Automatic Labeling/Annotation of Image and Video Data for Feature Extraction

Chief Architect - Jeff Kinard

Project Manager - Sam Hassebroek

Meeting Facilitator - Michael Boyle

Test Engineer - Dylan Hodge

Report Manager - Dylan Smith

Meeting Scribe - Mark Endeshaw

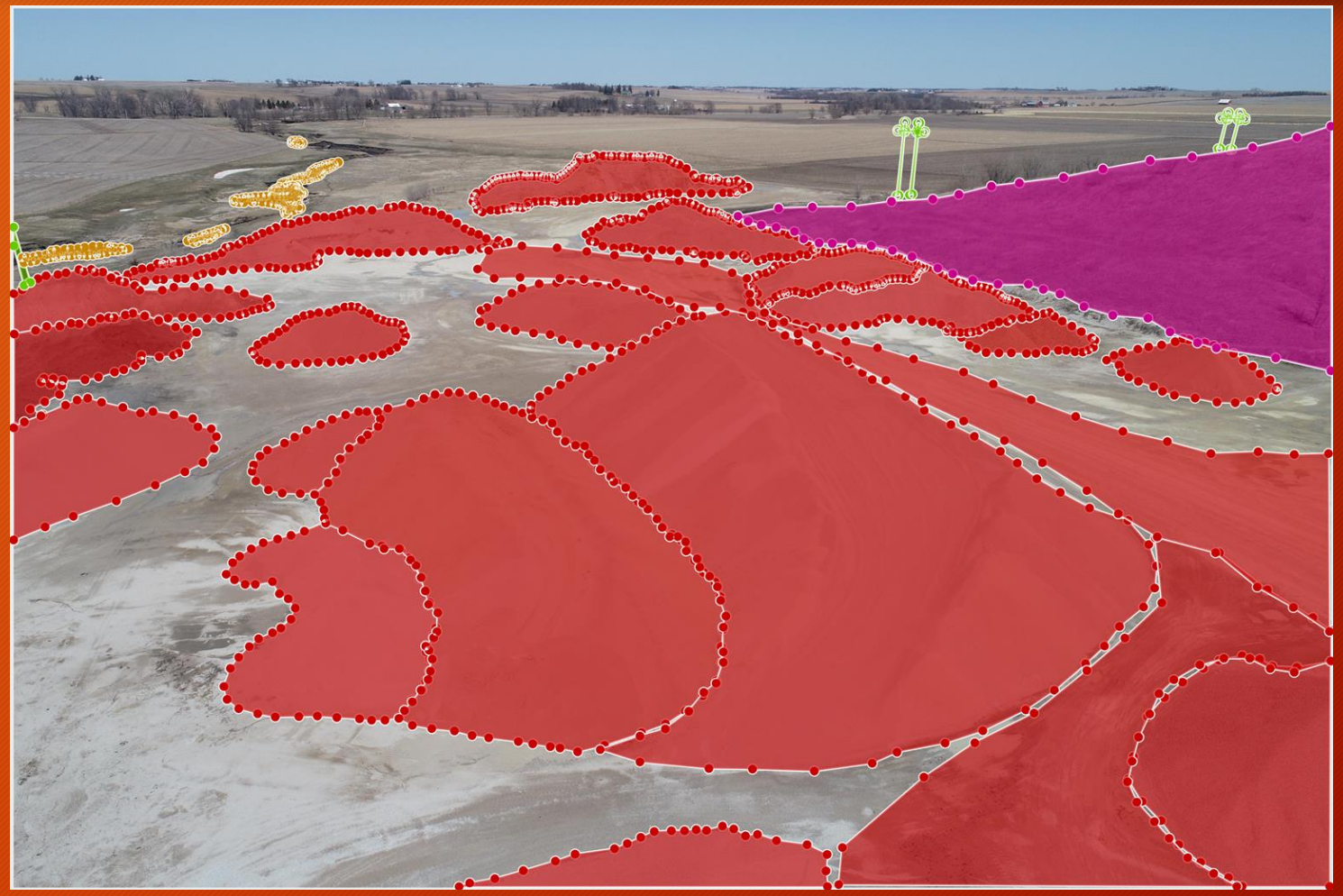
Client - Dr. Ali Jannesari

Advisor - Chandan Kumar

Project Vision

- Most Computer Vision (CV) projects revolve around training a Machine Learning (ML) model to classify images or objects in images. This requires an annotated image dataset that, historically, must be carefully compiled by hand.
- Our project seeks to bypass the inefficiency of annotating image data by hand through an automated approach.
- We will be developing a ML algorithm to take a set of unlabeled images and output a COCO-formatted annotation file with all the annotations and labels for object masks found within each image in the dataset.

Example Annotated Image



Functional Requirements

- Algorithm to automatically annotate all objects from any image
- Extension of the algorithm to annotate objects from video
- Further extension to include object masking
- Backup algorithm to annotate objects missed by the main algorithm
- Method of validating the result for each algorithm

Non-functional requirements

- Well-documented code and technical documents
- Ensure same dataset produces similar output if rerun (consistency)
- Minimize loss as much as possible
- Test out any possible bugs
- Use cutting-edge techniques and methodology

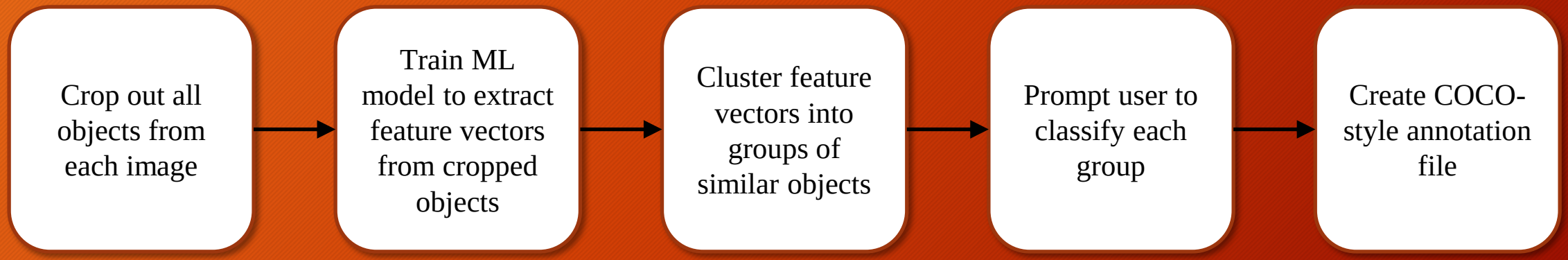
Technical and/or other constraints

- Since this is a research-based project and entirely theoretical, we have the ability to gain any compute resources we may need to complete this project.

Market Survey

- Current methods include similar feature extraction using CNN’s
  - Lack contrastive component
- Other methods involve allowing a CNN to extract the blobs
  - Only objects model is trained on will be accurately extracted (<1000 classes)

Conceptual Design Diagram



Components

I. Image Processing
   - Edge Detection
   - Blob Extraction
   - Cropping

II. Contrastive Learning
   - Build 2 Mask R-CNN models in parallel (w/o head)
   - Develop loss function for training
   - Backpropagate loss to both networks

III. Dimensionality Reduction
   - Use t-SNE or similar algorithm to reduce dimension from 1000 to 2 (or 3)

Components (cont.)

IV. Clustering
- K-based clustering algorithm

V. Classification
- Prompt user for labels
- Associate labels to masks accordingly

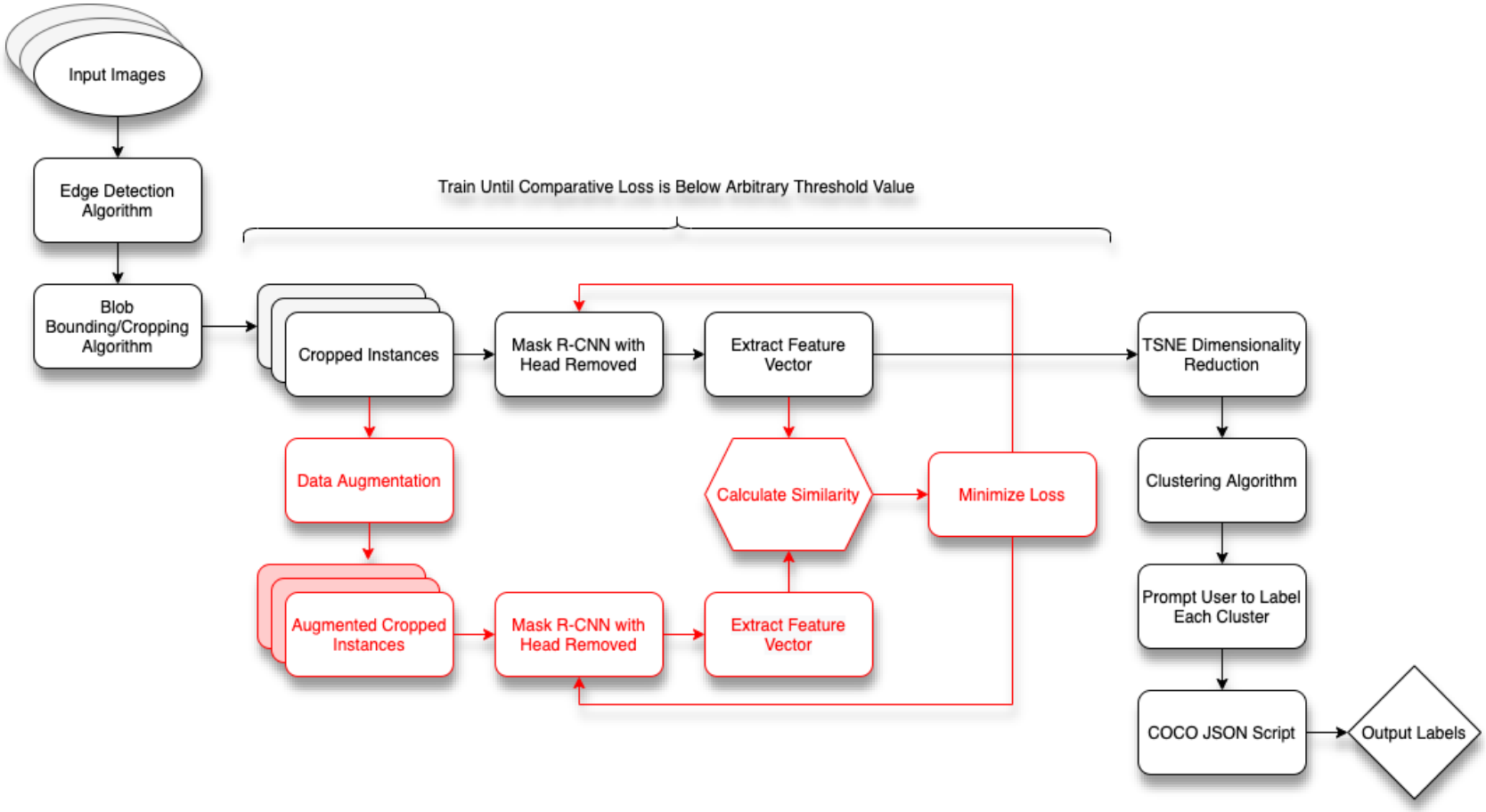
VI. Labeling
- Script to convert masks and labels in memory to COCO file

Risk Mitigation Plan

- Our project is using many pre-built parts helping to minimize our work and thus our potential error.
- We conducted research to build upon what is currently done and known.
- Weekly meetings with our advisor are held to ensure we have correct ideas and are on the correct track.
- Benchmarking against manual and pre-trained data to ensure that our goals are met.

System Design

- Our design is built upon a ML-based pipeline and written in python
- We will use the PyTorch framework to build, train and output the CNN used for feature extraction
- A pretrained Mask R-CNN model (with classifier layers removed) will serve as the backbone architecture for our feature vector extraction

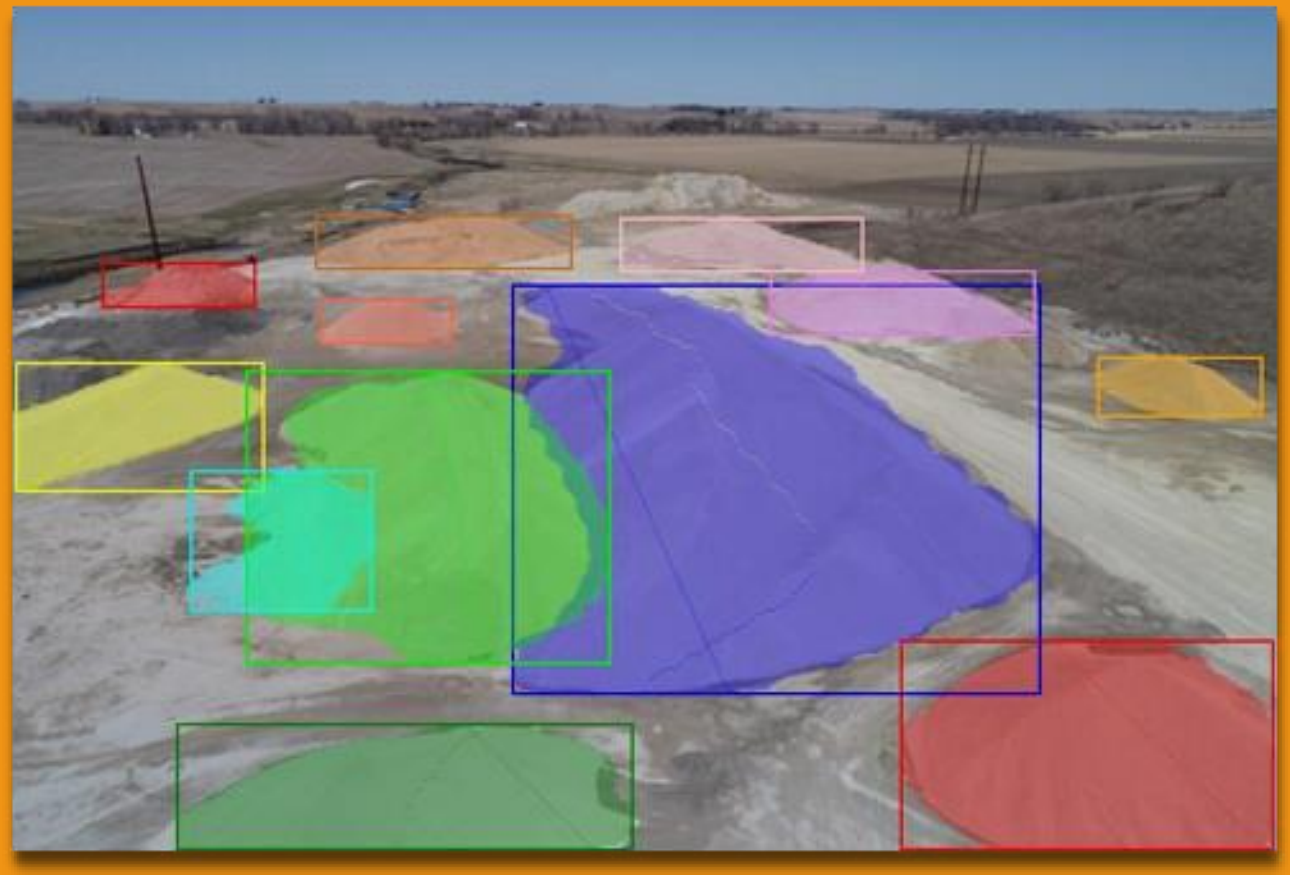


System Diagram

Prototype Implementations

- We currently do not have a prototype for the proposed design
- We have tested our ground-truth dataset using Mask R-CNN and Faster R-CNN models
  - The Faster R-CNN in particular saw mAP scores from 0.7-0.8 @ IoU=0.5
  - The Mask R-CNN had lower scores (<0.5 mAP), but looked good upon visual inspection
  - Discrepancies in accuracy most likely occur due to limited sample size of most classes

Mask R-CNN

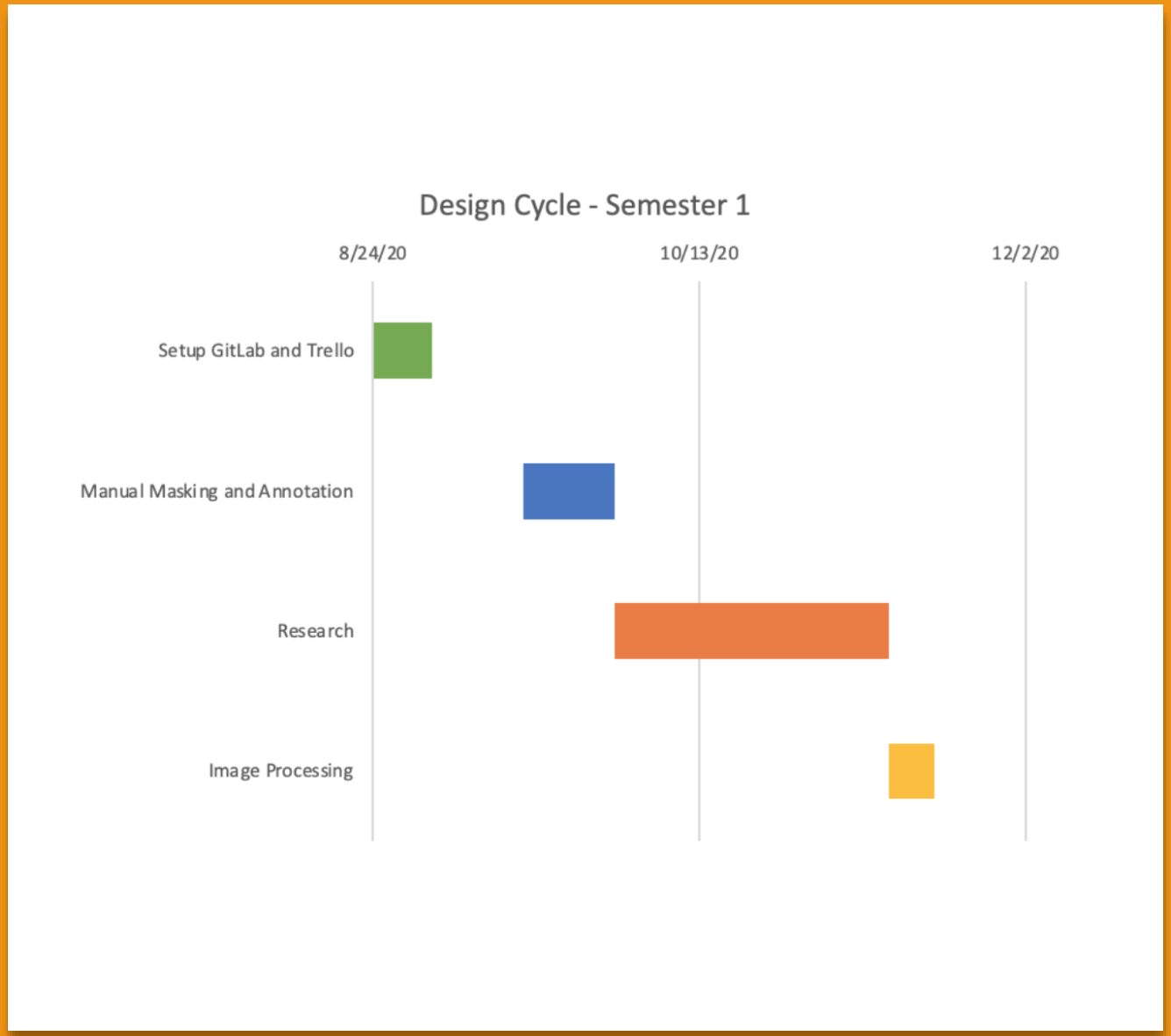


Faster R-CNN

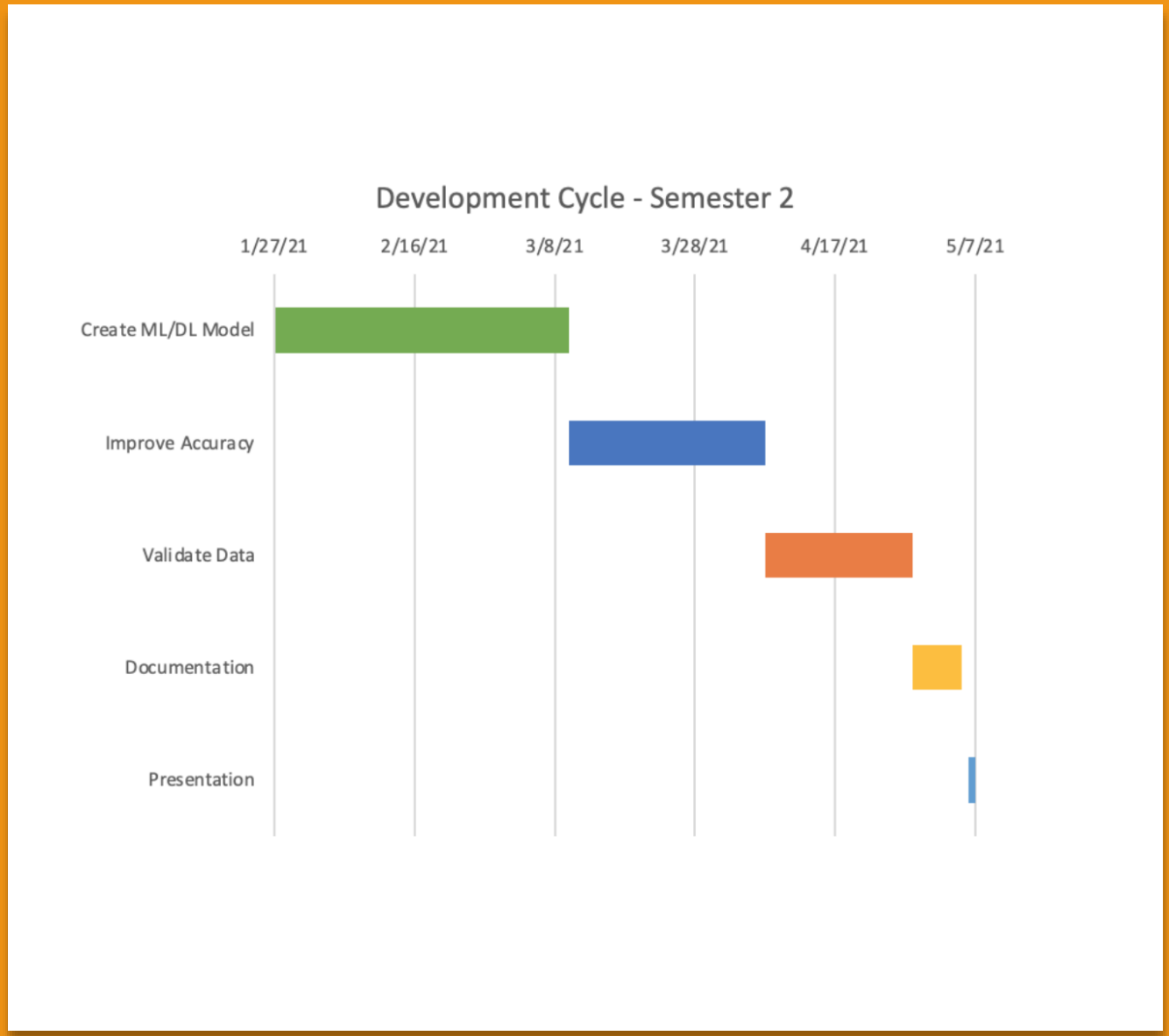
Future Prototype Implementations

- All the iterations will serve as prototypes as our project will continually change and evolve as we work.
- Each implementation will be tested and evaluated for its strengths and weakness.
- New Prototypes will be built upon the previous ones to correct problems and to improve the overall product.

Fall Semester Schedule



Spring Semester Schedule



Task-wise Milestones

- Edge Detection/Blob extraction Algorithms
  - Extracts object blobs within 90% accuracy when compared to ground-truth
- Contrastive learning model
  - Minimizes loss to as close to 0 as possible
- Clustering algorithm
  - Create consistent groups of feature vectors
- Labels-to-COCO script
  - Create correctly-formatted COCO annotations file
- Documentation
  - Ensure code is documented in its entirety

Overall milestones

- Assemble entire pipeline with tested components
- Get accuracy of model to >0.9 preferably (or as high as possible)
- Ensure model works on new dataset
  - Requires visual inspection

Testing Plan Description

- Since our project is largely research-based, testing is difficult
- Test to make sure the pipeline runs successfully
- Ensure the output of the pipeline is in our required format
- After running the pipeline, test the output against standard accuracy metrics
- Test to see how well our pipeline compares to other approaches of the same problem

Unit/Interface Testing

- For unit testing, we will manually observe each individual step of the pipeline and ensure that each step is performing as intended
- For image processing, we will observe the output to confirm images are processed and cropped correctly
- For the contrastive learning model stage, we will compare several feature vectors that represent similar object instances and ensure that the vectors have similar data
- For dimensionality reduction, we will transform the data created from the vectors and visualize it to see if it is possible to create clusters from it

Unit/Interface Testing (cont.)

- When the vectors are clustered, we will look at them to make sure objects in a cluster are clustered correctly.
- When the objects are labeled, we will compare them against our ground truth labels to ensure that they have been labeled correctly

Acceptance Testing

- After running through all the steps, we will make sure the pipeline ran as intended
- We will compare the results of the output to pre-trained algorithms that we have developed
- We will use mAP (Mean Average Precision) scores to determine the accuracy of our developed pipeline

Conclusions

- We have gone through extensive research on image annotation
- We have generated the ground-truth dataset
- Our design has been finalized for implementation next semester.
- Next semester we will build out and test our project against the baseline we have established