VisionAnything:

Real-time Image Segmentation using SAM & Deep Learning

Nasrullah Nazaruddin

Contents

List of Tables

List of Figures

1 INTRODUCTION

1.1 Background

Computer Vision(CV) is one of the most vital forefronts of today's emerging technologies. As a field that intertwines itself with AI and countries across the globe are researching, innovating new ways to implement or enhance our daily lifestyle, especially in child development. CV explores object recognition and segmentation, techniques that allow delineation and identification of differing objects within digital images by implementing AI. This allows the computer to understand and detect objects and animals similar to how we as children take time to train our eyes to understand what we are seeing making CV a potential key to further developing a child's cognitive function.

However, emerging technologies like computer vision left a noticeable gap in leveraging the opportunity in image segmentation to help in children's development. A noticeable gap can be filled new CV model called "Segment Anything". The "segment anything" model, also dubbed as the SAM, is a significant advancement in computer vision that enables the segmentation of any object or animal within an image. This allows the ability to isolate objects from their backgrounds, increasing the efficiency of accurate image detection and segmentation.

1.2 Motivation

Compared to other models used in computer vision, the "segment anything" model is a relatively new model with a steep learning curve, implementation towards image segmentation and displaying distinct classification of objects or animals in different images shows great potential for further development but lacks the opportunity since most models used are implemented into medical treatment and identification. This prompts fewer resources in CV tools to be used for childcare development.

Given the scarcity of tools assisting children in learning about everyday objects which are a fundamental aspect of cognitive development, VisionAnything aspires to leverage the "segment anything" models of computer vision.

1.3 Objectives & Scope

VisionAnything is designed to provide a new learning experience for children during the beginning of cognitive training and development. Focusing on everyday objects, VisionAnything has the potential to play a crucial role in moulding a child's cognitive development, promoting an enhanced understanding and recognition of their surrounding world. This innovative method is anticipated to make significant contributions to the array of available educational resources, highlighting intuitive learning and interaction.

The primary objective of VisionAnything is to conceptualise, design, and develop a child-centric application that facilitates the identification of everyday objects and animals. This will be coupled with an emphasis on an intuitive user interface to ensure seamless interaction with the model.

This project will explore:

- A combination of trial and error of using the "segment anything" model to find out which best model will be perfect for VisionAnything, keeping in mind to be user-friendly to children.

- Innovating and researching code-efficient methods to effectively use the “segment anything” model and YOLO-NAS, a neural architecture.
- Training said models and generating the best training date to be used for VisionAnything.
- Effects generated from each model based on differing test images.

2 LITERATURE REVIEW

2.1 Convolutional Neural Networks (**CNN**)

The Convolutional Neural Network, commonly referred to as CNN, is one of the main types of deep learning algorithm that is used for image recognition and classification. Object detection and facial recognition are some of the many areas in CNN that are commonly used. The CNN model trains the machine to recognise the patterns of the image it is being trained on. Patterns like edges of the object, colour and image shape are used when the image input passes through convolutional layers to train the machine to understand and recognise the image pattern and detect a similar one based on the pattern.

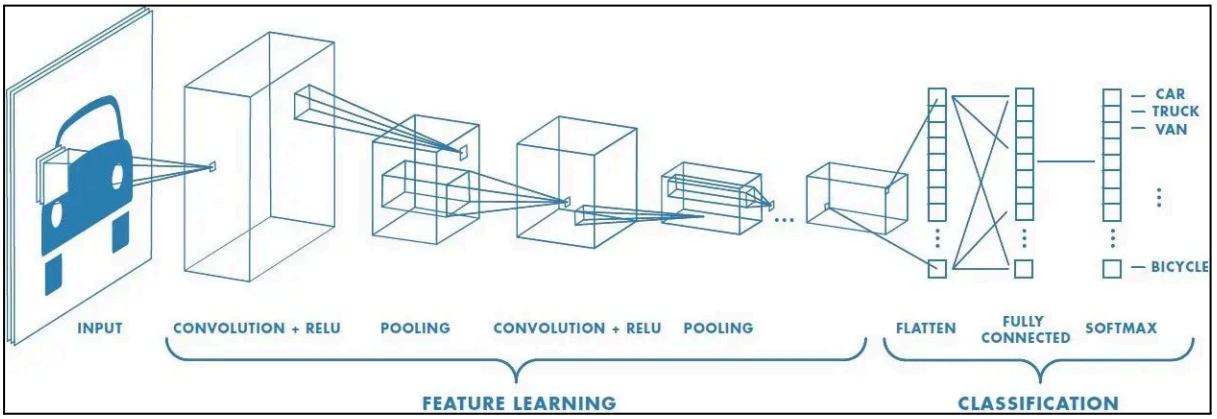


Figure 1. A simple CNN model with multiple hidden layers.
(https://saturncloud.io/blog/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way/)

In respect to this project, the input of the image whether it be an object or animal, the input will go through a series of machine learning layers and ultimately into the convolutional layers which make up the CNN. It will then class the image and identify it into the following categories for training the machine.

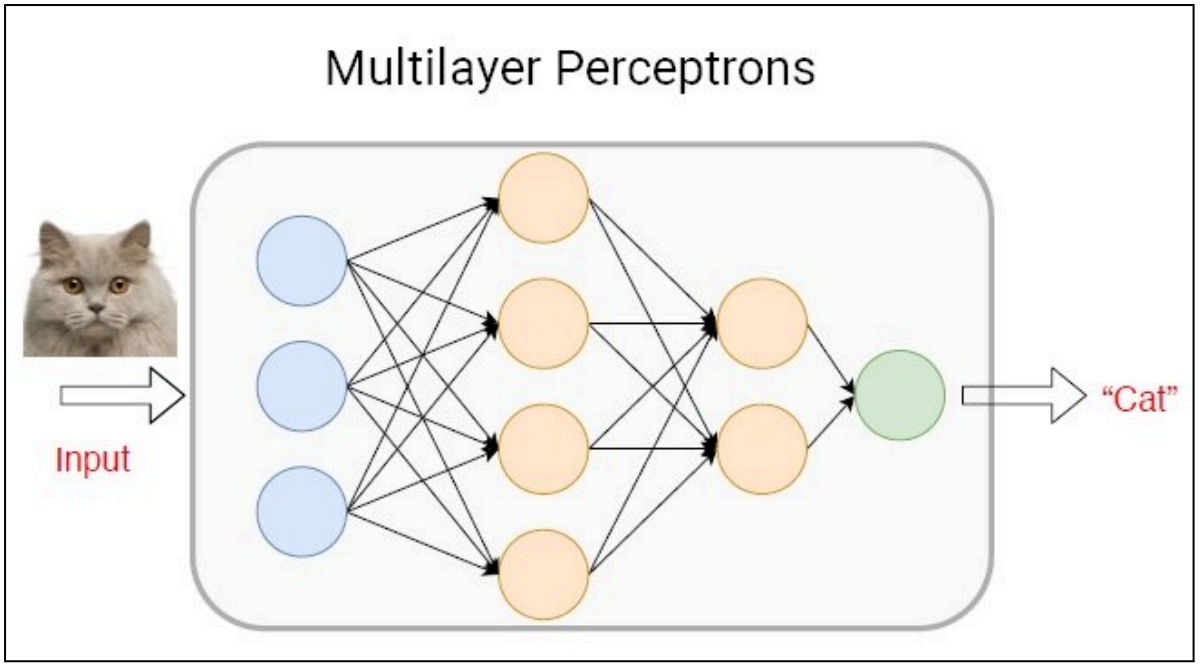


Figure 2. Neural Network CNN layering with input and output.
https://reneelin2019.medium.com/neural-network-concepts-explained-what-are-logits-c39dd96b2e91

2.2 (**SAM**) Segment Anything Model

The Segment Anything Model, commonly called the SAM, is the world's first immense multi-scaled promptable interactive foundation image segmentation model. It allows us to input an image with multiple variable objects and animals and it can detect, classify and even highlight said images all at the same time.

Unlike traditional image segmentation models that require extensive task-specific modelling expertise, SAM removes the need for such specialization. The primary objective is to simplify the segmentation process by serving as a foundational model that can be prompted with various inputs, including clicks, boxes, or text, making it accessible to a broader range of users and applications.

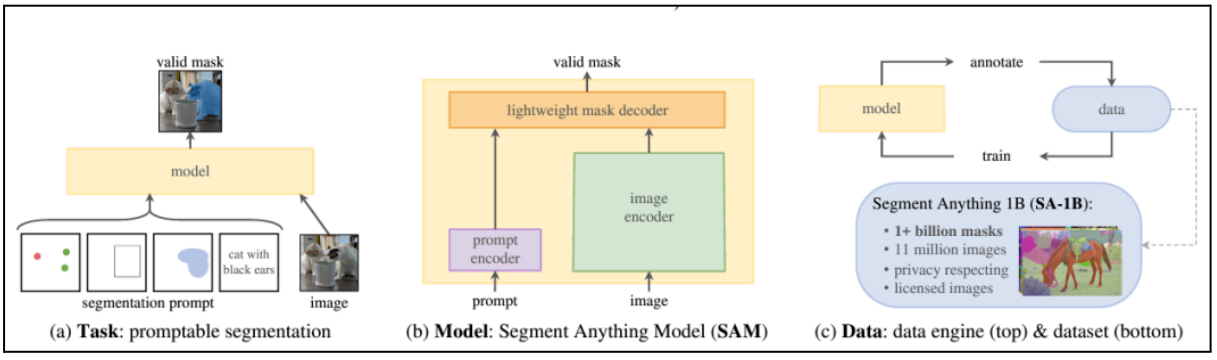


Figure 3. Segment Anything Model (SAM) Model framework
(https://viso.ai/deep-learning/segment-anything-model-sam-explained/)

A SAM has three components, three core components: an image encoder built on Vision Transformers (ViT); a prompt decoder; and lastly, a mask decoder. Each description is provided below:

2.2.1 Image Encoder

The image encoder is at the core of the SAM's architecture, a sophisticated component responsible for processing and transforming input images into a comprehensive set of features. By using a transformer-based approach, this encoder compresses images into a dense feature matrix. This matrix forms the foundational understanding from which the model identifies various image elements.

2.2.2 Prompt Encoder

The prompt encoder is a unique aspect of SAM that interprets various forms of input prompts, be they text-based, points, rough masks, or a combination thereof. It will then translate these prompts into an embedding that guides the segmentation process. This enables the model to focus on specific areas or objects within an image as the input dictates.

2.2.3 Mask Encoder

The Mask Encoder synthesizes the information from both the image and prompt encoders to produce accurate segmentation masks. This component is responsible for the final output, determining the precise contours and areas of each segment within the image. The image encoder first creates a detailed understanding of the entire image, then the prompt encoder adds context, the mask decoder uses this combined information to segment the image accurately, ensuring that the output aligns with the input prompt's intent.

For this project, we will be testing each variant of the Segment Anything Model to test how long would it take for each model to generate masks on two different labelled test images. As for the original SAM model, it took the longest with a total of 145 seconds to generate mask. The other image was not tested due to its first test.

145 seconds to generate mask (2 minutes, 25 seconds)

2.3 (**Mobile SAM**) Mobile Segment Anything Model

The Mobile SAM is a tailored version of the SAM Model and it is used mainly for deploying onto Mobile applications like smartphones or tablets. It maintains the main functionalities of the original SAM model where it allows for promptable and interactive image segmentation.

The idea of using Mobile SAM is to focus more on using resources efficiently and being capable of fast inference. This is what makes Mobile SAM useful for real-time applications where computing resources are limited.
As documented between Mobile SAM and the original SAM, the Mobile SAM is a lightweight model but does not show clear differences compared to the original SAM. Refer below to observe the results between the two models. We can observe that MobileSAM makes a satisfactory mask prediction similar to that of the original SAM but generates the mask at a much faster rate than the original at 83 seconds.

83 seconds to generate mask (1 minute, 23 seconds)

2.4 (**Grounded-SAM**) Grounded Segment Anything Model

The Grounded Segment Anything Model (Grounded-SAM) represents an advancement in the realm of image segmentation, particularly tailored for educational applications and grounded in real-world interactions. Unlike traditional segmentation models that rely solely on abstract representations, Grounded-SAM incorporates contextual information and semantic understanding to enhance the accuracy and relevance of segmentation results.

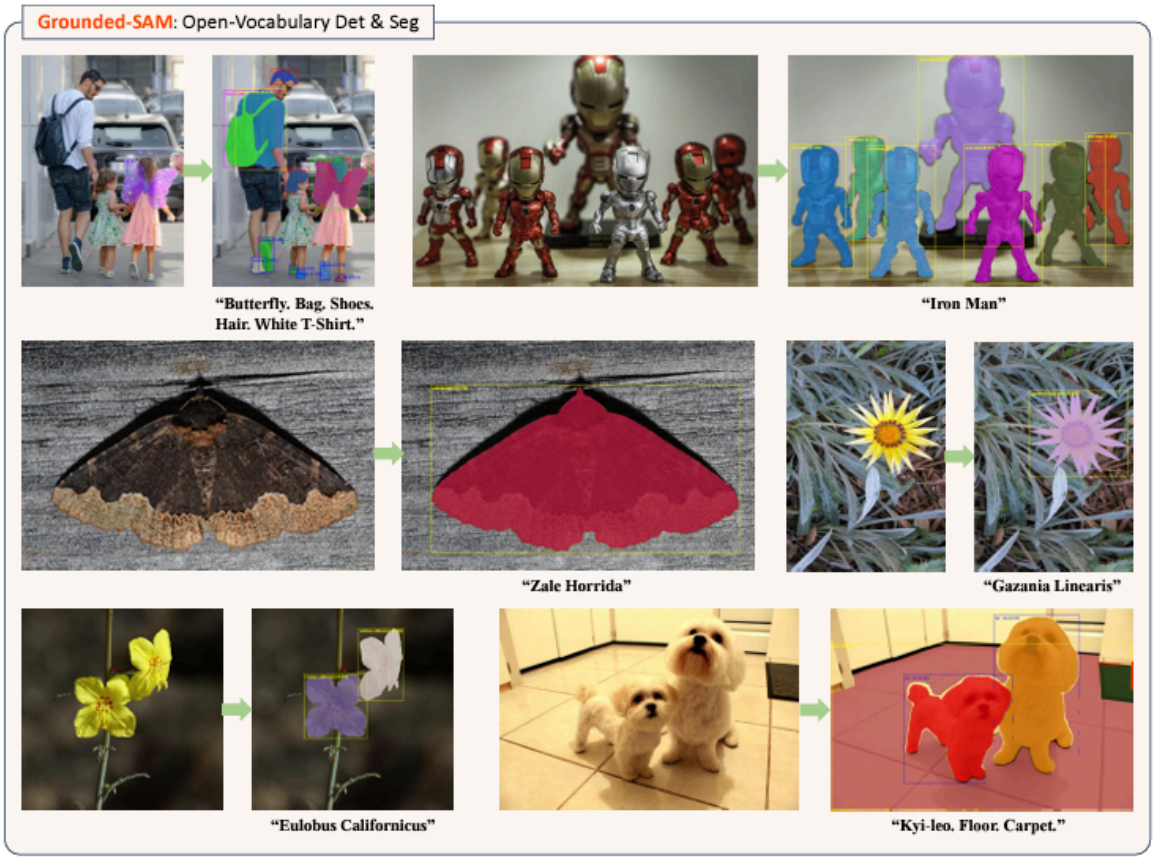


Figure 4. Grounded-SAM Image Segmentation and Labelling (https://arxiv.org/pdf/2401.14159.pdf)

What makes Grounded-SAM unique is its ability to simultaneously detect and segment corresponding regions within images based on arbitrary text inputs provided by users, and as such can seamlessly integrate with other Open-World models to accomplish more intricate visual tasks.

When using Grounded-SAM on a test image shown below, it was able to create a bounding box provided prompting and afterwards mask the image prompted. This segments the labelled image and provides a border around the labelled image. Now, if one were to prompt for the original image to be changed into a new one, Grounded SAM will change it based on the provided prompt. The visual change of the image will then replace the original image.

1. Insert Image

2. Bounding Box around Prompt

3. Image Masking

4. Replacing Original Image with New Prompt (“angry dog”)

(https://felinebreedingacademy.com/course-category/cat-care/)

2.5 Model Comparison

In this section, we will be comparing each model and evaluate based on its benefits and use cases on whether they correspond with VisionAnything. The goal is to choose the correct model which can be used easily and is capable of classifying images and labelling them fast enough and accurately.

2.5.1 CNN vs SAM

Convolutional neural networks (CNN) are one of the main types of deep learning algorithms that is used for image recognition and classification. While CNN requires extensive task-specific modelling expertise, the Segment-Anything Model (SAM) removes the need for such specialization. This allows us to focus more on developing the application and deploying it without a heavy focus on training the model. For this project, we will focus more on using the SAM model, or variants of SAM models like MobileSAM and Grounded-SAM to be used as the model for VisionAnything.

2.5.2 MobileSAM vs SAM

MobileSAM keeps the same pipeline as the original SAM. It inherited the pre-processing, post-processing, and all other interfaces from the original SAM. In theory, assuming everything is the

same except for a smaller image encoder, those who use the original SAM for their projects can adapt to MobileSAM with almost zero effort.

Whole Pipeline (Enc+Dec)	Original SAM	MobileSAM
Paramters	615M	9.66M
Speed	456ms	12ms

Whole Pipeline Parameters and Speed used between Original SAM and MobileSAM

Figure 5. Whole Pipeline Differences between Original SAM and MobileSAM

MobileSAM performs on par with the original SAM (at least visually) and keeps the same pipeline as the original SAM except for a change in the image encoder. Specifically, we replace the original heavyweight ViT-H encoder (632M) with a much smaller Tiny-ViT (5M), allowing the image to be processed at a faster rate. On a single GPU, MobileSAM runs around 12ms per image: 8ms on the image encoder and 4ms on the mask decoder. This makes MobileSAM a more efficient model to be used compared to the original SAM due to its faster rate of image classification.

Model	parameter	1 point prompt
SAM-B	136M	7.02s
FastSAM-s	11M	1.12s
FastSAM-x(default)	68M	2.99
MobileSAM	9.66M	1.43s

Based on The figure provided, it shows SAM with 136M parameters to with a 7.02s 1 point prompt but MobileSAM has a lesser parameter at 68M with a 1.43s 1 point prompt. That is double the speed of the original SAM.

Figure 6. MobileSAM parameter and speed difference

2.5.3 MobileSAM vs Grounded-SAM

MobileSAM is applied to Grounded-SAM as well. Grounded-SAM can simultaneously detect and segment corresponding regions within images based on arbitrary text inputs provided by users and it can seamlessly integrate with other Open-World models to accomplish more intricate visual tasks. However, MobileSAM can also comparatively detect and segment corresponding regions within images, albeit not simultaneously like Grounded-SAM.

While Grounded-SAM does provide the best functionality, it is not strongly as compatible on a mobile device as MobileSAM as Grounded-SAM is closer to the original SAM in terms of processing and generating the image mask. MobileSAM is better suited for VisionAnything and to compensate for the functionalities of a Grounded-SAM model, we will implement the Grounding Dino framework into our model.

2.6 Grounding Dino

Grounding Dino is a zero-shot object-detector framework that is done by combining a Transformer-based DINO detector and grounded pre-training. The model uses a dual-encoder-single decoder design which consists of an image backbone and a text backbone to extract the image and text features from the inputs. This allows the Grounding Dino to classify the images and labels and read back into the dataset to output and classify the image.

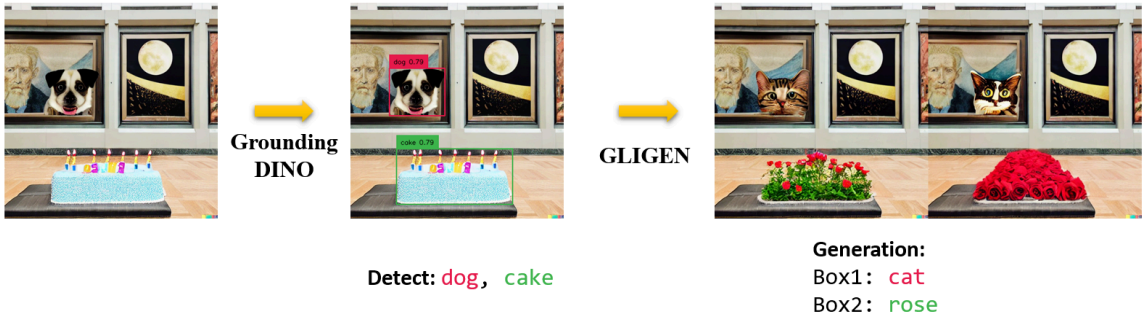


Figure 7. Grounding DINO Framework (https://github.com/IDEA-Research/GroundingDINO)

After testing all types of SAM used above, we will be using MobileSAM for VisionAnything as the model. Mobile SAM is perfect for VisionAnything with its lightweight model being capable of providing the necessary image classification and image labelling needed without a long waiting time to wait for the model to generate a mask. MobileSAM is also applicable to mobile devices which provides the best usability for children and alike to be able to use the application without much lag time.

VisionAnything is designed to provide a revolutionary learning experience to children during their crucial formative years. As such, it has been decided to use MobileSAM paired with GroundingDino while using MobileSAM weights with the GroundingDino Transformer. This allows the web application to be able to label the images and details and be capable of outputting the image for its users.

3 ARCHITECTURE & TECHNOLOGIES

3.1 Proposed Web App Architecture

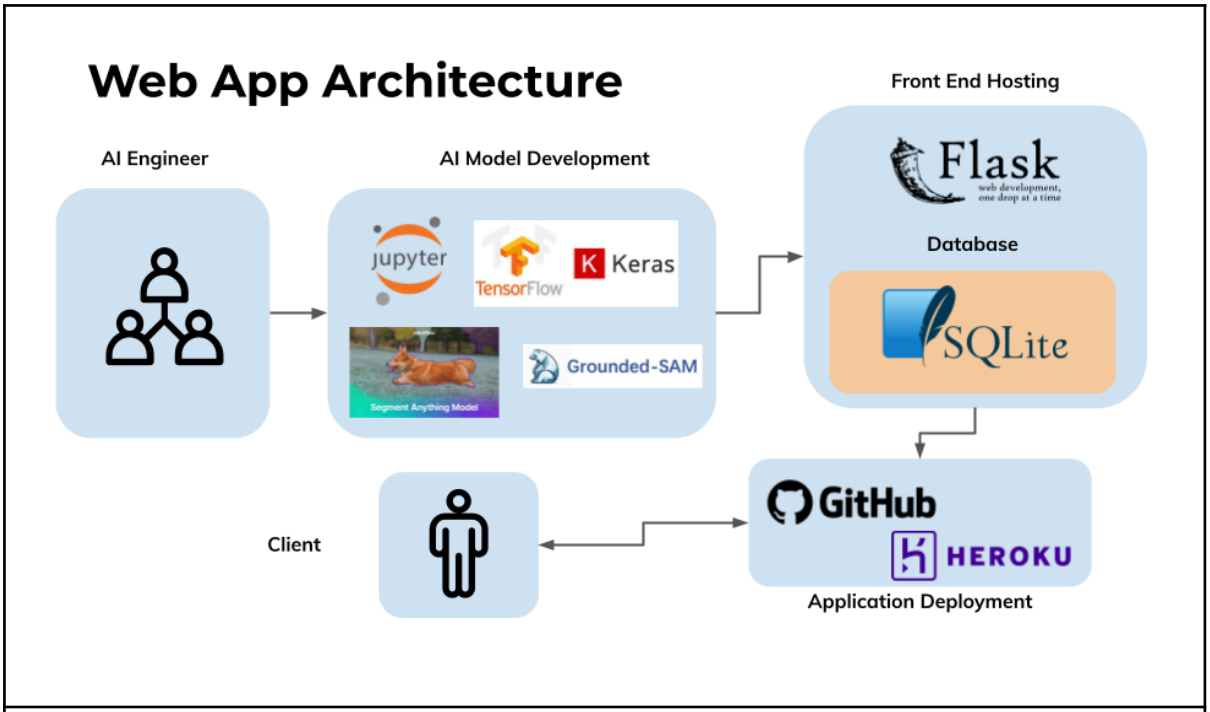


Figure 8. Web App Architecture *(Done in Slides)*

3.2 Technologies Used

- ### 3.2.1 Tensorflow Keras

TensorFlow is an open-sourced end-to-end platform with a library for multiple machine learning tasks. Keras itself is a high-level neural network library that runs on top of TensorFlow. Both provide high-level APIs used for easily building and training models, but Keras is more user-friendly because it's built-in Python.

TensorFlow is a math library used for neural networks and offers a flexible, comprehensive ecosystem of community resources, libraries, and tools that facilitate building and deploying machine learning apps, making it best suited for dataflow programming across a range of tasks.

Keras covers every step of the machine learning workflow, from data processing to hyperparameter tuning to deployment. It was developed with a focus on enabling fast experimentation. This gives users full access to the scalability and cross-platform capabilities of TensorFlow.

Keras is adopted into Tensorflow which can be accessed via the tf.keras module. For this project, we will be using Tensorflow Keras to be integrated into our web application.

- 3.2.2 PyTorch

PyTorch is a relatively new deep-learning framework based on Torch. Developed by Facebook's AI research group and open-sourced on GitHub in 2017, it's used for natural language processing applications. PyTorch has a reputation for simplicity, ease of use, flexibility, efficient memory usage, and dynamic computational graphs. It also feels native, making coding more manageable and increasing processing speed.

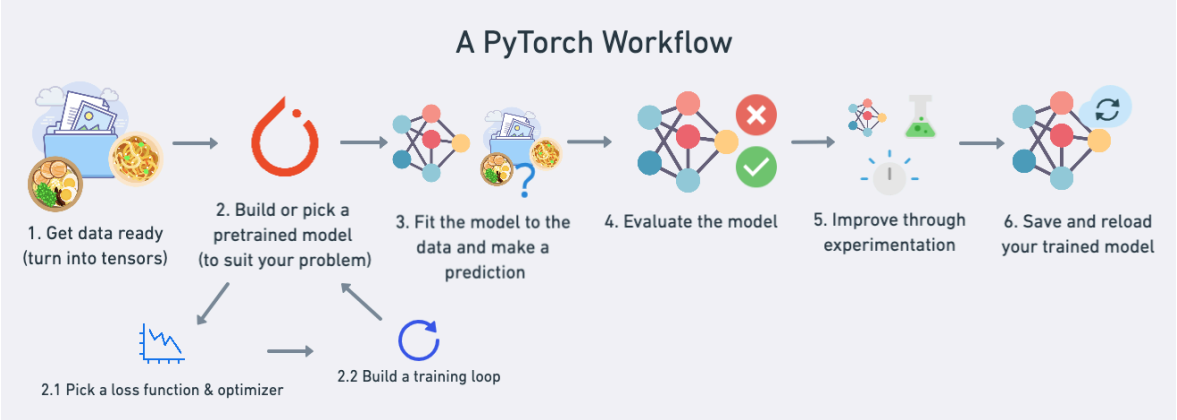


Figure 9. PyTorch Workflow (https://ir.linkedin.com/in/elham-baghbani-66a796101)

Its two main features are:

1. Tensor Computation (similar to NumPy) with strong GPU (Graphical Processing Unit) acceleration support
2. Automatic Differentiation for creating and training deep neural networks

- 3.2.3 Flask

Flask is a web application framework written in Python that lets you develop web applications easily. It has a small and easy-to-extend core: it's a microframework that doesn't include an ORM (Object Relational Manager) or such features.

It does have many cool features like URL routing and a template engine. It is a WSGI web app framework. For this project, we will use Python Flask alongside SQLite to build and develop the web application platform and features for VisionAnything

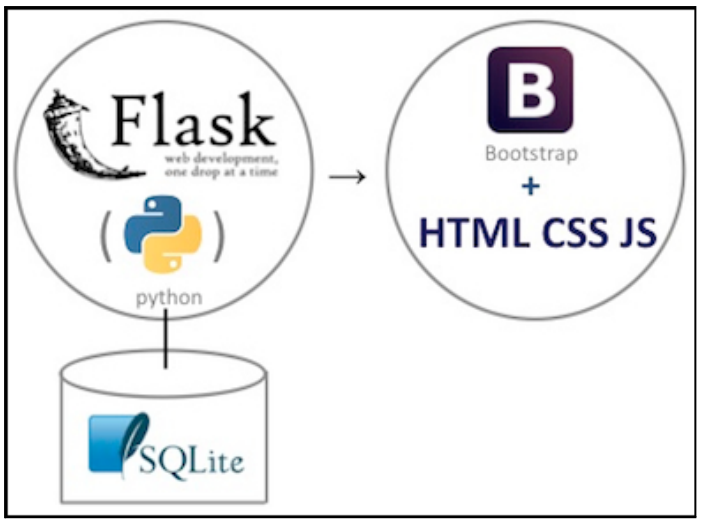


Figure 10. Python Flask used with SQLite (https://medium.com/@gauravc2708/deploy-your-ml-scikit-learn-code-to-production-d470899f9bf5)

- 3.2.4 SQLite

SQLite is an embedded, server-less relational database management system. It is an in-memory open-source library with zero configuration and does not require any installation. Also, it is very convenient as it's less than 500kb in size, which is much less than other database management systems.

As an open-source software, SQLite is serverless and does not need a different server process or system to operate. This allows the system to facilitate us to work on multiple databases in the same session simultaneously and makes it flexible for us as users.

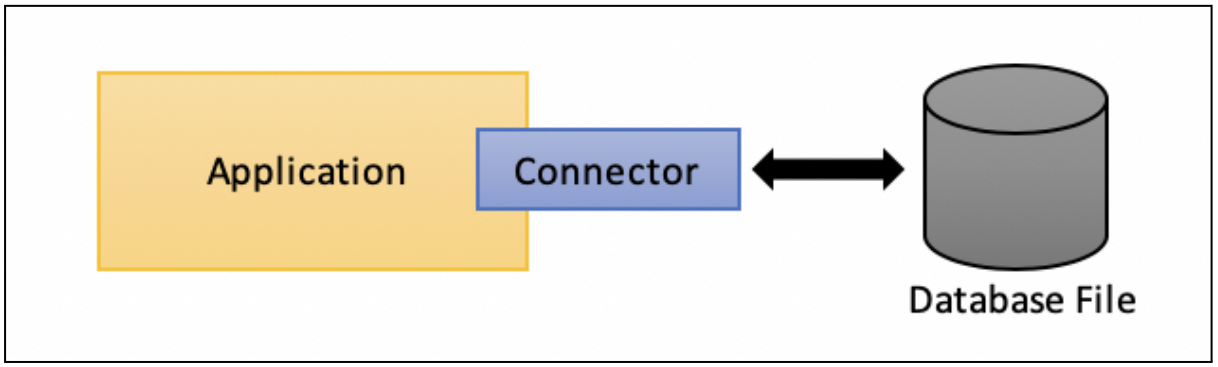
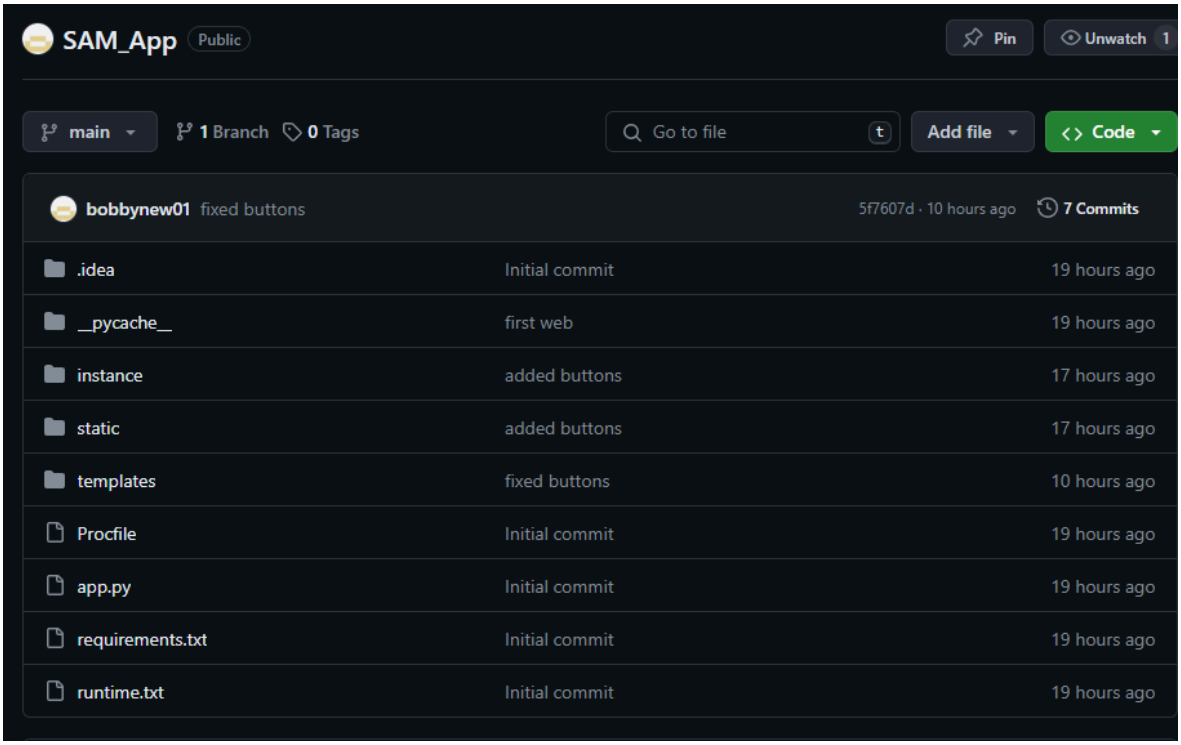


Figure 11. SQLite System Workflow (https://www.spheregen.com/sqlite/)

The main use of SQLite is that it is mainly used to develop embedded software for devices including mobile devices. This makes it perfect for us to implement SQLite for VisionAnything.

- 3.2.5 Github

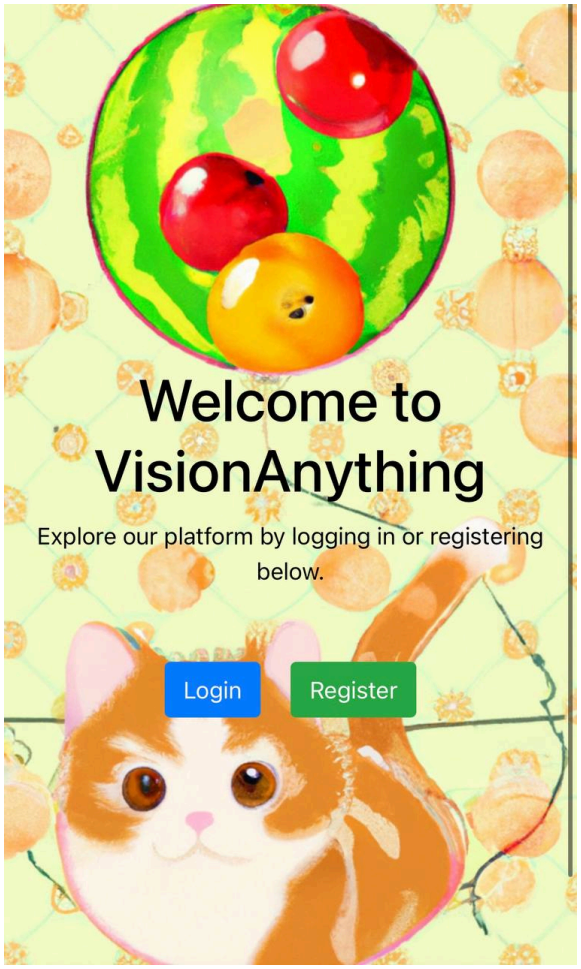
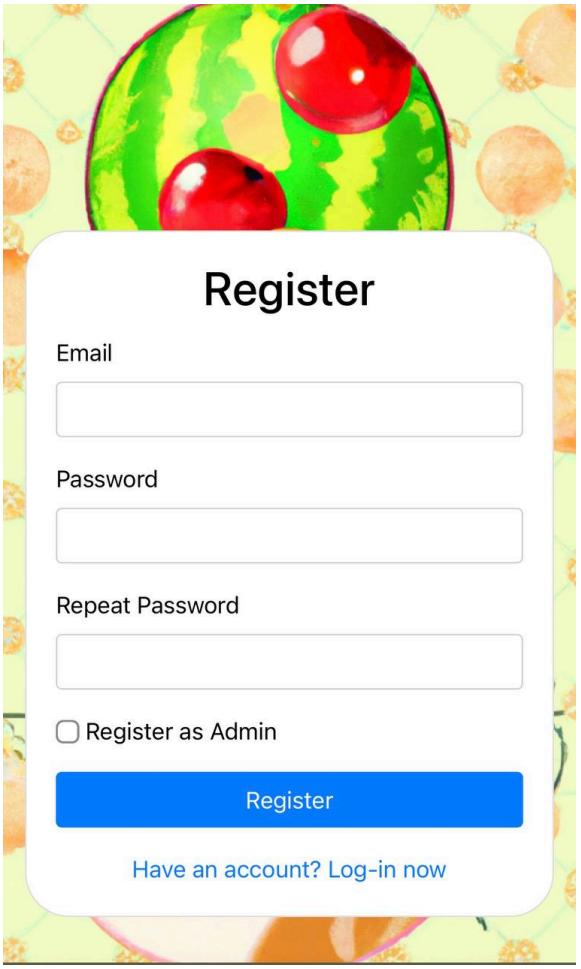


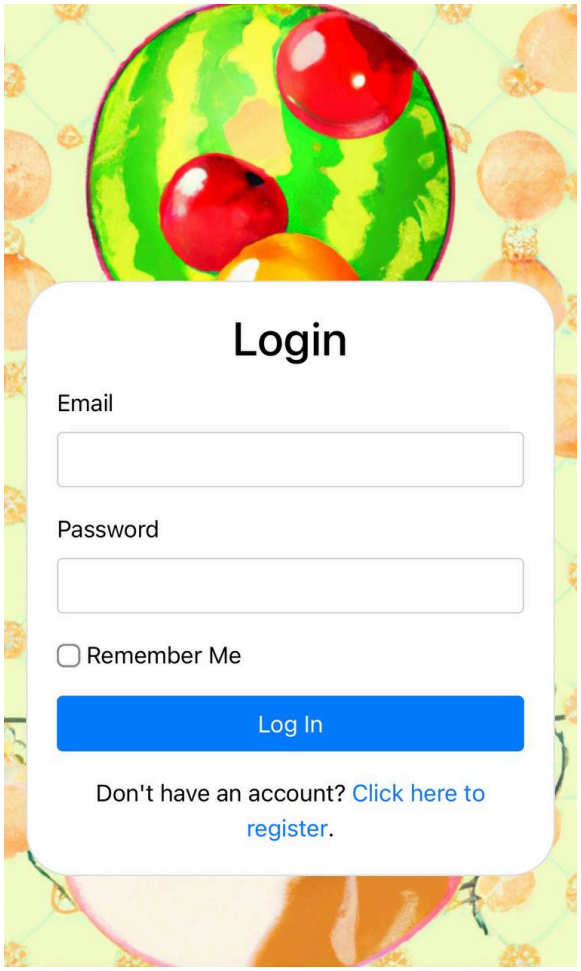
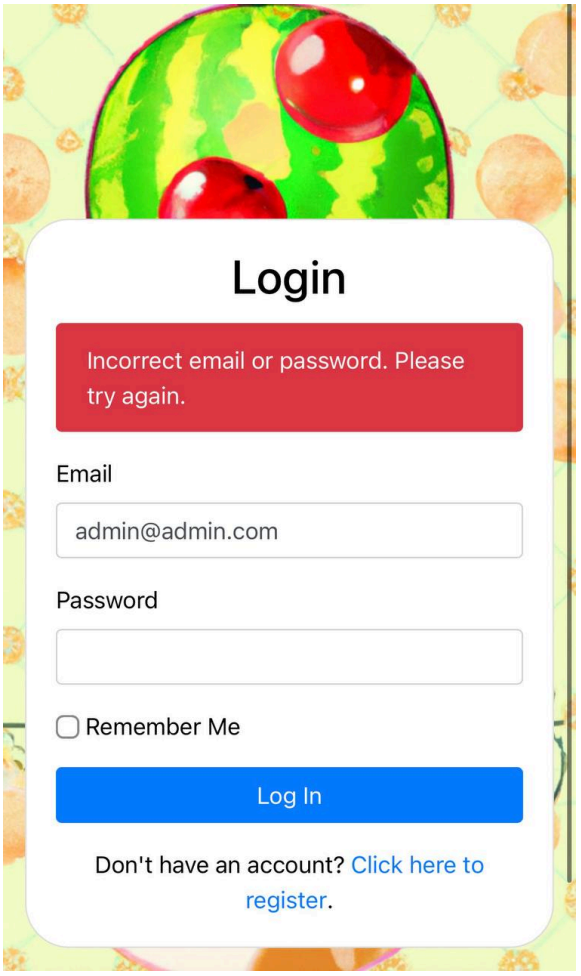
A version control system like Github is used to upload the files of the web application to a repository. This serves as a way to save changes and build the web application from scratch with login functionalities and usage of the MobileSAM/Grounded-MobileSAM Model.

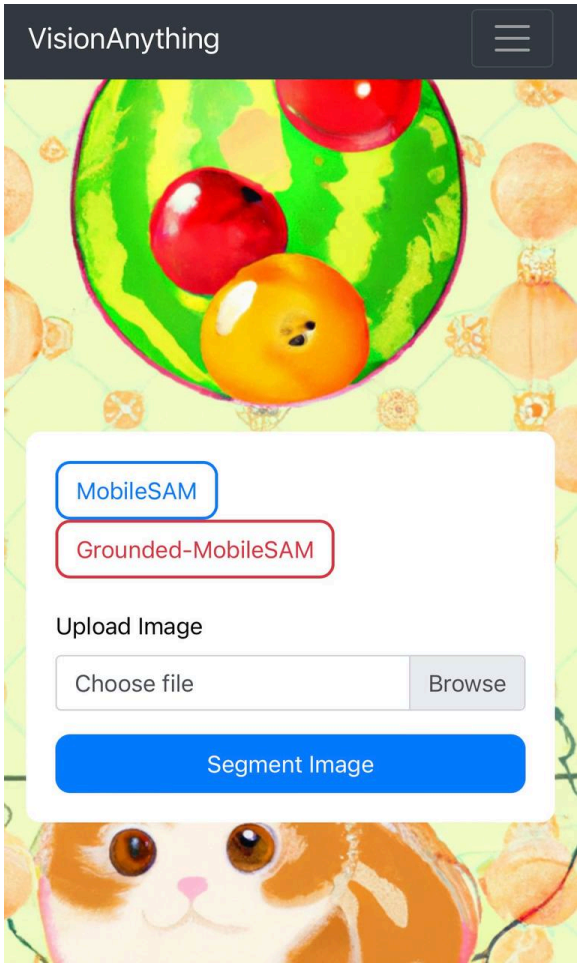
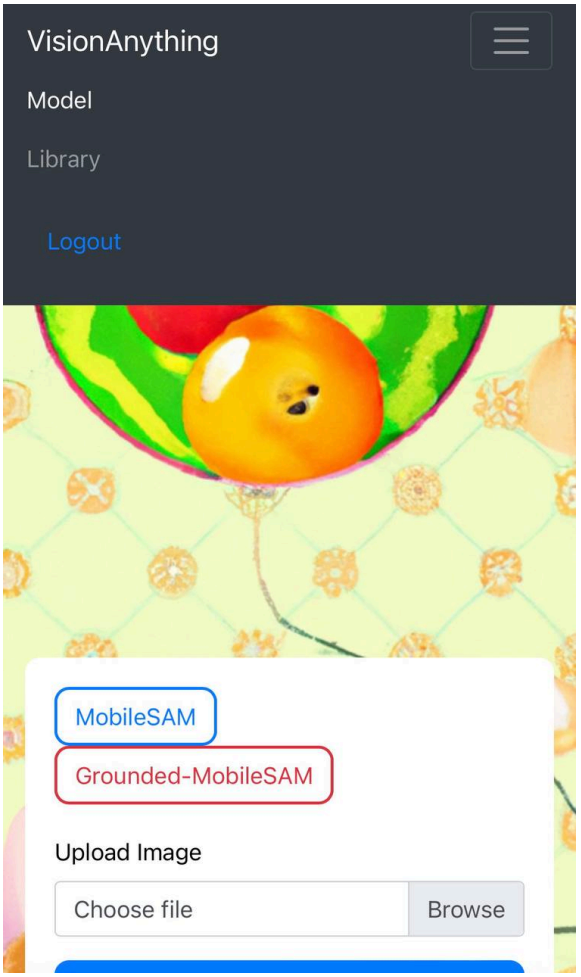
- 3.2.6 Heroku

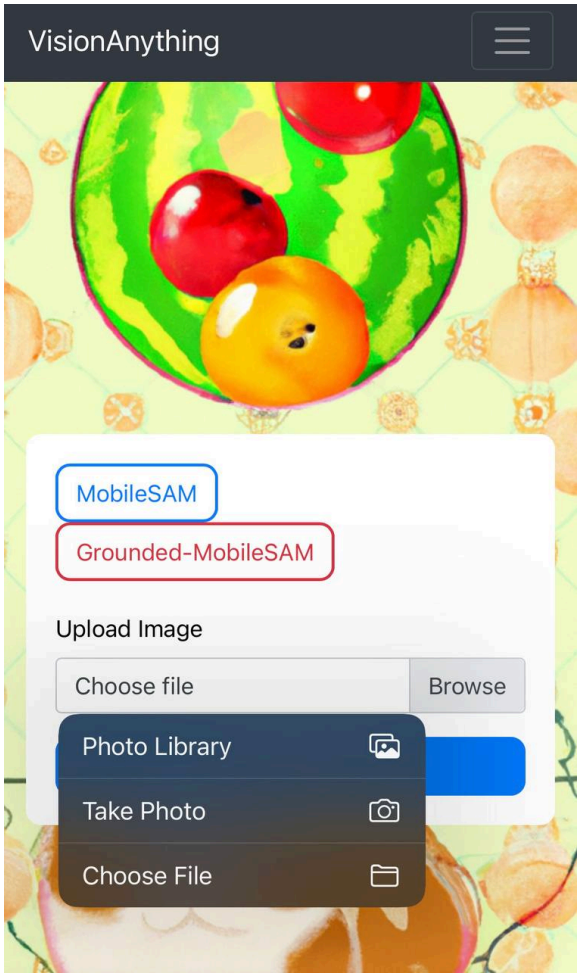
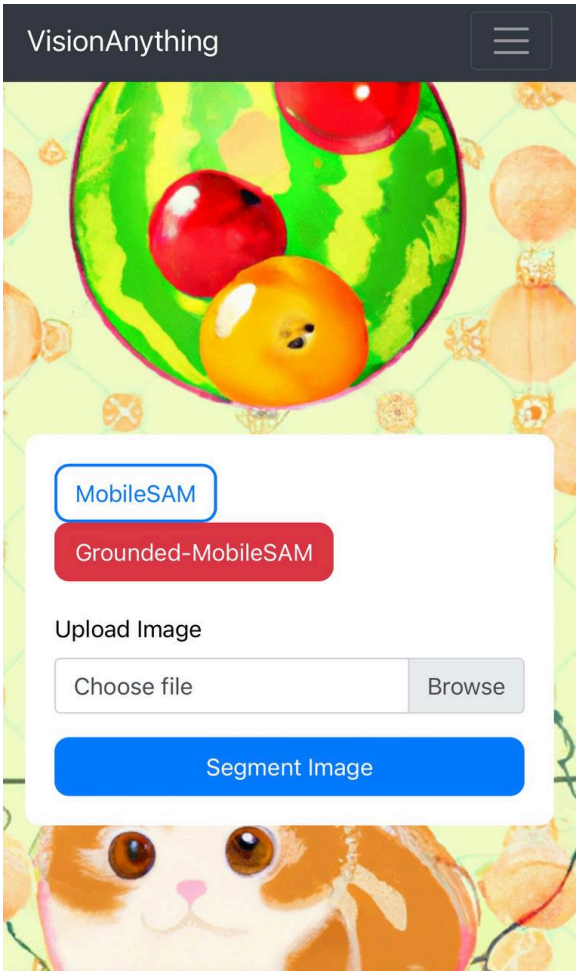
Heroku is a container-based cloud Platform as a Service (PaaS). Heroku is used to deploy, manage, and scale modern apps. Heroku is fully managed, giving us the users the freedom to focus on our core product without the distraction of maintaining servers, hardware, or infrastructure.

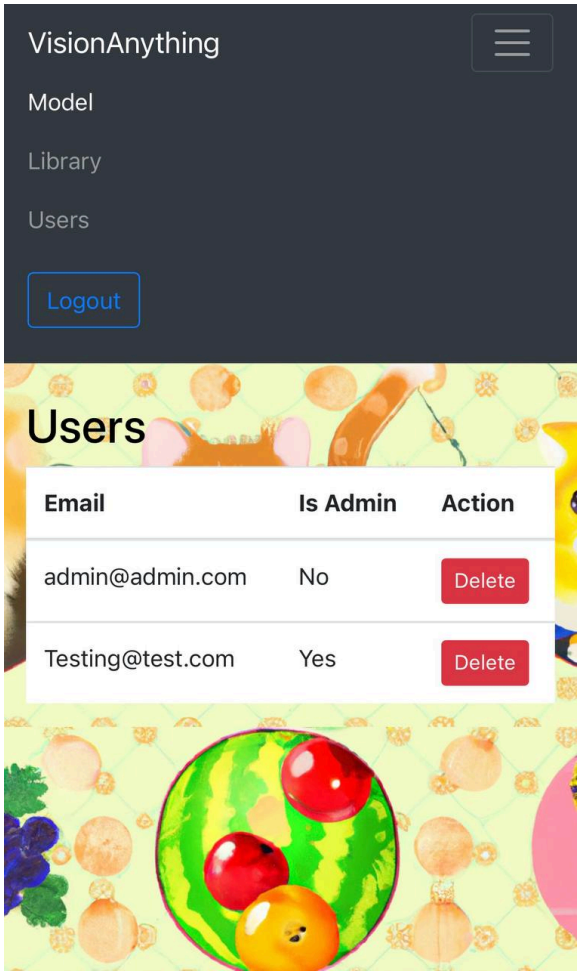
Heroku makes the processes of deploying, configuring, scaling, tuning, and managing apps as simple as possible so that we can focus on what's most important. This helps us develop VisionAnything without much hassle and helps us deploy the application once we are ready.

	
Landing Page: This is the landing page of VisionAnything, It is designed to be children-friendly.	**Registration Page**: Users can register here by registering their email and password Afterwards, it will be sent to the backend database Users can also register as an admin.

	
Logging Page: Once registered, the user will be routed to the login page, The user will then log into the web app	**Error Handling in Login Page**: If the user enters the wrong email or password that is not found in the database, a message will return saying that the email or password is incorrect.

	
Home Page: Once logged in, the user will be welcomed into the home page	**Navbar as User**: The user can log out or go to the library, the library is where the uploaded photos of the user go, user can view uploaded photos and download segmented photos

	
Uploading or Taking Image: The user can upload or take a photo from his or her mobile device and immediately segment the image.	**Selecting SAM Model**: Once logged in, the user can either select from 2 models, from MobileSAM to segment any image or Grounded-MobileSAM where it can segment an image and use a prompt to segment the image.



Admin Panel:

If the user is registered as an admin, the user can log in as an admin and have an additional tab to view all users that are registered into the web application.

The user can also choose to delete other registered users if need be.

4 EXPERIMENTS

In this section, we will go over the experimentation and implementation of our model and the technologies provided. All relevant information not found above will be listed below as well as our findings.

4.1 Environment Setup

- ### 4.1.1 Hardware

All experiments were conducted using the hardware configurations shown in Table I.

Table I Hardware Configuration	
Component	Model
CPU	AMD Ryzen 7 5800H
GPU	NVIDIA GeForce RTX 3060
RAM	32 GB

Table I: Hardware Configuration

- ### 4.1.2 Software

All experiments were conducted using the software packages shown in Table II.

Table II Software Configuration	
Software Package	Version No.
Python	3.10.13
OpenCV	4.6.0
PyTorch	2.2.1
TorchVision	0.17.1
Torchmetrics	1.1.2
Tensorflow	2.10
Flask	2.2.5
SQLite	3.41.2
SAM	https://github.com/facebookresearch/segment-anything

MobileSAM	https://github.com/ChaoningZhang/MobileSAM
Grounded-SAM	https://github.com/IDEA-Research/Grounded-Segment-Anything

Table II: Software Configuration

4.2 Datasets

For this project, we have used a total of 3 datasets. Each with different purposes for training the model such that we can detect and classify the images correctly.

Table III Datasets Used		
Dataset	Model	Use-Case
Microsoft-COCO	Grounded Dino	Used Grounded Dino as the model for Microsoft-COCO Also used to show what datasets we used.
Fruits Dataset (Images)	CNN (convolutional neural network)	The dataset used to train the CNN model
Cats and Dogs Images	CNN (convolutional neural network)	The dataset used to train the CNN model

Table III: Datasets Used

Microsoft-COCO is a large-scale object detection, segmentation, and captioning dataset. This dataset helps us show what datasets we used alongside the rest and provides a benchmark for algorithms to compare the performance of real-time object detection. The format of the COCO dataset is automatically interpreted by advanced neural network libraries. The figure below shows a standard benchmark for comparing the performance of state-of-the-art computer vision algorithms such as YOLOv4 and YOLOv7:

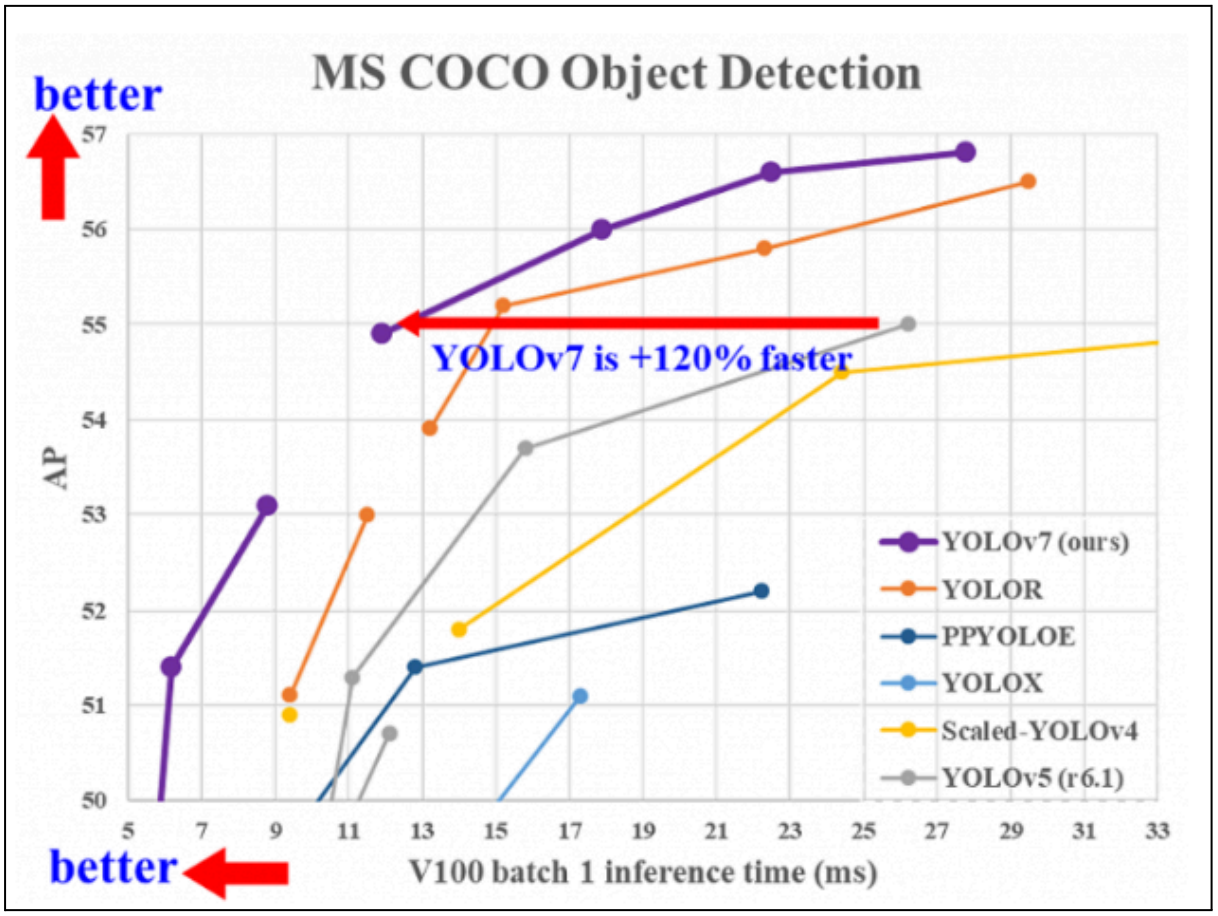
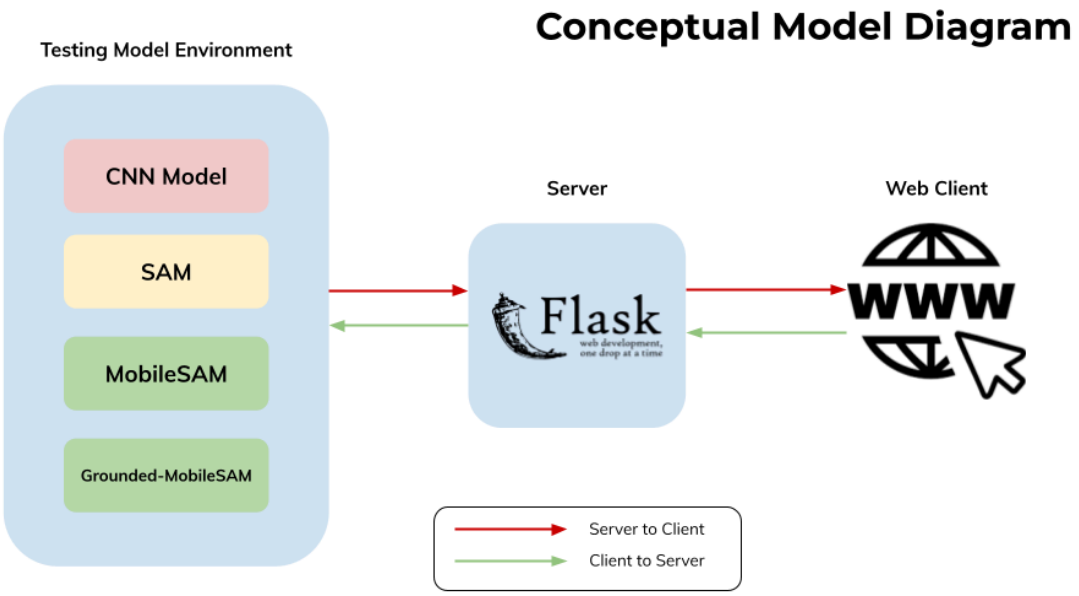


Figure 12. Microsoft COCO Object Detection Threshold (https://viso.ai/computer-vision/coco-dataset/)

The other datasets imported by Kaggle are used to train the CNN model to test the accuracy and labelling of the model. They will also be used as reference images for the model to classify any new or otherwise objects sent into the application.

4.3 Implementation Details



Done in Slides

Figure 13. Conceptual Model Diagram

Testing Model Environment:
The Tensorflow library was used to build the baseline CNN model. We then used both datasets of fruits and pets obtained from Kaggle's image classification model. The Segment Anything Model (SAM) weights were downloaded from GitHub and stored in the local machine for us to test segmented images. Afterwards, we compared SAM with MobileSAM which as mentioned in the model comparison, yielded better results for MobileSAM with faster segmentation with a small portion of the parameters contained at 50% lesser than the original SAM

We used the grounding dino library together with the mobileSAM model weights to create a robust model to predict and segment images and classify objects using prompts. Since the grounding dino is trained on the COCO dataset it can recognise images.

Server
POST and GET requests from the server and client are handled by the flask server.

Web Client
Web applications and web page routes are hosted on Heroku.

4.4 Execution of Experiment

- 4.4.1 Training the baseline

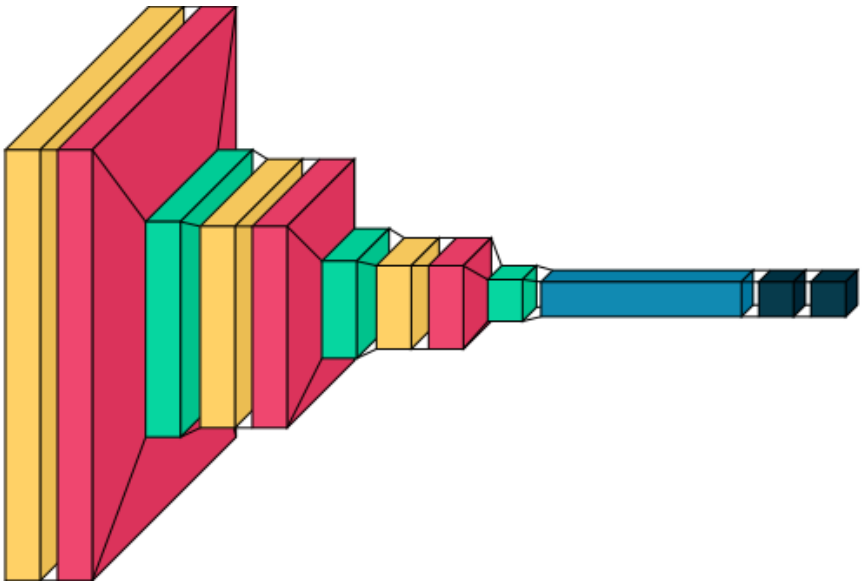


Figure 14. Layers (https://discuss.pytorch.org/t/image-classification-architeture/166327)
There are a total of 13 layers, with 11 hidden layers.

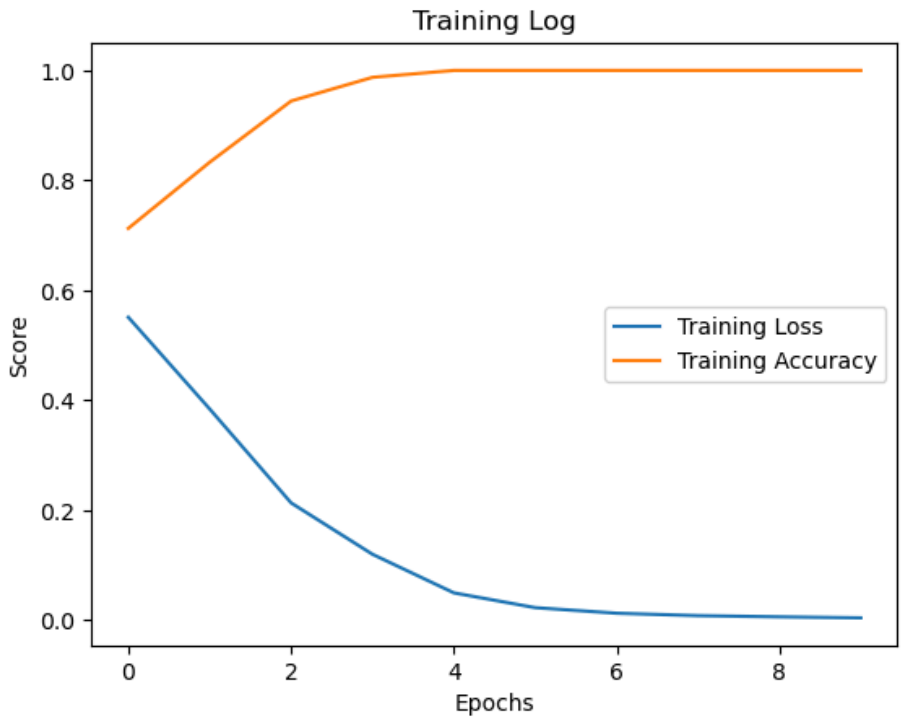


Figure 15. Training Log
The model was trained with 10 epochs.

- 4.4.2 Training on the evaluated model (MobileSAM)

```
Time taken to generate masks: 83.8721 seconds
```

```
print(len(masks))
print(masks[0].keys())
```

```
21
dict_keys(['segmentation', 'area', 'bbox', 'predicted_iou', 'point_coords', 'stability_score', 'crop_box'])
```

```
plt.figure(figsize=(20,20))
plt.imshow(image)
show_anns(masks)
plt.axis('off')
plt.show()
```

Automatic Segmentation parameters for MobileSAM took 83~ seconds to generate masks.

- 4.4.3 Training on pseudo-labels using Grounded-MobileSAM

The Grounded-MobileSAM can detect the cat with a confidence accuracy of 91%

5 CONCLUSION

5.1 Final Results & Discussion

To reiterate our objective, our primary goal was to create a weakly supervised framework that can produce results and perform similarly to their fully supervised counterparts. Through this project, we devised a system for pseudo-label propagation to alleviate the burden of acquiring pixel-level annotations by leveraging state-of-the-art foundational models. During experimentation, We tried to implement our framework, which includes training a fully-supervised baseline for comparison; training models on generated pseudo-labels using different loss functions; and incorporating CRF as a post-processing step to refine the output masks from our models. Table V below shows the final performance results for all models trained in this project:

Table IV:
Model Comparison

Model	Time
SAM (Semgent Anything Model)	2 minutes, 25 seconds
Mobile-SAM (Mobile Segment Anything)	1 minute, 23 seconds

This shows how Mobile-SAM generates and segments almost twice compared to the original SAM

The best-performing model was the one trained on CE with CRF as a post-processing step. The worst model was trained on Dice Loss, which contradicts our hypothesis of achieving a better model performance by using a loss function which accounts for class imbalance. Therefore, we chose the best-performing model (Weak CE + CRF) to be implemented into our framework. To conclude, we believe that while our framework is viable and performance is comparable to that of our fully-supervised baseline. However, we did not manage to match or beat it in terms of mIoU performance. This is reflected in our results as our standalone baseline achieved a mIoU of 59.20% and the result produced from our weakly supervised framework including CRF was 56.34%, leaving us a gap of 2.86%.

5.2 Masks Showcase

- 5.2.1 The Good Masks




From observation, we show some of the good-quality masks generated by the image model. We believe large prominent objects are easily segmented out by our framework. We also delved deeper into segmenting an object within the image based on this prompt: (Fruits: Rambutan prompt & Pet: Cat) and it segmented perfectly.

- 5.2.2 The Bad Mask


Upon further segmentation of other test images, this model deviated quite far from the actual image label it presented. This classifies the “bad masks” generated from the model due to too many objects and details. Further testing with provided prompts (Fruit prompt: watermelon (not segmenting watermelon properly & Pet prompt: dog) and it did not properly segment the object like the fruit and the dog.

5.3 Limitations

The web application works well and can be conducted as long as connected to wifi and a mobile device. However, Based on our work and findings, we found that even though we used MobileSAM with Grounding DINO, we found that the web application can be sometimes laggy. This may be caused by the high bandwidth and latency required on the web app. The web application is also limited to taking photos and segmenting the image in the mobile app, this might end up needing to host the model via API or in the cloud. To resolve this, we may need to experiment with live video segmenting due to the model being hosted on the web app and may need some time to conduct a GET request to activate and initiate the model.

Another limitation we found difficult was colours. It was difficult for the web application to segment anything to classify colours. Based on our findings, this is due to the web application being trained to segment objects and figures by using CNN logic. Since the Segment Anything Model (SAM) is designed for segmenting tasks, it helped focus on identifying and delineating specific regions or objects within the image. However, the model would not inherently recognize colours due to its primary function not focusing on that aspect/

These limitations mainly stem from bug issues and fixes that can be done with enough time. We can implement the fixes if we ever want to host the model on a live camera to conduct live image segmentation and identification. This ends up limiting the web application in being able to detect sensitivity in input placement and low-contrast environments.

5.4 Future Work

Additional work can be done to improve the current web application and image model. We believe that the best improvement would be to capture the images using real-time detection, classify other objects simultaneously or even make our model segment faster.

How we can implement those solutions possibly by implementing adaptive thresholding to improve object detection under varying lighting conditions by dynamically adjusting segmentation thresholds based on image analysis. We can also expand the object detection libraries so that we increase the variety and specificity of objects recognizable by the model by incorporating diverse datasets tailored to specific use cases or environments. We would then need to streamline the image processing pipeline to reduce latency and enhance performance, ensuring real-time responsiveness without creating new neural networks. These implementations can help make the model segment faster and classify objects more efficiently.

Our web application can be further enhanced by developing intuitive user interfaces that allow users to interact with the segmentation process, such as adjusting segmentation parameters or manually refining segmentation areas. These are some of the future works on this project that can be implemented. This web application is still open to new possibilities and we can even utilize cloud-based processing to handle complex segmentation tasks, enabling lightweight client-side applications to benefit from advanced computing resources.

Model	parameter	1 point prompt
SAM-B	136M	7.02s
FastSAM-s	11M	1.12s
FastSAM-x(default)	68M	2.99
MobileSAM	9.66M	1.43s

Figure 16. Model Parameters

Explore models such as the FastSAM-s (small) as its 1 point prompt time is 1.12s, which is shorter than MobileSAM by 0.31s.

6 REFERENCES

[1] Boesch G. “Segment Anything Model (SAM) - The Complete 2024 Guide - viso.ai.” Viso Suite. https://viso.ai/deep-learning/segment-anything-model-sam-explained/ [accessed Mar. 9, 2024]

[2] Chauhan, G. “Deploy your ML, scikit-learn code to production.” Deploy your ML, scikit-learn code to production. https://medium.com/@gauravc2708/deploy-your-ml-scikit-learn-code-to-production-d470899f9bf5 [accessed March 21, 2024]

[3] Grounded-SAM Contributors. “Grounded SAM: Assembling Open-World Models for Diverse Visual Tasks.” https://github.com/IDEA-Research/Grounded-Segment-Anything/blob/main/CITATION.cff [accessed Mar. 10, 2024]

[4] Grounded-SAM Contributors. “Grounded-Segment-Anything.” GitHub Repository. https://github.com/IDEA-Research/Grounded-Segment-Anything/blob/main/CITATION.cff [accessed Mar. 10, 2024]

[5] Jocher, G., & Zhang, C. “MobileSAM (Mobile Segment Anything Model” Ultralytics YOLOv8 Docs. https://docs.ultralytics.com/models/mobile-sam/#adapting-from-sam-to-mobilesam [accessed Mar. 10, 2024]

[6] Koue, K. “What is Segment Anything Model (SAM) & its Uses.” Sama. https://www.sama.com/blog/what-is-segment-anything-model-sam-its-uses [accessed Mar. 13, 2024]

[8] Meel, V. “What is the COCO Dataset? What you need to know in 2024 - viso.ai.” Viso Suite. https://viso.ai/computer-vision/coco-dataset/ [accessed Mar. 13, 2024]

[9] LIN, R. “Neural Network Concepts Explained — What Are Logits | by Renee LIN | Medium.” Renee LIN. https://reneelin2019.medium.com/neural-network-concepts-explained-what-are-logits-c39dd96b2e91 [accessed Mar. 20, 2024]

[10] Microsoft. “Microsoft COCO: Common Objects in Context.” Microsoft COCO. https://arxiv.org/abs/1405.0312 [accessed Mar. 13, 2024]

[11] ModelBit. “SAM - Segment Anything Model Guide: Revolutionizing Image Segmentation.” Roboflow. https://www.modelbit.com/model-hub/segment-anything-model-guide [accessed Mar. 14, 2024]

[12] Roboflow. “GroundingDINO Object Detection Model: What is, How to Use.” Viso Suite. https://roboflow.com/model/groundingdino [accessed Mar. 13, 2024]

[13] Saha, S. “A Guide to Convolutional Neural Networks — the ELI5 way.” Saturn Cloud. https://saturncloud.io/blog/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way/ [accessed Mar. 9, 2024]

[14] Skalski, P. “How to Use the Segment Anything Model (SAM).” Roboflow Blog. https://blog.roboflow.com/how-to-use-segment-anything-model-sam/ [accessed Mar. 13, 2024]

[15] Zhang, C., Qiaoyu, & Han, d. s. “This is the official code for the MobileSAM project that makes SAM lightweight for mobile applications and beyond!” Viso Suite. https://github.com/ChaoningZhang/MobileSAM [accessed Mar. 9, 2024]