YOLO!

Anabelle Chang (anabelle), Christine Kwon (cmkwon), Christy Lee (christyl)

Advisors: Professor Olga Russokovsky, T.A. Fangyen Wei

INTRODUCTION

Goals

- To create an object detector that detects Grab n' Go items from the Frist Gallery that can be expanded into a price calculation application which determines the total price of the items present in an image.
- To analyze how YOLO's algorithm responds to different numbers of classes, different weights generated in training, and various challenges of inconsistency which are common in everyday objects; this will allow optimization in both detection and classification in new datasets.

Related Works

- Yi, Zhang, et al. "An Improved Tiny-yolov3 Pedestrian Detection Algorithm." Optik, Urban & Fischer, 13 Feb. 2019.
- Amazon Go
- 85°C Bakery Autonomous Checkout

Challenges

- Object Color Variation
- Object Logo Distinction
- Object Shape Distortion

METHODS

Overview

- Our dataset was self-curated with Google image scraping and manual photography at Late Meal, then annotated using VoTT. This allowed us to train a tiny YOLO v3 detector that is customized to items found in the Frist Gallery and can be a practical application of computer vision for Princeton University students.

Dataset

- *Preparation:* use of Fatkun batch download to mass download approximately 2,200 sample images with varying dimensions, angles, and lighting of 15 distinct classes. Manual photography of items at Late Meal.
- *Annotation:* the bounding boxes of each item were drawn with Vott and exported into the YOLO dataset format.
- *Aggregation:* each class was labeled separately, so a script was used to combine images and their corresponding annotations into a singular data folder. Additional scripts were used to eliminate duplicate or unnecessary data.

Google Colaboratory

- To speed up the process of training for our object detection, we used Google Colab for its GPU.

Tiny YOLO-v3

- We used a smaller, quicker version of YOLOv3, the You Only Look Once object detect algorithm, known for its speed and use of only a single convolutional neural network.

Figure 4: images and labels of the 15 classes our detector can identify.

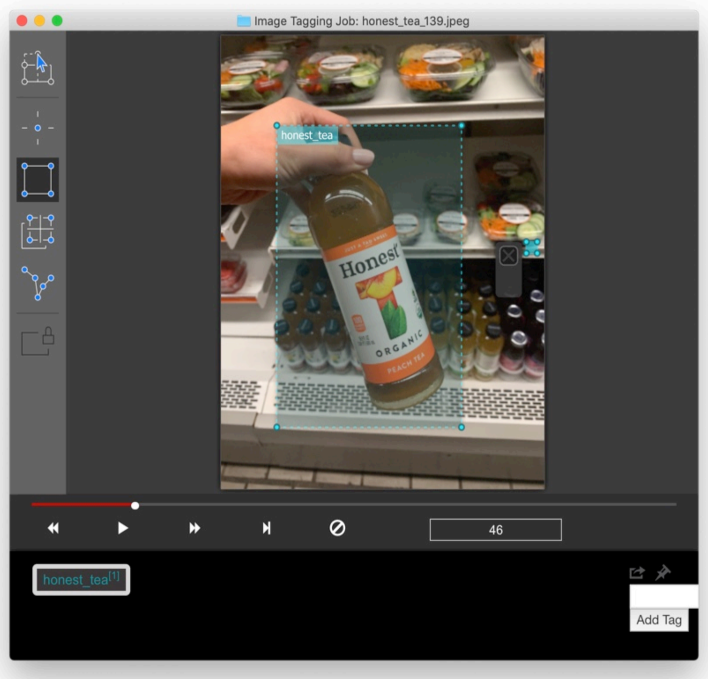

Figure 5: VOTT v1.7.2 software used for manually tagging images.

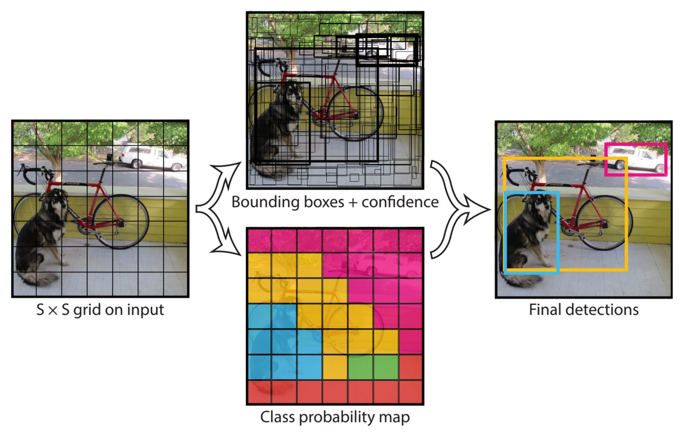

Figure 6: the detection model.

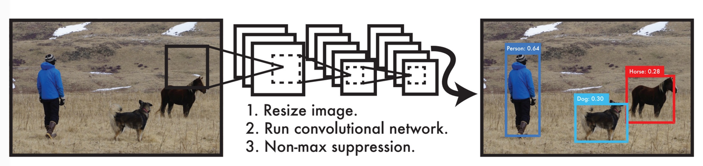

Figure 7: YOLO image processing technique.

RESULTS

Mean Average Precision (mAP) Analysis

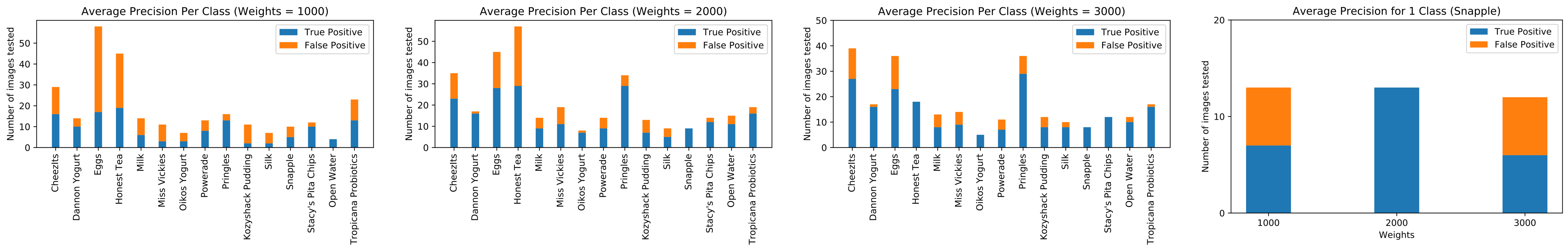

Figure 1: The first 3 graphs show the average precision (AP) for a 15-class detector with weights after 1000, 2000, and 3000 epochs, respectively. The last graph shows the AP for a single-class detector with weights after 1000, 2000, and 3000 training epochs, respectively.

- For the multi-class detector, the mAP for using the weights after 3000 epochs is highest at 73%. However, the mAP for the single-class detector was highest while using the weights after 2000 epochs at 85.71%

Multi-Class vs. Single-Class Detection

- We wanted to compare the accuracy of a multi-class detector with a single-class detector trained on the portion of the original dataset which corresponds to only one of the labels.
- We expected a single-class detection to have more false positives, while a model trained on many classes would have more false negatives.

Figure 2: the items predicted by the single-class detector with objectness scores of 96%, 77%, 61%, 17%, and 65% under the label "snapple."

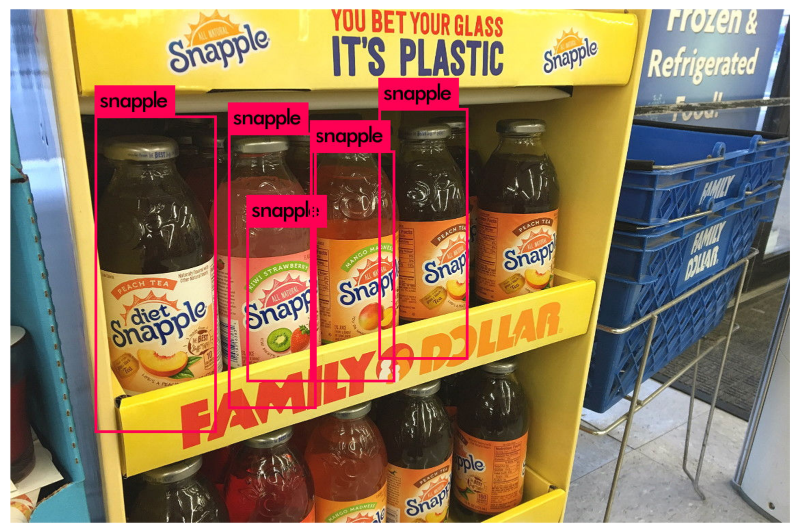

Figure 3: the items predicted by the multi-class detector with objectness scores of 30%, 31%, 16%, 30%, and 12% under the label "snapple."

CONCLUSIONS

Strengths

- YOLO v3 is very efficient while maintaining relatively high accuracy.
- Dataset is tailored toward our goal of creating an app specifically for Princeton University students.

Weaknesses

- Used a minimal number of images for training (~100-200 per class) due to time constraints.
- YOLO v3 sacrifices precision for speed compared to other methods (i.e. RetinaNet).

Future Work

- Autonomous price checking application for Princeton University students to use at Late Meal.
- Optimizing detection with additional images and more precise boundary box labeling for training.
- Experimentation with training on images of the complete product versus logos.

REFERENCES

- AlexeyAB. "AlexeyAB/Darknet." GitHub, January 10, 2020. https://github.com/AlexeyAB/darknet/.
- Kamal, Amro. "YOLO, YOLOv2 and YOLOv3: All You Want to Know." Medium. Medium, October 9, 2019. https://medium.com/@amrokamal_47691/yolo-yolov2-and-yolov3-all-you-want-to-know-7e3e92dc4899.
- Rafi. "Train Your Own Tiny YOLO v3 on Google Colaboratory with the Custom Dataset." Medium. Medium, July 25, 2019. https://medium.com/@today.rafi/train-your-own-tiny-yolo-v3-on-google-colaboratory-with-the-custom-dataset-2e35db02bf8f.
- Redmon, Joseph. YOLO: Real-Time Object Detection. Accessed January 13, 2020. https://pjreddie.com/darknet/yolo/.
- Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi. "You Only Look Once: Unified, Real-Time Object Detection." *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.* https://doi.org/10.1109/cvpr.2016.91.