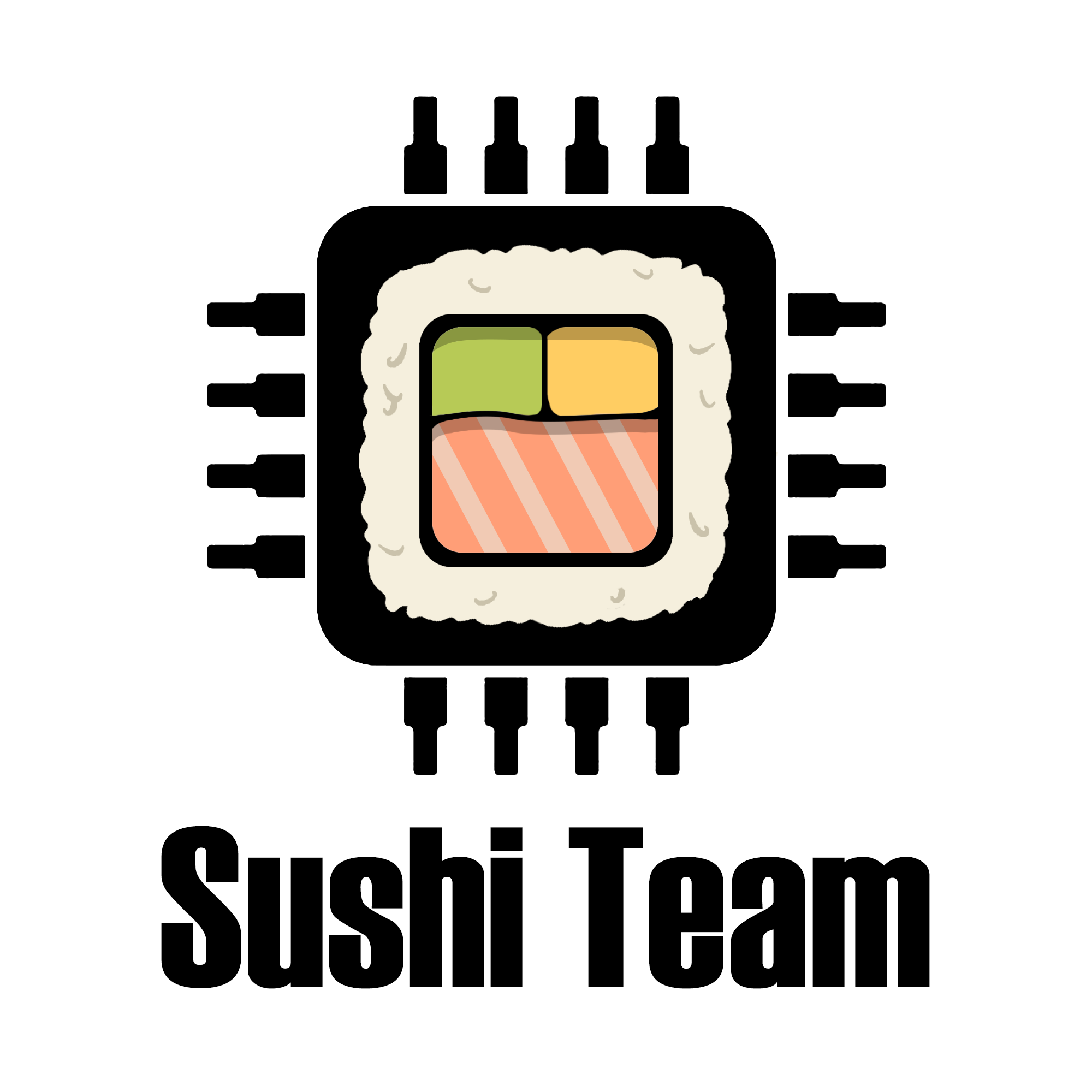
**Prototype #1 Meeting Agenda**

When & Where: March 29th, 2022 ~ 12:00pm - 1:00pm @ Zoom

Attendees: Dr. Jared Macshane, Dr. Shaunn-inn Wu, Sushi Team



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Beginning/Formal Greetings (Noah - Team/Scrum Leader)**

**Current Requirements / Network Diagram (Jordan - Documentation/Training)**

* Develop a system to receive and process Google Street images from a user (the Web App team).
  + Google Cloud/ML-Server Instance | TCP connection
* Develop a Machine Learning algorithm that must be able to detect instances of litter in the received Google Street images.
  + fixIT Dataset | Roboflow | Converted Dataset | YOLOR ML-Algorithm
* Once the litter instances have been identified, the system must then process and output the results into a suitable format, like JSON, for the user (Web App Team).
  + Processed Google Street image results
* Once the results have been made, the system must then return the processed data to the user (Web App Team).
  + TCP connection

**Project Management (Juan - Project Manager)**

* Recap estimates vs actuals, for both Hours & Costs. Costs based on a $23 hourly wage. An additional Estimated cost for access and utilization of a server is set to be $200.
  + Phase 1 - Going over Mask R-CNN Model and gaining a better understanding of machine learning concepts.
  + Phase 2 - Looked into the YOLOR model and switched to a new approach of a 2 phase model, where we first detect the litter in Google Street images, then, conditional, categorize that litter.
* Recap main tasks within Prototype #1.
  + Categorizing Litter: Annotation Software Development
  + Detect Litter in Images: ML Algorithm Development
  + Detect Litter in Images - Python Script Development
  + ML Training
  + ML Testing & Algorithm Processing
  + ML Algorithm/Script Output Adjustments

**List of Deliverables (Miguel - Programmer)**

* Provide a functional litter detection algorithm that consists of:
  + Getting an input of Google Street images
  + Processing them through the litter detection
  + Output the data, consisting of images with bounding boxes, and a JSON file, signifying the total amount of litter detected in the picture.
  + Send the output data to the Web App.
* To achieve this, we are using the YOLOR model as the engine for litter detection.
* Also, our repository, which will contain the code, documentation, and eventually the fully functional, modified YOLOR model.

**Analysis & Design (Keith - Programmer)**

* YOLOR is a unified network that joins explicit and implicit knowledge.
  + The benefits of this model are that it’s lightweight and has a higher FPS with greater accuracy than similar models.
* We began training utilizing a previous semester’s Google Street Image dataset. This dataset was already annotated with bounding boxes. The dataset contains 2592 images and 9371 annotated objects.
* Utilized a team Google Cloud Instance to host our training, detection, and storage.
* We also set up a Weights and Biases account and connected our model to it. This online tool allows us to monitor our training in real time and compare the effects of different hyperparameters.

**Prototype #1 Functionality (Noah - Team/Scrum Leader)**

* How YOLOR can work with user provided images, to process them through object detection.
  + YOLOR's command line functionality.
    - DEFAULT values for certain variables, and how they can be changed & utilized.
    - Specify file paths for both the Source and Output of the detection code.
    - All is done prior to detection itself, allowing us to implement the TCP connection in the same portion of code.
* TCP pipeline, from the Client (Web App team) to the Server (ML Algorithm), through Socket Programming.
  + Start sending multiple images/files to Server.
  + Create a new path in the YOLOR directory to not only hold them, but to also define it as the Source for the litter detection.
  + Once detection is done, the Server will store them in the specified default Output – inferences/output, in the YOLOR directory.
* TCP pipeline, from the Server (ML Algorithm) to Client (Web App team), as well as modifications to the code for flexibility.
  + The processed results - copies of the provided images, with bounding boxes around the detected objects, and a JSON file.
  + The Client code creates a path to receive the processed results, ML-Output.
  + Server running through a loop - ready for multiple batches during its use.
  + Additionally, to avoid complications of GPU usage, unnecessarily used space, and re-doing already processed images again, paths will be removed, contents and all, and the model will unload, for each run.

**Analysis & Design - Issues/Solutions (Keith - Programmer)**

* YOLOR:
  + Built for speed - only able to achieve an AP value of 16% after 900 epochs.
  + Limited to bounding box annotations.
* fixIT Dataset:
  + Poorly annotated, and poor resolutions, Google Street image dataset.
* Solutions
  + Improved street image collection parameters to maximize the litter space resolution.
  + Use Roboflow to reannotate the existing dataset to increase the quality of these annotations. Roboflow will also allow us to validate annotations made on the new dataset prior to adding them into our model.
  + A different machine learning model called YOLOv5.
    - Polygon to bounding box translation
    - Contains a better auto-anchor utility
    - Higher precision on the same data as YOLOR, though at a slightly lower FPS rate.
    - An export utility is also included which will allow us to move our weights to different models if needed.

**Final Thoughts/Goals (Miguel - Programmer)**

* Images sources
  + In addressing the dataset limitations, we tried to look for different sources for street view images, to get better resolutions.
  + However, we decided to stick to Google Street images, because it provides the most consistent coverage of street view images.
* Datasets
  + In addressing the poor annotations from the previous dataset, we first tried to find a better set of annotations for litter detection. Unfortunately, they were not compatible with the Google Street images.
  + Now, we are addressing the poor annotations from the previous dataset by modifying their annotations to increase the quality and accuracy.
    - Again, we are using Roboflow to standardize the annotation process, in order to avoid misunderstanding and delays. It gives a good interface to keep tracking on the annotations or images in real time. (An account is needed)
  + Additionally, we are requesting a new dataset of approximately 10,000 Google Street images, from the Web App team, to train with for better results.
    - They will utilize the recommended Pitch and FOV (field of view) parameters to get better angles and resolution on the Google Street images.
    - Dr. Schultz provided students will then annotate them.

Confirm and agree our current direction, and next direction for the coming phase:

* Litter Detection -> Should we continue towards refinement?
* Litter Categorization -> Migrate to a different model for categorization?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**As a team - communicate with Jared on further requirements.**

**Summary**

**Meeting adjourned!**