the outcome and the model and the process, etc., so I trained the model again but next time with 161 images, and… nothing, same result; until I finally reached model 207 that I was able to see results in the right direction.

Each time I trained the model, It took time to be trained. The model 10 took a few minutes, the model 161 took over 1 hour, so, I keep adding more and more, and I moved from 161 images to 207 images and the training took over 3 hours. When I processed the full 711 images, it took over 8 hrs of training and I really did not see too much improvement, but it did a great job with other pictures that were not good at model 207, so this proves somehow that the more images you use to train any model, the better predictions.

To mix things a little more, I did model 711B without shuffling the dataset (images) at the beginning to really see if there was a difference on overfitting and generalization and I think it does. It is hard to tell with only 711 images but it is clear that the more images the better model outcome for sure.

Below are some samples of pictures, with their masks and how the CNN model behave on model 10, 161, 207, 711 and 711B. As you also can see, the SAM module sometimes did not selected the tennis shoe as the primary image, so the model got a lot of noise and trained on noise as well, which is a perfect example on how important is to have good clean data and good labels to start with, or the models will not perform at best.



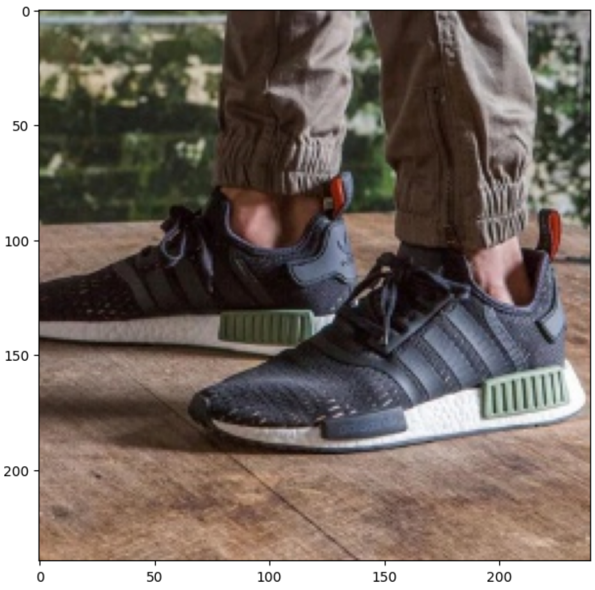

Original
Mask
Model 10
Model 161
Model 207
Model 711
Model 711B
Original
Mask
Model 10
Model 161
Model 207
Model 711
Model 711B
Original
Mask
Model 10
Model 161
Model 207
Model 711
Model 711B
Original
Mask
Model 10
Model 161
Model 207
Model 711
Model 711B
Original
Mask
Model 10
Model 161
Model 207
Model 711
Model 711B
Original
Mask
Model 10
Model 161
Model 207
Model 711
Model 711B
Original
Mask
Model 10
Model 161
Model 207
Model 711
Model 711B

Lessons Learned:

Doing this project, it really opened my eyes to better understand the amount of time, effort and resources required to do any computer vision work, Image segmentation requires hundreds or even thousands of very well labeled images to train the models, All models are not easy to train and require a lot of resources to learn fast, Perhaps one of the most learning lessons I got out of this project, was to learn how pictures are labeled not only on the tagging for humans like a "dog", "cat", "shoe", etc. but more important at the pixel level using masks. I never thought that the concept of mask could be used so well to train a model based on the characteristics of picture, but more important to be able to separate what is important and what is noise.

Image classification is not easy, it requires a lot of time to collect, label, clean and train the models and without big companies like Amazon, Google, Facebook, etc. that have spend millions of dollars in building clean datasets and libraries via open source, I do not think we would be so advance on this field of computer vision as we are now.

Summary:



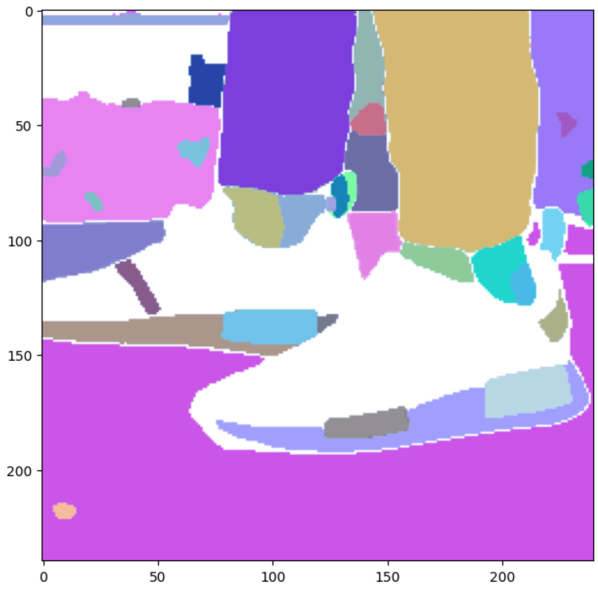


Image segmentation is not easy, requires a lot of work and working with very clean datasets where the models can learn good quality inputs. I can say for sure, that using the SAM pre-train model was a huge savings of time and processing, but it is not 100% perfect when it comes to find the correct mask for the object you want to do more learning. As you can see, the picture above, SAM segmented the picture in so many mask, that even the mask for for the tennis shoes has 4 mask each, making it very hard to build an automated process to extract the correct mask out of an image.

I think the other factor that I never thought was that tennis shoes have a lot of shapes and colors within the same shoe, so when it comes to image segmentation, it is really hard to put all together as just one image to be used for.

Overall, I think the model behaved and learned very well considering all the factors of not having a 100% clean dataset, in addition, it did a good job to demonstrate the capabilities of SAM in conjunction with CNN models on image segmentation. I think there is a huge opportunity on this field to also have a better dataset with labels and masks pre-trained, so it can be easier to consume and build more specific purpose.

The Table blow is to summarize the amount of time it took to run the models and build the mask files using my personal MacBook M3 model.

Time Table:

	Read 711 Images and create 711 Mask Files	Model 10	Model 161	Model 207	Model 711	Model 711B
Total Time:	28 Hrs.	1 min.	1.5 Hr.	3 Hrs.	8 Hrs.	8 Hrs.

GitHub Project Link:

Please find all pictures, trained models and code in the below Git Repository:

https://github.com/jorgeerodriguez/CNN-Image-Segmentation