NAIC AI TECHNICAL

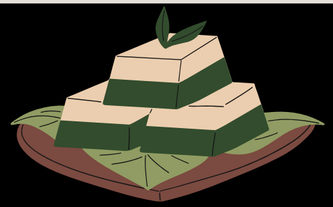
TEAM

CANTBYTEUS

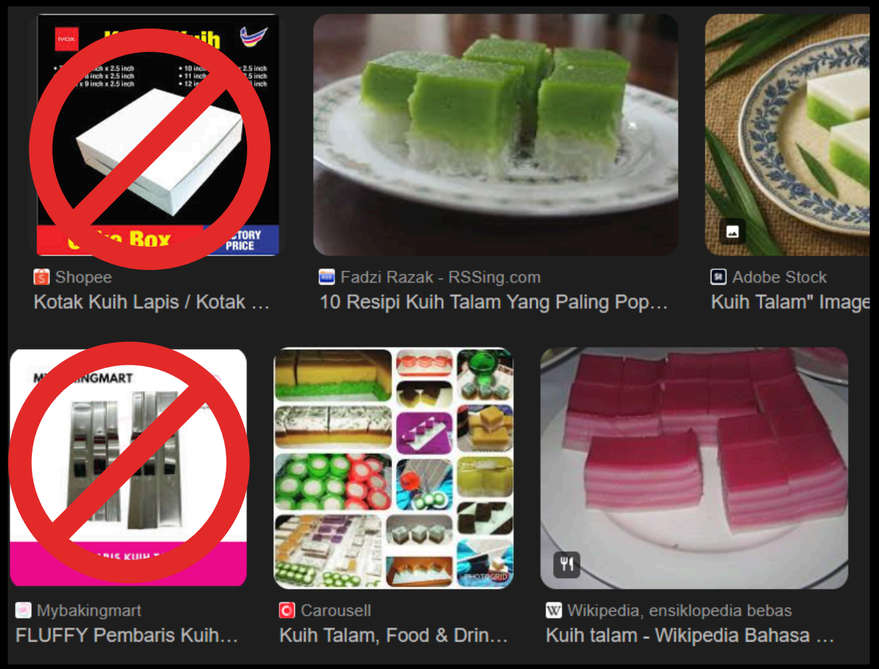
Ong Chong Yao
Ng Tze Yang
Terrence Ong Jun Han
Chin Zhi Xian

github.com/chong-yao/naic

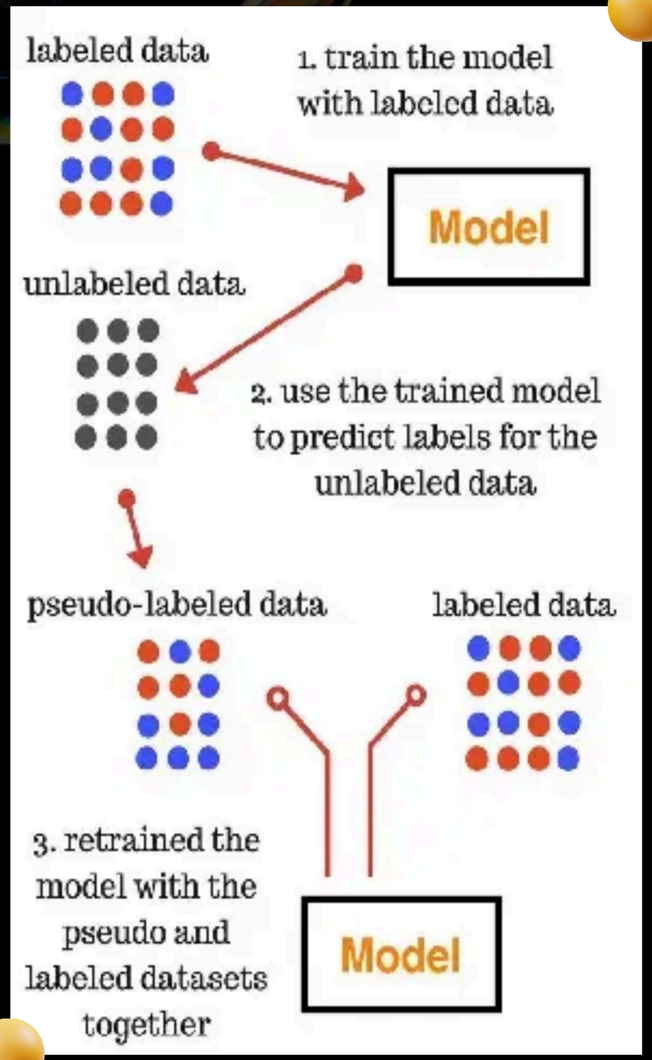
DATA COLLECTION PROCESS



- Scraped ~2500 images per class from Bing and Google and pre-existing datasets
- took photo physically with different brightness, and those that were bit
- Removed duplicates by computing tensor differences between images
- Manually filtered out inoperable images:
- unrelated
- corrupted
- low quality



PSEUDO-LABELLING TECHNIQUE



WHY?

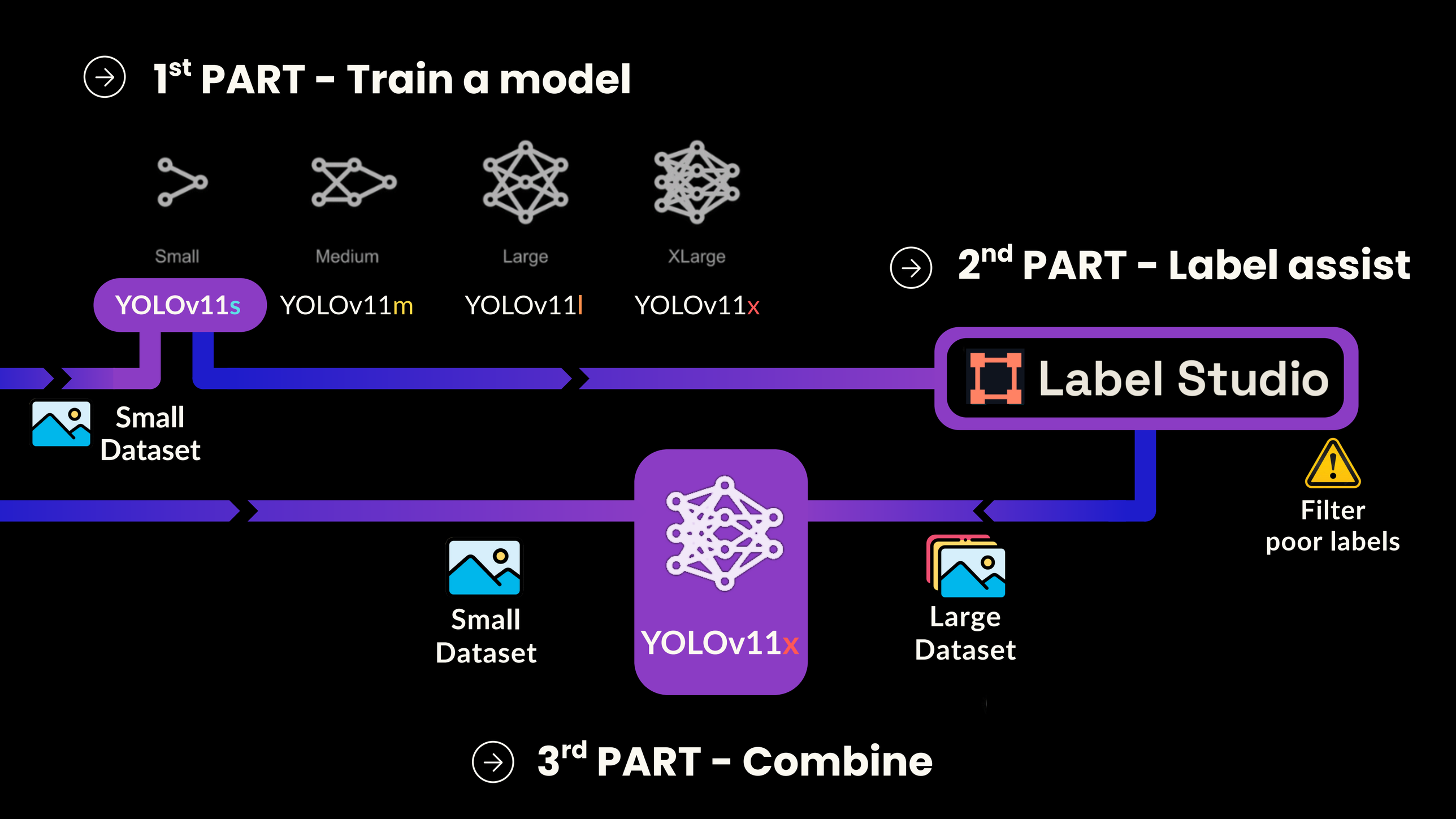
- Large dataset to manually label all of them

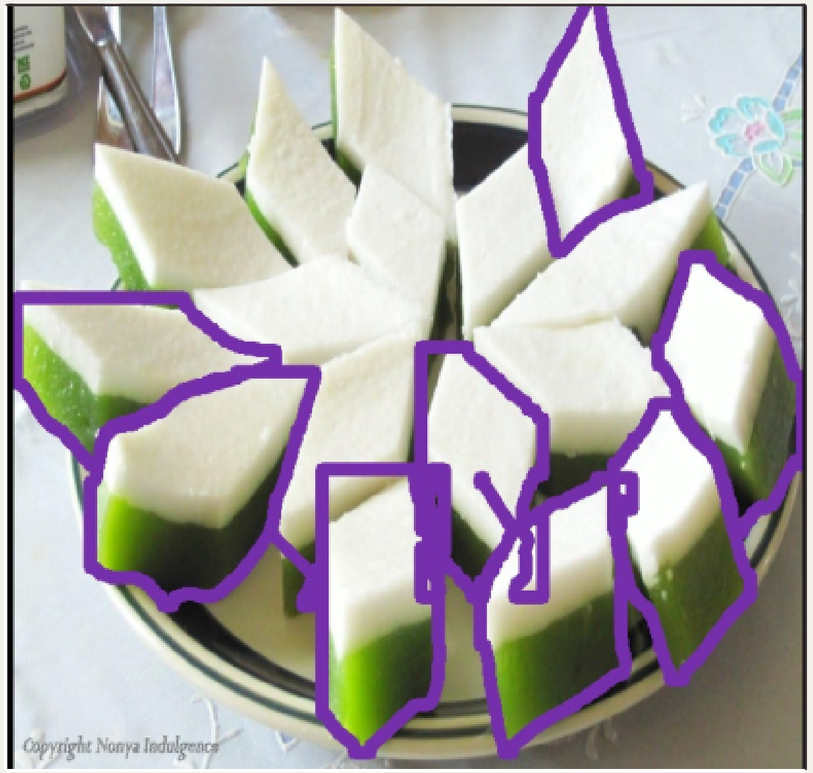
BENEFITS

- Assist in automating parts of the process

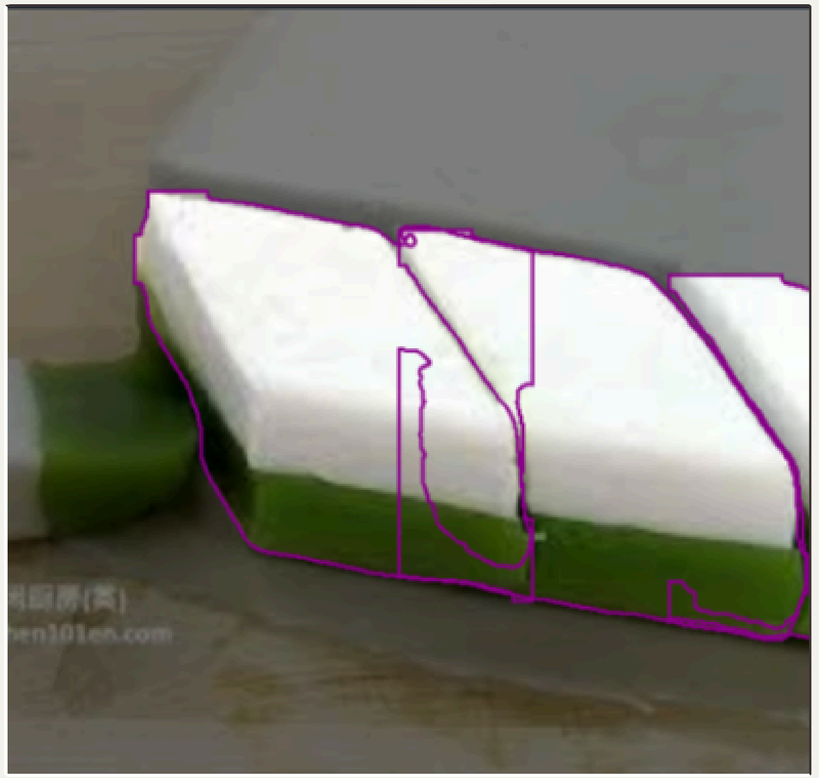
OVERVIEW (3 Main Parts)

1. Train model (with small dataset)
2. Use model to assist in labelling larger dataset
3. Combine all labelled dataset for final model


1st PART - Train a model
Small
Medium
Large
XLarge
YOLOv11s
YOLOv11m
YOLOv11l
YOLOv11x
2nd PART - Label assist
Label Studio
Small Dataset
Filter poor labels
Small Dataset
YOLOv11x
Large Dataset
3rd PART - Combine

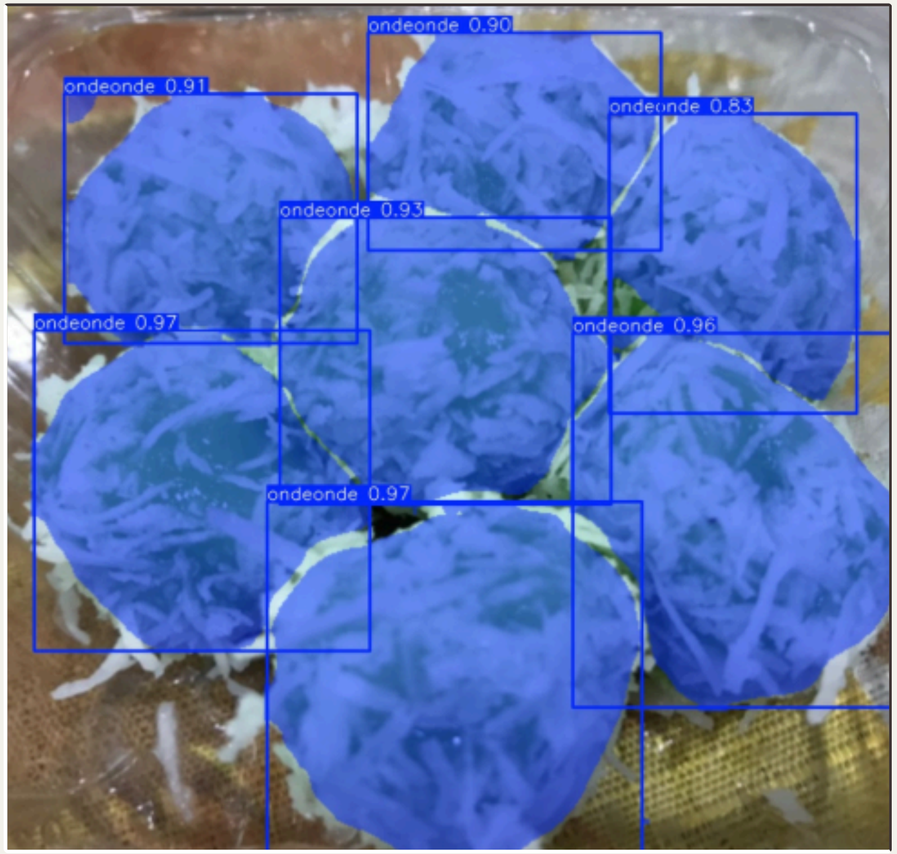


REJECT #1



REJECT #2

REJECT #3



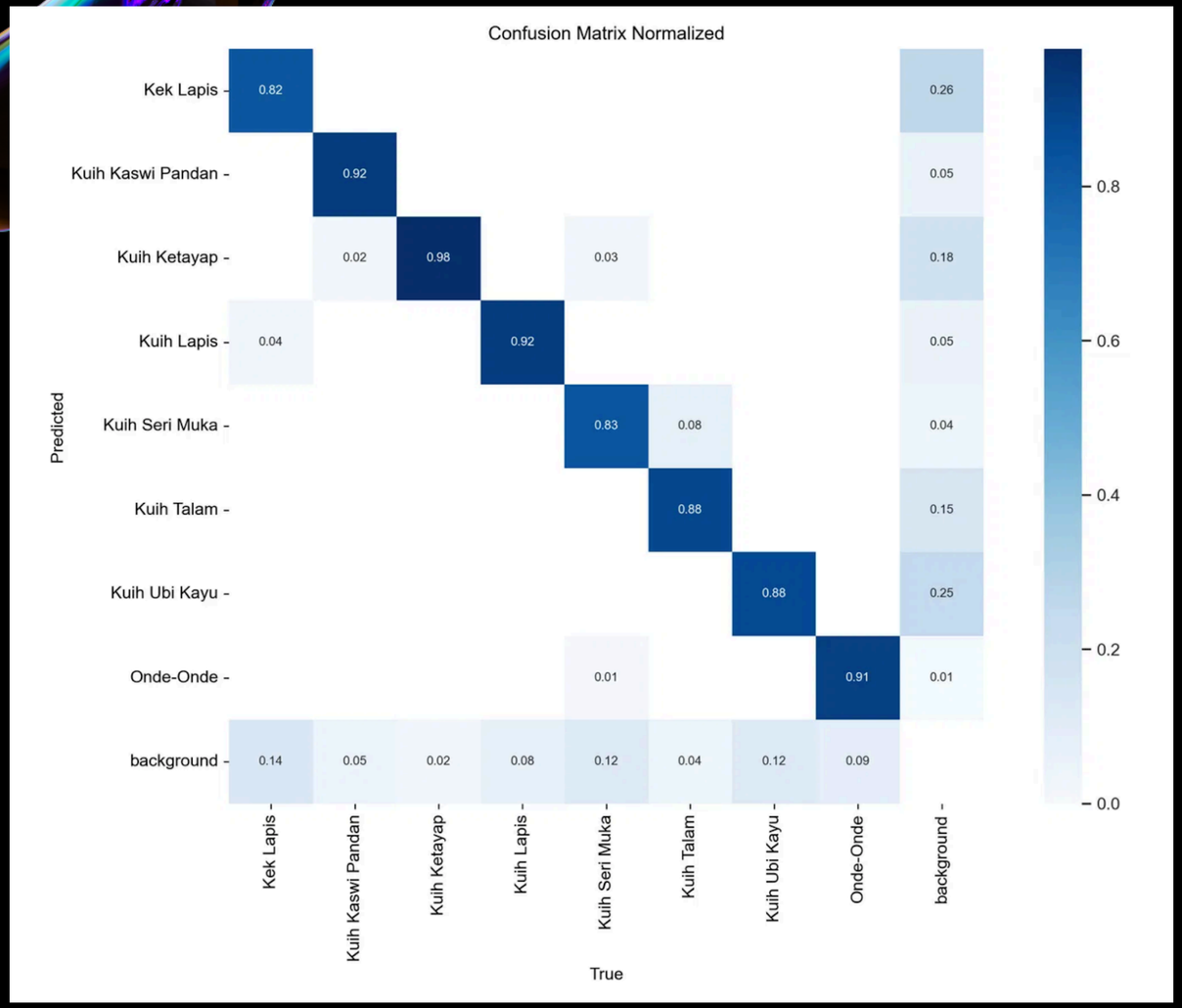
ACCEPT #1

DATASET VISUALIZATION

- Final dataset: 98 images per class with perfect annotations
- Different mask colors represent 8 different kuih classes
- Split: 90 images for training, 8 for validation per class
- Augmentation tripled dataset size while preserving color integrity

Final dataset: 784 images ((90 + 8) × 8 classes)

CNN SEGMENTATION MODEL



Segmentation improved classification by focusing on kuih itself

Designed custom YOLOv11x-seg model to increase kernel size, width, depth, and channels

Model then trained on COCO 2017 dataset first, then only on the kuih dataset

Near-perfect confusion matrix for the 8 kuih classes

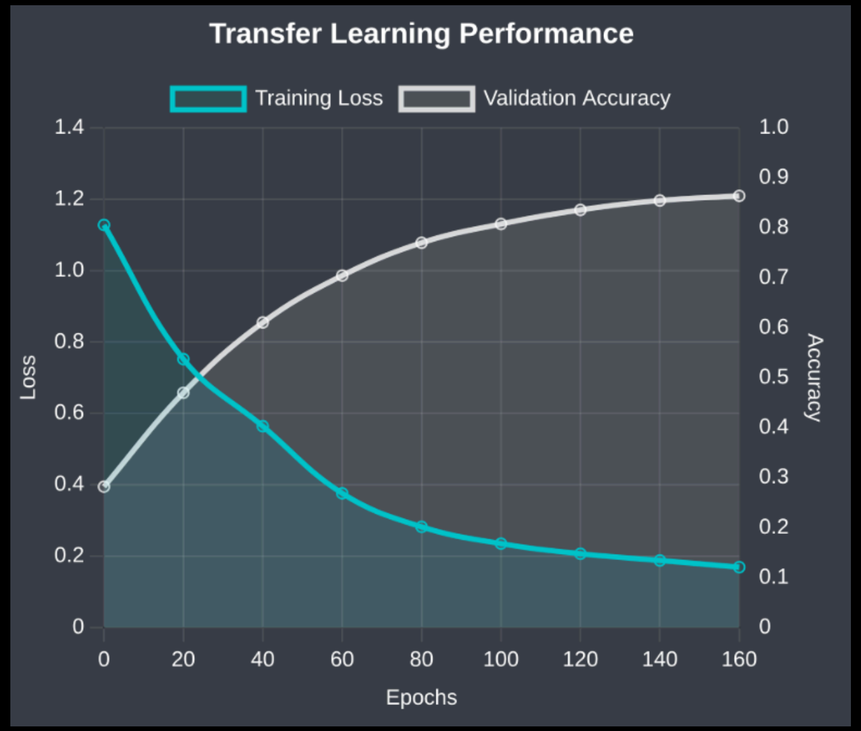
MODEL DEVELOPMENT PROCESS

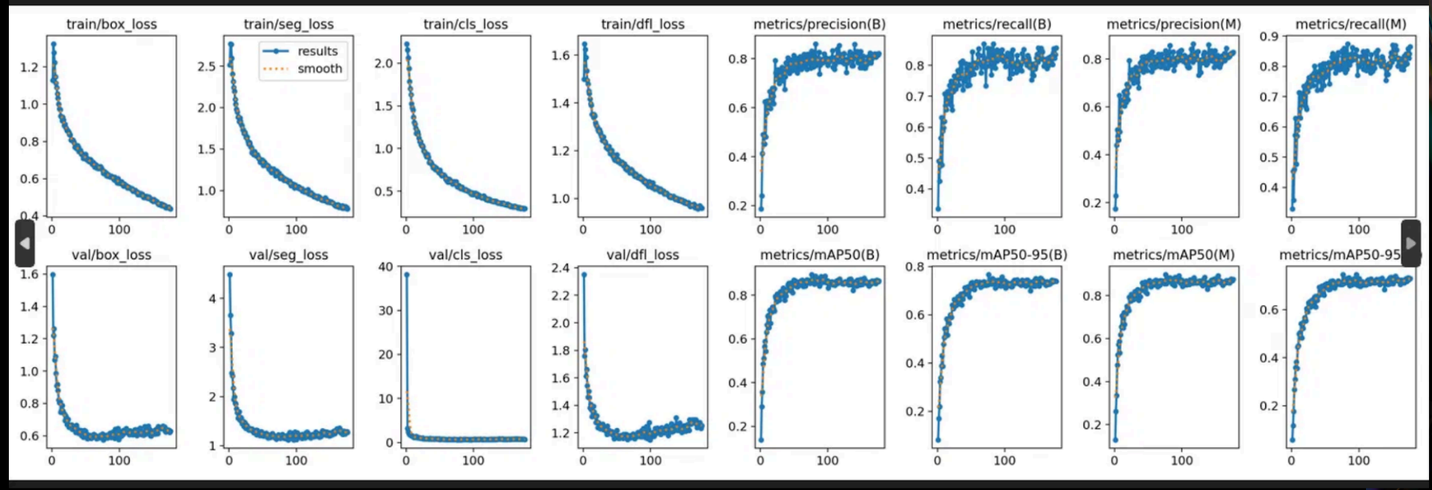
Used Pytorch with Ultralytics library and CUDA acceleration

Modified YOLOv11-seg configuration with additional attention layers

Pre-trained on COCO 2017 dataset to prevent overfitting on small kuih dataset

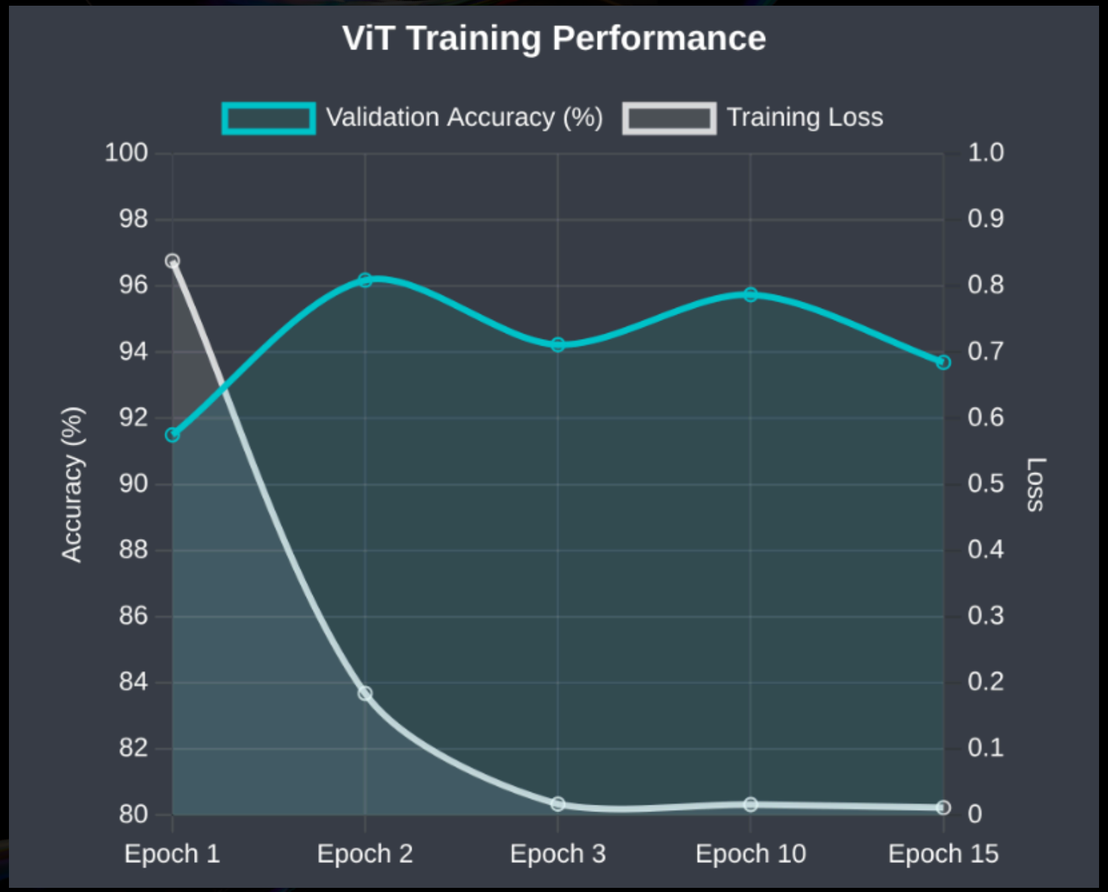
Normalized image exposure for consistent light balance during inference





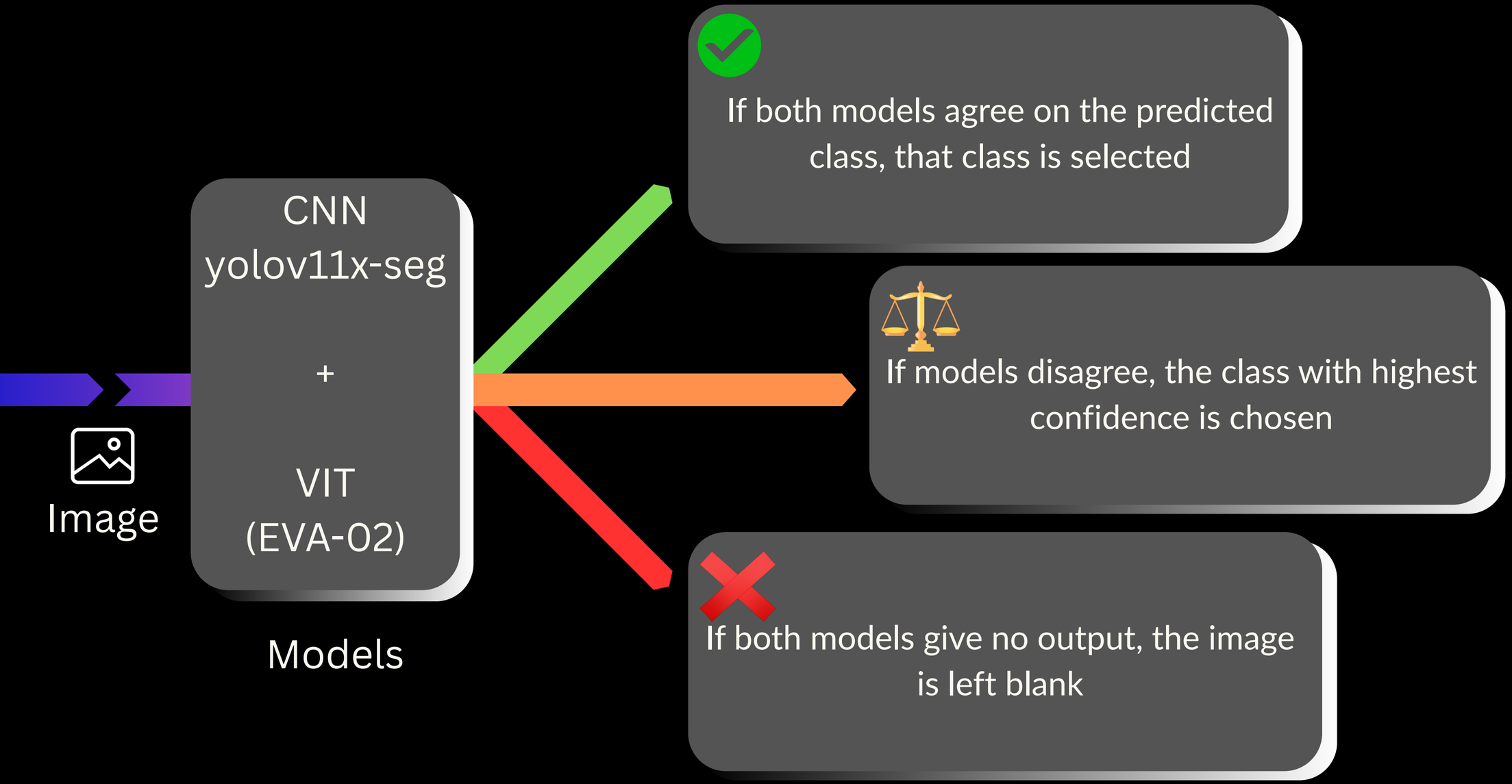
VISION TRANSFORMER

```
c:\Users\ochon\.conda\envs\naic\Lib\site-packages\tqdm\auto.py:21: TqdmWarning: IProgress not found. Please update jupyter and ipywidgets. See ht
  from .autonotebook import tqdm as notebook_tqdm
Classes: ['Kek Lapis', 'Kuih Kaswi Pandan', 'Kuih Ketayap', 'Kuih Lapis', 'Kuih Seri Muka', 'Kuih Talam', 'Kuih Ubi Kayu', 'Onde-Onde']
Epoch 1/10: 100%|██████████| 270/270 [01:23<00:00,  3.23it/s]
Epoch 1: Train Loss = 0.8603
Epoch 1: Val Accuracy = 91.89%
Epoch 2/10: 100%|██████████| 270/270 [01:25<00:00,  3.14it/s]
Epoch 2: Train Loss = 0.1891
Epoch 2: Val Accuracy = 96.68%
Epoch 3/10:   4%|▍         | 10/270 [00:17<01:54,  2.27it/s]
```



- Splits images into patches and transforms them into tokens
- Self-attention mechanism captures relationships across entire image
- Used EVA-02 model pretrained on ImageNet 22k for transfer learning
- Fast convergence (loss plummeted after only a few epochs) but required careful monitoring to prevent overfitting

ENSEMBLE APPROACH
HOW IT WORKS



WHY ENSEMBLE?

Feature	YOLOv11x-seg (CNN)	EVA-02 (ViT)	Advantage of Ensemble
Speed	✅ Real-time: fast	❌ Slower	⚖️ Good balance of speed & accuracy
Local Feature Detection	✅Really good	⚖️Moderate	✅Strong fine detail preservation
Global Understanding	⚖️Moderate-limited	✅Really strong context awareness	✅Captures both local and global information
Robustness	❌Sensitive to variation	✅More robust	✅More stable prediction overall

CantByteUs

THANK YOU

for your time and attention

github.com/chong-yao/naic