Pedestrian Detection

Tsugunobu Miyake, Bach Le, Junnan Shimizu

Context

What is our project?

- A deep learning model to detect pedestrians in video footage.

Why is this useful?

- Pedestrian detection is essential in many domains such as: autonomous driving, traffic management, safety statistics, and more.
- For example, pedestrian detection is essential for autonomous driving to prevent accidents and increase safety.
- Another example, pedestrian detection can be used to detect suspicious people in public areas using security camera footage.

Possible Approaches

There are many approaches that can be taken to accomplish the pedestrian detection task, however, we decided to analyze high-resolution still images.

Approach #1 - Region-based Convolutional Neural Network (R-CNN)

- R-CNN uses a selective search algorithm to detect possible candidates for general objects, and then applies a CNN to those candidates to classify them.

However, R-CNNs are slow to train and slow to predict. Possible workarounds include using:

- Fast R-CNN, which only applies on CNN for all candidate regions
- Faster R-CNN, which trains the algorithm that searches for candidates as well as the training of the classification of the candidates.

Possible Approaches

Approach #2 - You Only Look Once (YOLO) Model

- YOLO is an object-detection model originally produced in 2015 and iteratively improving since.
- YOLO treats the object detection model as a regression problem after separating images into multiple subsections which enables end-to-end training.

- However, YOLO is seen to have low detection accuracy in crowded cases.

Another potential approach we did not explore:

- Localized Semantic Feature Mixers - higher accuracy even when the people are clustered in the picture.

Dataset

Ped-traffic dataset of TJU-DHD datasets

- Images taken from a driving car
- 13,858 training images and 2,136 validation images. Test images are not annotated.
- Approximately 1 to 20 annotated pedestrians in the image.
- All images have 1624 x 1200 pixels resolution.
- Images are taken over a year, with diverse weather conditions, season, and time of the day.

- There is also a Ped-traffic dataset that contains images taken on an university campus, which we can use to cross-validate.

Proposed Methods

Initial Approach - Regression:

- Used regression approach with VGG16 CNN model to predict the number of pedestrians.
- The mean squared error did not improve from 5.

Revised Approach - YOLO v8

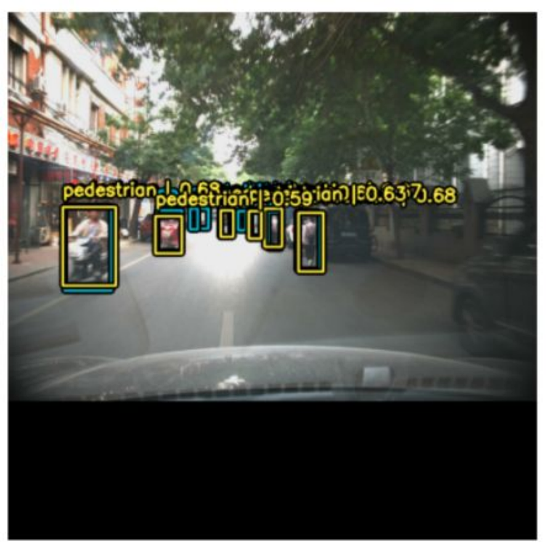
- Trained using YOLO v8 (2023) by Ultralytics, implemented in KerasCV.
- Used pre-trained backbone trained on Microsoft COCO dataset.
- Data Augmentation: Images were randomly flipped, scaled, cropped, and resized.
- For the prediction, non-max suppression was applied to combine duplicate pedestrian detections.

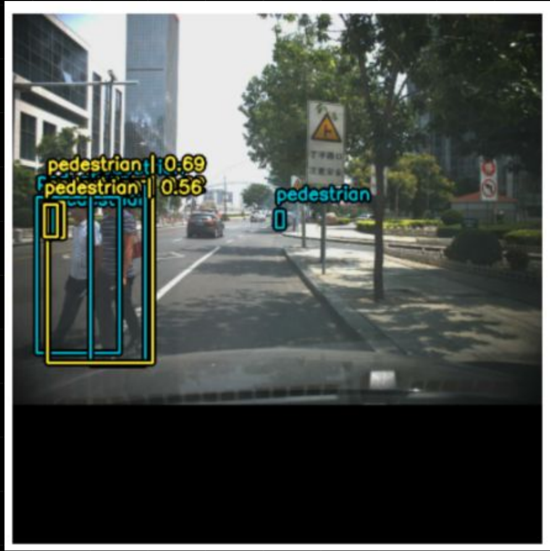
Evaluation Method

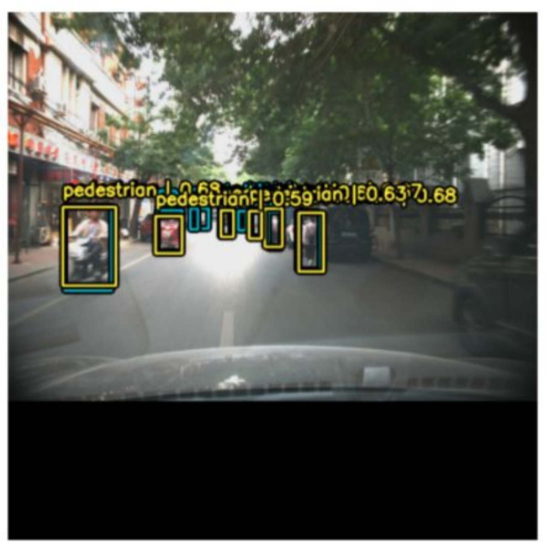
- To evaluate the strength of our YOLOv8 model, we used Complete Intersection over Union (CIoU)
  - CIoU is an extended version of Intersection over Union (IoU), a metric to measure and compute localization errors in object detection models.
  - This is done by calculating the overlap between two bounding boxes, the prediction bounding box and the ground truth bounding box.
- CIoU takes image proportion and the difference in center coordinates into account to evaluate results more accurately.

- We compared the accuracy of predicting the number of pedestrians to compare the VGG16-based model with the YOLO-model.
- For the YOLO-model, this is done by counting the number of predicted bounding boxes and actual bounding boxes.

Sample prediction by YOLO v8







Results and Discussions

Regression Model Performed a better result.

- YOLO v8 based: box loss = 2.94, MSE = 7.09, accuracy = 16%
- VGG-16 based: MSE = 5.37, accuracy = 25.1%

Why did this happen?

- Pretrained YOLO v8 backbone might be too small.
- The dataset contained a bounding boxes that are extremely small, or overlapped with another bounding box

Future research

- Train YOLO model with larger pretrained backbone
- Cross-validate our model with another dataset (ped-campus)
- Investigate another architecture that is more accurate with crowded people.

References:

1. Car ADAS. (2021, September 22). *Understanding ADAS: Pedestrian Detection.* https://caradas.com/understanding-adas-pedestrian-detection/
2. Paul, M., Haque, S. M., & Chakraborty, S. (2013). Human detection in surveillance videos and its applications - A Review. *EURASIP Journal on Advances in Signal Processing*, *2013*(1). https://doi.org/10.1186/1687-6180-2013-176
3. Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *2014 IEEE Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1109/cvpr.2014.81
4. Girshick, R. (2015). Fast R-CNN. *2015 IEEE International Conference on Computer Vision (ICCV)*. https://doi.org/10.1109/iccv.2015.169
5. Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(6), 1137–1149. https://doi.org/10.1109/tpami.2016.2577031
6. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/cvpr.2016.91
7. Khan, A. H., Nawaz, M. S., & Dengel, A. (2023). Localized semantic feature mixers for efficient pedestrian detection in autonomous driving. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/cvpr52729.2023.00530
8. Pang, Y., Cao, J., Li, Y., Xie, J., Sun, H., & Gong, J. (2021). Tju-DHD: A diverse high-resolution dataset for Object Detection. *IEEE Transactions on Image Processing*, *30*, 207–219. https://doi.org/10.1109/tip.2020.3034487
9. Ultralytics. (2024, April 17). *Home*. Ultralytics YOLOv8 Docs. https://docs.ultralytics.com/
10. lukewood, Stenbit, I., & Patel, T. (2023, August 10). *Keras Documentation: Object Detection with KerasCV*. Keras. https://keras.io/guides/keras_cv/object_detection_keras_cv/
11. Keras Team. (n.d.). *Keras Documentation: CIoU Loss*. Keras. https://keras.io/api/keras_cv/losses/ciou_loss/