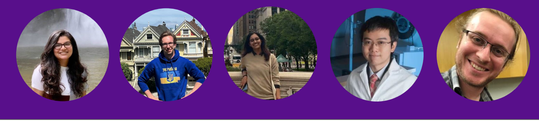
Training AI to recognize objects of interest to the blind community

Khevna Parikh, Lev Paciorkowski, Tharangini Sankarnarayanan
Mentors: Kevin Chan, Giles Hamilton-Fletcher, Chen Feng



Abstract

Goal: Using artificial intelligence ("AI") techniques in computer vision to locate the objects in the image and recognize specific objects important to the blind community.

Methodology:

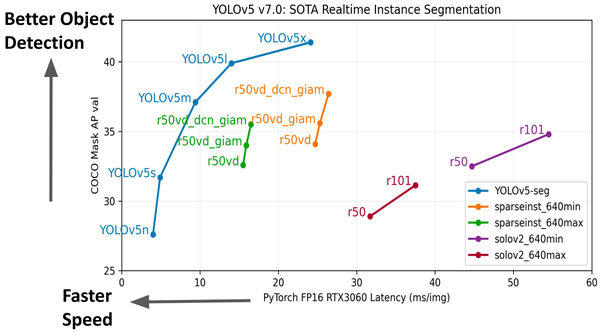
- ❏ Train a convolutional neural network (CNN) to detect specific objects of interest to the blind community.
- ❏ Implement techniques for rebalancing a training set exhibiting severe class imbalances.

Results/Discussion: Filtering repetitive images from the training set improves the model's overall performance after training.

What is YOLO?

'You Only Look Once (YOLO)' is a collection of state-of-the-art computer vision models.

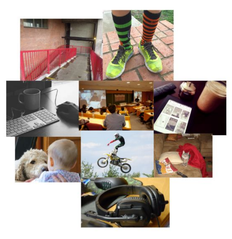
- ❏ Designed to mimic animal vision, YOLO models break an image apart into chunks and then learn to identify and compose bounding boxes around specific objects.
- ❏ There are multiple versions of YOLO models, varying from smaller architectures used for simple object detection tasks and larger architectures used for more complex detection tasks.
- ❏ In general, the more complex architectures require higher training maintenance but ultimately result in superior performance.

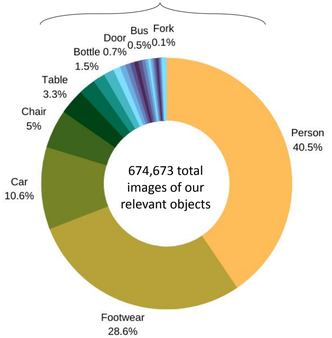


We utilize YOLOv5x in our analysis which outperforms historical models.

Data

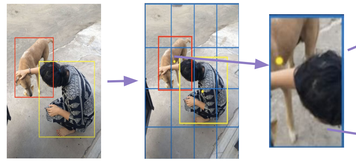
Google Open Images V6 consists of 1.9 million images, with information on bounding boxes and labels for over 600 object classes. Based on participant questionnaires, surveys and research, 35 objects were identified as important to the blind community.





Model Architecture & Training

YOLO divides the image into a grid system, where each grid cell is responsible for detecting objects within itself.



Since the center of the dog falls into this grid cell, this grid cell is responsible for detecting the dog. It does not consider the person even if a part of them is present. The model's output for a cell is vectorized as:

Confidence score of the box: P(Object) X IOU	Bounding Box Height	Bounding Box Width	Bounding box center X coordinate	Bounding box center Y coordinate	Probability of class 1	Probability of class 2	…	Probability of class k

For this cell, the vector would be:

Center of Dog is present in this cell, so calculate the coordinates of bounding box of Dog

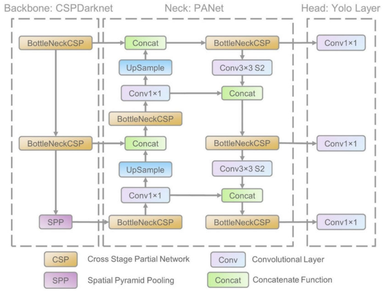
Probability of Dog = 1
Probability of Person = 0

1	2	1.3	0.05	0.3	1	0

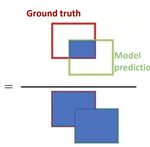
We have 16 grid cells, each one with a vector of size 7. In one forward pass, you can make all the predictions at once, there is no need to do 16 iterations, hence the name You only look once.

The full internal architecture consists of 3 main segments:

(1) Backbone
(2) Neck , and
(3) Yolo Layer.



Loss Functions: YOLO uses three loss functions during training



1. *Intersection over Union (IoU) box loss measures the bounding box error.*

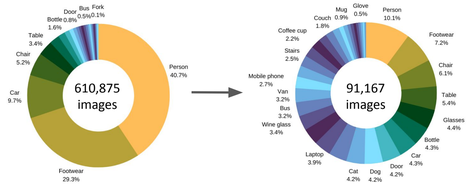
$$\ell(x, y) = L = \{l_1, \ldots, l_N\}^\top, \quad l_n = -w_n \left[y_n \cdot \log \sigma(x_n) + (1 - y_n) \cdot \log(1 - \sigma(x_n)) \right]$$

2. *Binary Cross Entropy object loss captures errors in the model's confidence score that an object exists.*

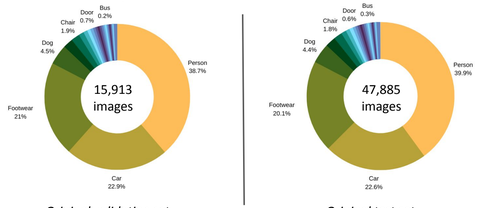
$$\ell(x, y) = L = \{l_1, \ldots, l_N\}^\top, \quad l_n = -w_{y_n} \log \frac{\exp(x_{n,y_n})}{\sum_{c=1}^{C} \exp(x_{n,c})} \cdot 1\{y_n \neq \text{ignore\_index}\}$$

3. *Cross Entropy class loss captures classification errors of the model's predictions.*

To improve the training of our model, we carefully pruned the training set to have a more balanced class distribution. However, to evaluate the model in the most realistic setting, we kept the original validation and test datasets unpruned.



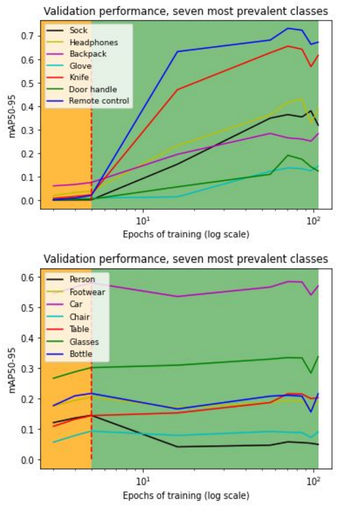
Original training set (90% random split) *Pruned training set*



Original validation set (2.5% random split) *Original test set (7.5% random split)*

By selectively eliminating the superfluous images containing the most common objects, pruning enabled each training epoch to be completed 9x faster. It helped the model to learn how to detect the less frequent, but still important objects.

Results



Orange shaded region: *usage of full training dataset.*

Green shaded region: *usage of pruned training dataset.*

Performance on held-out test dataset

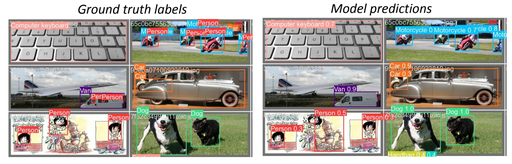
Class	Prevalence (by instances)	Precision	Recall	mAP50	mAP50-95
Cat	0.61%	0.863	0.884	0.913	0.793
Dog	1.34%	0.829	0.884	0.899	0.787
Mobile phone	0.26%	0.699	0.886	0.843	0.77
Coffee cup	0.23%	0.749	0.738	0.811	0.742
Bus	0.46%	0.581	0.751	0.752	0.661
…	…	…	…	…	…
Bottle	1.55%	0.217	0.389	0.232	0.189
Couch	0.16%	0.346	0.483	0.247	0.189
Glove	0.06%	0.125	0.351	0.166	0.125
Chair	4.98%	0.29	0.233	0.156	0.0969
Person	40.51%	0.288	0.043	0.119	0.0603
Overall Average	**100%**	**0.488**	**0.533**	**0.496**	**0.400**

Note: test performance results calculated on the best model during training, selected by unweighted average validation mAP50-95 across all classes.

Key highlights:

- ❏ Careful pruning of the training set improved the model's validation performance on the least prevalent classes.
- ❏ However, this comes at a trade-off of a more limited ability to learn the most prevalent classes.

Ground truth labels *Model predictions*



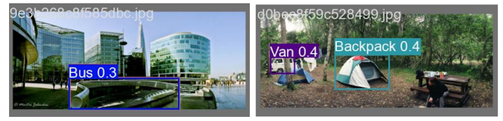
- ❏ Precision is the degree of exactness of the model in identifying only relevant objects.
- ❏ Recall measures the ability of the model to detect all ground truths representing the proposition of TPs among all ground truths.
- ❏ AP@α is Area Under the Precision-Recall Curve(AUC-PR) evaluated at α IoU threshold. Mean Average Precision (MAP) is the average of AP values over all classes.

$$AP@\alpha = \int_0^1 p(r)\, dr \qquad \text{mAP@}\alpha = \frac{1}{n} \sum_{i=1}^{n} \text{AP}_i \quad \text{for n classes.}$$

Conclusions & Future Steps

Conclusions

- ❏ When the training dataset exhibits severe class imbalances, careful pruning to balance out the class distribution is effective in helping the model learn the least standard classes.
- ❏ Some of the model's mistakes are understandable - e.g. confusing 'Car' with 'Van'. On paper, the model's performance metrics are low for 'Person'. However, many pictures of people involve large groups where it is difficult to label every individual precisely, as in the examples.
- ❏ The model is generally confident and correct in identifying 'Cat' and 'Dog'. This can be particularly helpful for blind persons in locating guide dogs.
- ❏ The model's ability to locate small, everyday objects, such as knives or remote controls, which can be found anywhere in a room, could be helpful to blind people.



Future Steps

- ❏ Train for semantic segmentation with DeepLab – a different model architecture. Compare classification ability with object detection approach. Will semantic segmentation do better in some classes than in predicting bounding boxes?
- ❏ Can we further prune the training set or use more specialized machine learning techniques to improve performance in some of the more common classes?
- ❏ Integrate the trained model into the AI-Sigh' app to be used by the blind community.

Acknowledgements

We want to thank our mentors, Kevin Chan, Giles Hamilton-Fletcher and Chen Feng from NYU Langone Health and NYU Tandon School of Engineering, for their support and direction throughout the project. Additionally, we thank the Capstone instructors, teaching assistants and staff for their teaching of the course and bootcamps. Finally, we thank the NYU High-Performance Computing admin team.

References

- ❏ YOLO V5 https://github.com/ultralytics/yolov5
- ❏ Google OpenImages V6 https://storage.googleapis.com/openimages/web/factsfigures_v6.html
- ❏ R. Mohammed, J. Rawashdeh and M. Abdullah, "Machine Learning with Oversampling and Undersampling Techniques: Overview Study and Experimental Results," ICICS 2020
- ❏ Diwan T, Anirudh G, Tembhurne JV. Object detection using YOLO: challenges, architectural successors, datasets and applications. Multimed Tools Appl. 2022 Aug
- ❏ Bronskill, John and Massiceti, Daniela and Patacchiola, Massimiliano and Hofmann, Katja and Nowozin, Sebastian and Turner, Richard E., Memory Efficient Meta-Learning with Large Images, NeurIPS 2021
- ❏ BCE Loss - https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html
- ❏ Cross Entropy Loss - https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html?highlight=cross+entropy#torch.nn.CrossEntropyLoss