Video-based Object Detection Using Voice Recognition and YoloV7

Issa Abdoul Razac Djinko
Department of Computer Science
and Information Technology
University of the District of Columbia
Washington, DC, USA
Email: issaabdoulrazac.djin@udc.edu

Thabet Kacem
Department of Computer Science
and Information Technology
University of the District of Columbia
Washington, DC, USA
Email: thabet.kacem@udc.edu

***Abstract*—Artificial Intelligence (AI) developments in recent years have allowed several new types of applications to emerge. In particular, detecting people and objects from sequences of pictures or videos has been an exciting field of research. Even though there have been notable achievements with the emergence of sophisticated AI models, there needs to be a specialized research effort that helps people finding misplaced items from a set of video sequences. In this paper, we leverage voice recognition and Yolo (You Only Look Once) real-time object detection system to develop an AI-based solution that addresses this challenge. This solution assumes that previous recordings of the objects of interest and storing them in the dataset have already occurred. To find a misplaced object, the user delivers a voice command that is in turn fed into the Yolo model to detect where and when the searched object was seen last. The outcome of this process is a picture that is provided as evidence. We used Yolov7 for object detection thanks to its better accuracy and wider database while leveraging Google voice recognizer to translate the voice command into text. The initial results we obtained show a promising potential for the success of our approach. Our findings can be extended to be applied to various other scenarios ranging from detecting health risks for elderly people to assisting authorities in locating potential persons of interest.**

***Index Terms*—Artificial Intelligence, Object Detection, Voice Recognition.**

I. Introduction

Two hundred years ago, no one could have imagined the technology would evolve to the extent it reached today. The basic technology we take for granted would have been considered witchcraft in the 1800s. Yet nowadays, technology has radically re-imagined the applications and services that we rely on in our daily lives. In particular, AI [1] is a field that aims to equip machines with the capability of dealing with information from the perception to the inference. Actually, this field is not quite new since the first reference to AI dates back to 1943 when McCullouch and Pitts [2] formally defined the first artificial neuron for the Turing Machine. Since then, this field has seen considerable development at a fast pace in recent years as it can be used for event prediction, speech recognition or even visual perception.

Conversely, video surveillance has benefited from the recent developments in cloud computing and communication but the challenge of detecting objects from images and video sequences has not been fully overcome yet. Several research efforts, such as in [3], proposed techniques based on machine learning, deep learning or even optical flow methods in this context with concrete performance gains.

In this paper, we tackle this problem from a different angle as we leverage video-based object detection and voice recognition to help people locate misplaced items. Our approach takes advantage of Yolo and Google voice-text recognizer. The latter is used to set up keywords by converting the user's voice into text before feeding it to Yolo that is used in turn to detect the object the user is searching for. When our Yolo model receives a keyword, it assigns a specific version of the trained model to find the objects and saves the resulting video, sorted by date, into a pre-defined folder. Therefore, the user can know where the searched item was seen last by watching the latest video.

The research objectives of this paper are (1) to explore how machine learning can be leveraged to identify objects from sequences of images and videos with high performance (2) to build an easy-to-use solution to find misplaced items in the household by combining machine learning and voice recognition (3) to leverage the outcome of this paper in tackling similar problems such as detecting health risks for elderly people.

The rest of the paper is organized as follows. Section II presents the related work. Section III explains the basic notions of Yolo. Section IV describes our approach in details. Section V presents the simulation setup while Section VI discusses the results of the conducted experiments. Section VII concludes our paper and highlights our future work.

II. Related Work

Karmarkar and Honmane [10] proposed a system to help visually impaired people using Yolo. The model was trained with the Coco dataset of 330K images of various daily used objects [10]. A bounding box is then generated around each detected object. The method generates five values to estimate the position and displacement of the object. When the camera focuses on the object, a triangle is developed around it and the closer the object is to the camera, the width and angles of the triangle increase. Then, the approach determines the distance to the object in question. The paper is different from our in the sense that our study assumes that the user is able to see and locate the recording video in order to find where the object was seen last. Also, we do not rely on the distance from the object to guide the user.

Zhang et al. [12] proposed using Yolo version 3 from camera feeds of an Unmanned Aerial Vehicle (UAV) to detect pedestrians. The authors rely on the box prediction feature of YOLO and the Feature Pyramid Network (FPN) to draw boxes upon the detected objects . The paper is very efficient when it comes to following something in motion, something or someone who is moving very fast. Our paper is different as it does not follow an object in motion, but we instead assume that the objects are located within the user's home and but they were just misplaced.

Jana and Biswas [11] proposed an approach that relies on recorded videos to identify any objects. By processing 40 frames per second (fps), the model divides images into NxN number of grids, effectively identifying the grids containing objects, and constructs a bounding box around them. The authors apply Yolo version 2 to detect and classify the objects, then assign an accuracy percentage. The findings of this paper are aligned with ours in the sense that the detection of objects is done through video recordings. However, we are running our own custom Yolo version detection model that achieves much better performance gains, as shown in Figure 1. Also, our goal is to find specific objects in our videos and the model was trained with our own dataset to guarantee maximum detection.

Priyankan and Fernando [13] proposed an approach based on Yolo to identify different species of fish by running an analysis on fish images. They created a mobile application by gathering these components using the following experimental setup: a PC equipped with core i7 CPU, 16GB of DDR3 memory, Ubuntu 14.04 64-bit, NVIDIA DIGITS 5, MATLAB R2012, and B-BOX-label. Since Yolo can use 40–90 fps, this approach used a neural network consisting of 24 convolutional layers. Then, the authors trained the model with 800 to 900 images and found 16 fish species. The model takes 3 to 20 seconds to detect the species. The test result revealed a 77% accuracy in bounding and classifying the fish species. Our paper is also customizing a model on a set of images of specific objects. However, our paper makes object detection based on videos while adding a voice-text-translation feature to fine-tune the objects the user is looking for.

III. Video-based Object Detection Fundamentals

Yolo is an object detection algorithm that divides any image into grids and determines the pixels in which the object is. When the location of the object is determined in the image, a bounding box is then drawn around it and labeled [7]. Yolo was first introduced in 2016 by Jason Redmon et al. [5]. Since then, there have been a great deal of Yolo versions that were proposed: actually 7 versions, and each one comes with its different levels of training. Figure 1 shows the differences in terms of performance of these versions when trained with the Coco dataset. The purple curve shows the latest one, i.e. Yolov7. It can be observed that Yolov7 outperforms the rest of the other five versions by a significant margin.

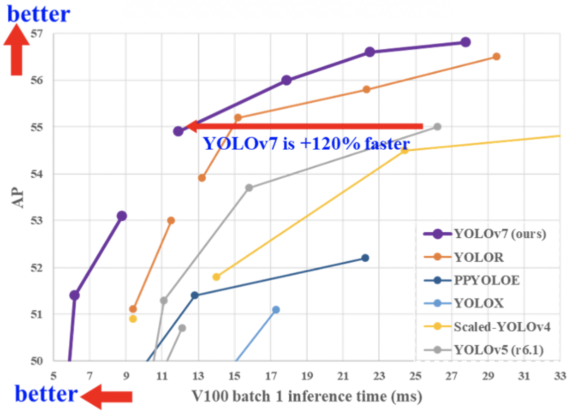


Fig. 1. Comparison of Yolo-v7 with previous versions [6].

IV. Proposed Approach

A. Approach Rationale

The motivation behind this paper is to provide an easy-to-use service for people to find misplaced items by combining several techniques including video-based object detection using a custom Yolov7 model, a Google voice recognizer and video surveillance data.

B. Approach Description

The approach can be summarized in 7 steps, as shown in Figure 2:

- The user sends a voice command to the application by providing the keyword of the searched object.
- The Google recognizer transcribes the analog voice input into text.
- It compares the scripted input with a predetermined dataset to see if any item was called forth.
- When this is completed, theGoogle voice-text recognizer function sends the desired keyword to the Yolov7 module that we customized with a set of specific keywords.
- When the Yolo module receives the keyword, it goes through the videos that were recorded every day in certain time intervals to detect when the object was seen last.
- The algorithm then saves a video with the bounding boxes including the time and date that the object was detected in a folder of our choosing.
- Finally, upon locating the latest video entry in that folder, the user gets an evidence of the last location where the searched object was seen last.

C. Video-Based Object Detection Algorithm

In line 1 of the algorithm, we define the different inputs with capital letters for simplicity. The user was given the letter A, the Google voice-to-text recognizer was given the letter Y, and the different custom models were given the letter X. Line 2 starts the event-based loop that detects when the user sends a voice command before the Google recognizer picks it up from

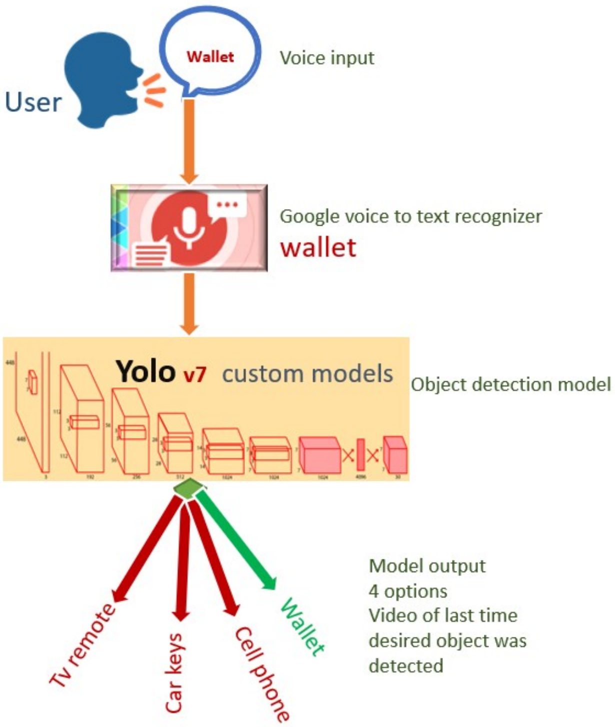


Fig. 2. High-level architectural view of the approach.

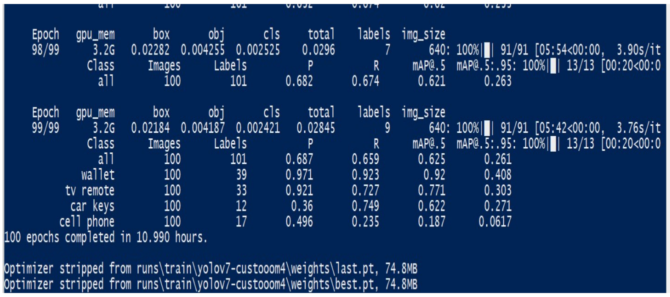
```
Algorithm 1: Video-based object detection algorithm.
1  Inputs: A=user, Y=Google voice-text recognizer,
     X=Yolov7 Custom Model;
2  while voice detected by Y do
3      Y asks for confirmation it is an input from A;
4      if keyword is confirmed then
5          Y sends keyword to X;
6      end
7      else
8          Star over;
9      end
10     if X receives keyword then
11         X runs last video;
12         X output object detection;
13         X saves output in folder;
14     end
15 end
```

there and transcribes the voice into clear text. In line 3, the Google recognizer asks for a confirmation from the user about the transcribed voice command. In lines 4-6, the algorithm checks if the keyword was validated before passing it to the object detection function leveraging the Yolo model. Lines 6-9 deal with the situation in which a keyword is not confirmed, which in turn restarts the process until another keyword is confirmed. In lines 10-14, after a keyword is sent to the Yolo model, the object detection function detects when the searched object was seen last, generates an image output, and saves it in a predetermined folder.

V. Simulation

The simulation process was the most interesting part of this project. First, we had to clone the Yolov7 from GitHub [8] then installed the required dependencies. In order to get better performance results, we did not want to use an already trained model for this project. That is why, we decided to train our custom model ourselves using the initial cloned model that was trained on the Coco dataset. Therefore, we took some images of the objects from the videos for the convolutions to successfully go through and find the objects. After collecting the images, we labelled them using ”Makesense.ai”, which is a free open-source tool to label images. The images were 461 in total: 80% for training and 20% for validation. The labeling process was quite long because we had to do this one image at a time. In the beginning of the training, the model collected 362 labelled images for training and 100 for validation. There were 95 convolutions in this base weight while the number of epochs was 100. Finally, we created a folder that would receive the output of the trained model.

Figure 3 shows the output of the anaconda power-shell Prompt at the very last epoch after the completion of the training process. The model found 100 images per recognized keyword with different precision levels per keyword, while some of the performance metrics we used included the recall rate and the mean average precision. The amount of time it took for the training to be completed was about 10 hours run on the local machine. It is important to note that we wanted to establish a proof of concept to show the feasibility of our approach and that is why we combined the four keywords and their corresponding images before extending this work in the future.



```
     Epoch   gpu_mem       box       obj       cls     total    labels  img_size
     98/99      3.2G   0.02282  0.004255  0.002525    0.0296         7       640: 100%|█| 91/91 [05:54<00:00,  3.90s/it
               Class    Images    Labels         P         R    mAP@.5  mAP@.5:.95: 100%|█| 13/13 [00:20<00:0
                 all       100       101     0.682     0.674     0.621     0.263

     Epoch   gpu_mem       box       obj       cls     total    labels  img_size
     99/99      3.2G   0.02184  0.004187  0.002421   0.02845         9       640: 100%|█| 91/91 [05:42<00:00,  3.76s/it
               Class    Images    Labels         P         R    mAP@.5  mAP@.5:.95: 100%|█| 13/13 [00:20<00:0
                 all       100       101     0.687     0.659     0.625     0.261
              wallet       100        39     0.971     0.923      0.92     0.408
           tv remote       100        33     0.921     0.727     0.771     0.303
            car keys       100        12      0.36     0.749     0.622     0.271
          cell phone       100        17     0.496     0.235     0.187    0.0617
100 epochs completed in 10.990 hours.

Optimizer stripped from runs\train\yolov7-custooom4\weights\last.pt, 74.8MB
Optimizer stripped from runs\train\yolov7-custooom4\weights\best.pt, 74.8MB
```

Fig. 3. Training details at the last epoch.

VI. Results and Discussions

A. Confidence Level

The different versions of Yolo come with advantages and drawbacks: the more advanced the detection model, the more data and time it requires to be trained. We considered version 7 to be appropriate for our paper as it offers much better performance, as explained in Section III. We also decided to use the base weight model of Yolov7 called ”Yolov7.pt”. As

Figure 4 shows, the confidence level curve for each searched keyword started very well. The various colors correspond to the different keywords we used in our experiments. It can be observed, for instance, that the light blue, which represents the wallet keyword, seemed to have benefited from much more images than the rest, thus making it overshadow the rest of the variables. The dark blue color represents the average of all variables. On average, the wallet confidence level curve performed much better at around 90 percent. For some reason, the cell phone, represented by the curve in red, exhibited a poor performance.

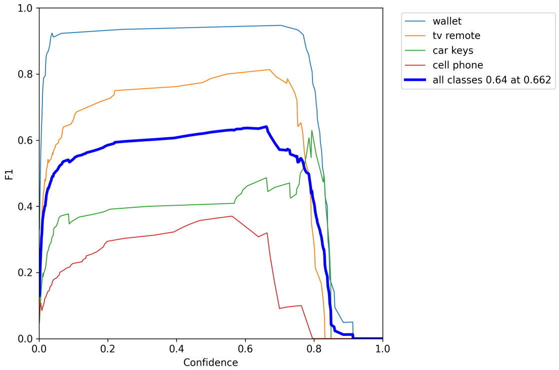


Fig. 4. Confidence level curve.

B. Precision and Confidence Level

As one of the goals of this paper was to improve the performance, we ran several rounds of training with different numbers of epochs in order to ensure we achieve the highest precision possible. As shown in Figure 5, the last training round showed high levels of prediction for the average variable of all keywords, which was on the rise, and overall most variables hit the 90% accuracy of precision level. This ultimately means that the model was able to see an object and make an accurate prediction of what it might be. Also, the model predicted that there are 90 car key images in the dataset. However, we had 100, then it can be concluded that the model was 90% precise with regard to that keyword.

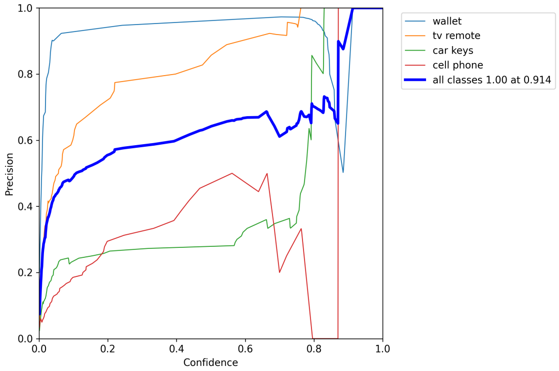


Fig. 5. Precision vs confidence.

C. Recall and precision

The recall is defined to represent the correlation between the true positives and the total number of predictions — the better the recall, the better the model. False positives and false negatives can be an issue in the output of the model, such as wrong labelling of some images. Therefore, the better the correlation between the precision and the recall, the better the model is. In Figure 6, we observed a big gap between the variables. This translates to the need to run more training with much more images from different angles and heights as the model is sensitive to the data it is fed with.

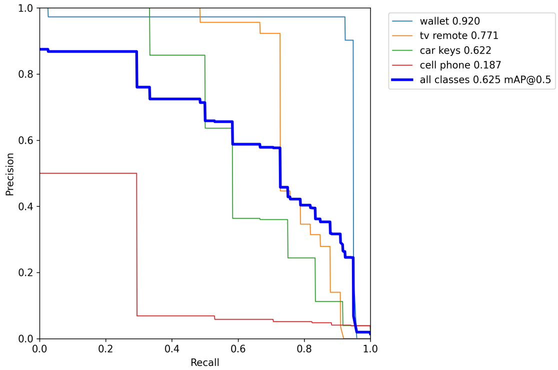


Fig. 6. Precision and recall curve.

D. Confusion matrix

Figure 7 shows the confusion matrix of our model, which is a method that is commonly used in the classification process. This matrix also shows the difference between what the model predicted versus what the real object being predicted actually is [9]. For instance, if the model predicts a tree, and in reality the object is a tree, then we call that a true positive. If the model predicts that the object is not a tree and, while the object is not a tree indeed, we call that true negative.

On the other hand, if the model predicts that there is a tree when it is not, that is called a false positive. When the model predicts no tree when there is one, that is called a false negative. In our case, the model did very well predicting the wallet variable. Indeed, it predicted the wallet with 95% accuracy. Regarding the tv remote, it reached up to 89% prediction accuracy. The cell phone keyword received a 71% accuracy in our model prediction.

E. Summary of the results

At the end of the 10 hours of training, we received a brief summary of the newly-trained model. Figure 8 shows the general summary of the model's performance. The new information is about the classification performance, the fact that the model recognized an object for what it is. The predictions in the training are shown in the first row and the validation is shown in the second row.

The resulting images of the model, shown in Figure 9, are satisfying in the sense that we accurately see a bounding box

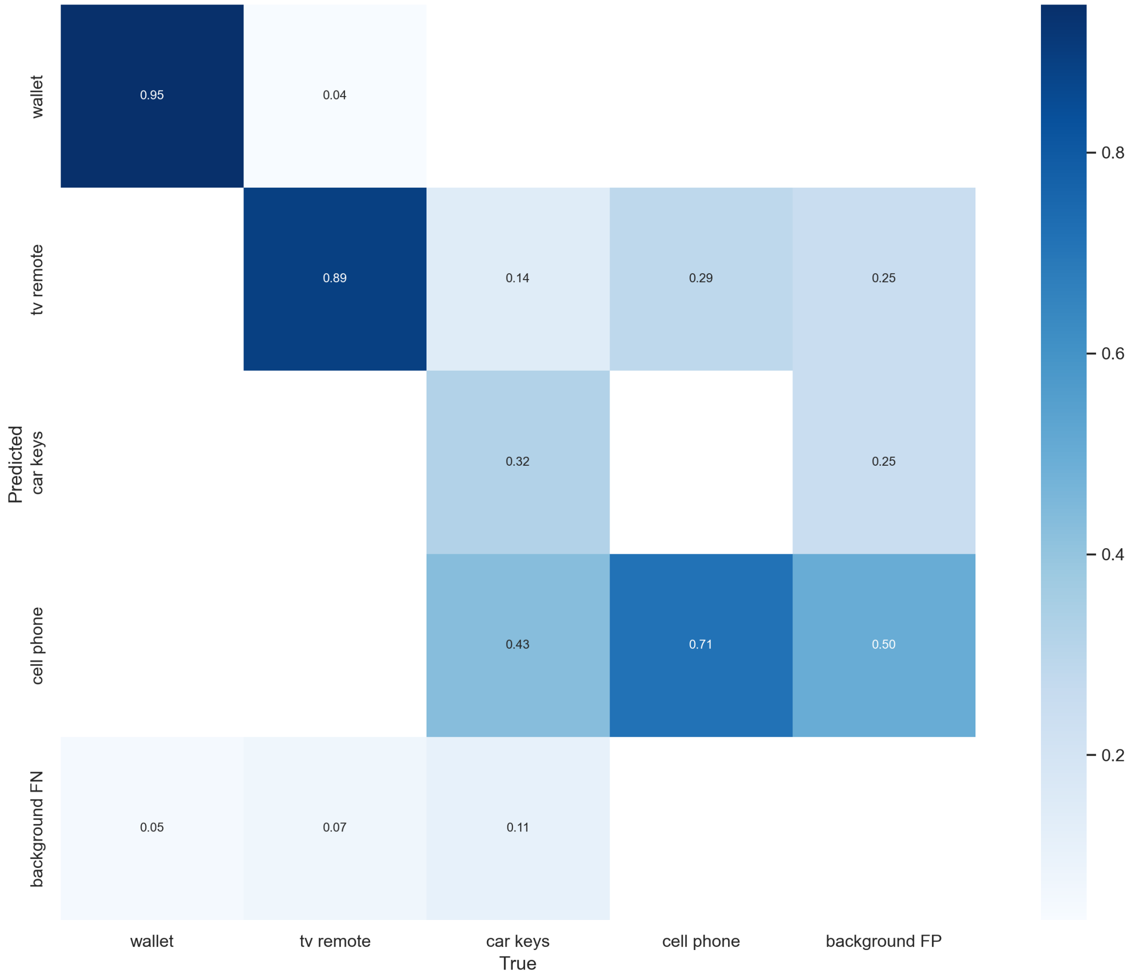


Fig. 7. Confusion matrix.

created around the objects of interest. The accuracy of the labels can be improved with more training and much more data to be fed into the model. The goal is to get an accurate distinction of the objection we might have lost. Therefore, when we place the four variables among many other objects, the model would be able to create a box with the objects we are looking for. Moreover, for a human eye that can recognize and categorize multiple objects instantly, we can say that the experiments that we conducted show the approach have promising potential of success at larger scale.

VII. Conclusion and Future work

In this paper, we leveraged Yolo object detection model along with Google voice recognition in order to help people find misplaced items in their household. Actually, we had more experience working with the previous versions of Yolo, but we had to figure out how to improve on them with regards to needed data for training, and much more computing power to sustain the high cost of processing. That is why we chose Yolov7 thanks to its better performance when compared to the previous versions. Also, we chose the base weight to train our custom model and fed the training model with 461 images in total before observing the outcome of the training phase. When the training was completed, the results showed a good potential of success. The objects that the training seemed to recognize best were the wallet and cell phone. That can be explained by the fact that both had a lot of images in training. The outcome of the experiments we conducted was a model that accurately creates a bounding box around the interested objects among many other uninterested objects and saves that in video evidence in a pre-determined folder. This can be considered a success because humans can easily recognize objects as long as they know where they might be.

In the future, we plan to train separate models in the cloud according to their specific object in order to reduce the training time. Also, to improve the performance, the voice input would choose one object linked with its model. In addition, we plan to feed the model with a much higher number of images per model per object from all types of angles and distances to guarantee maximum accuracy. Finally, we also plan to explore how we can extend this work to detect health risks for elderly

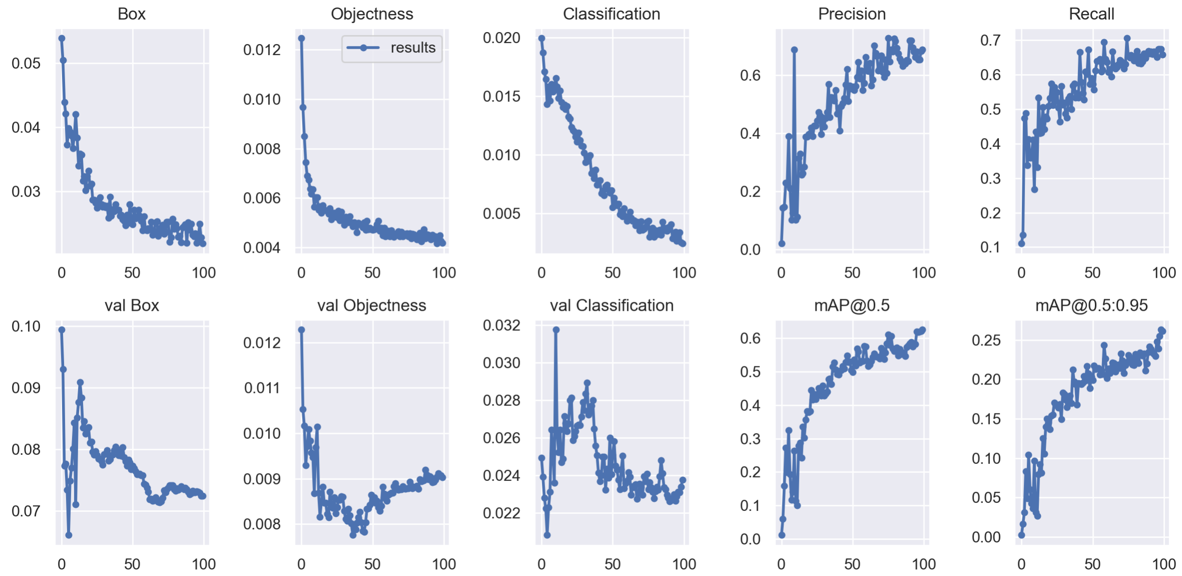


Fig. 8. Results.

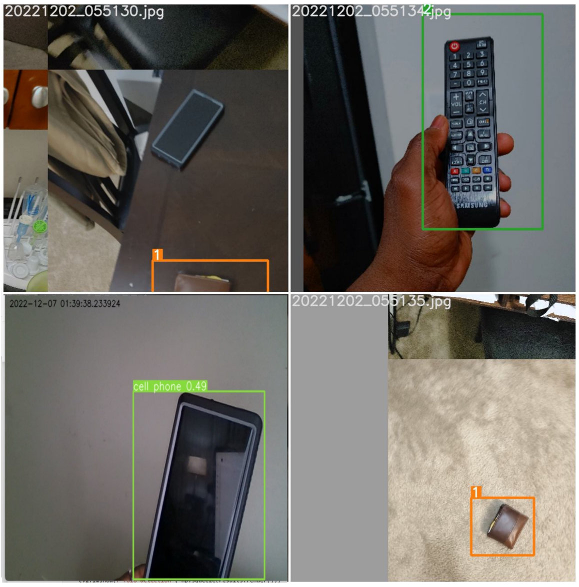


Fig. 9. Training output.

people such as falling.

Acknowledgment

This work was funded by the National Science Foundation (NSF) award 2011689.

References

[1] J. M. Helm et al., "Machine learning and artificial intelligence: definitions, applications, and future directions". Current reviews in musculoskeletal medicine, 13(1), pp. 69-76.

[2] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity". The bulletin of mathematical biophysics, 1943 pp.115-133.

[3] Y. T. Liu, "The Ultimate Guide to Video Object Detection", Toward Data Science available at https://towardsdatascience.com/ug-vod-the-ultimate-guide-to-video-object-detection-816a76073aef, [accessed January 2023].

[4] G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions" Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection". The 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 779-788.

[6] C. Y. Wang, A. Bochkovskiy, and H. Y. M. Liao, "YOLOv7: Trainable bag of freebies sets new state of the art for real-time object detectors", arXiv preprint arXiv:2207.02696, July 2022.

[7] H. Wen, F. Dai, and Y. Yuan, "A Study of YOLO Algorithm for Target Detection". The 2021 International Conference on Artificial Life and Robotics (ICAROB2021), January 2021, pp.70-73.

[8] Y. K. Wong, Implementation of YoloV7 on GitHub. available at https://github.com/WongKinYiu/yolov7, [accessed March 2023].

[9] S. Visa, B. Ramsay, A. L. Ralescu, and E. Van Der Knaap, "Confusion matrix-based feature selection". The Twenty Second Midwest Artificial Intelligence and Cognitive Science Conference, 2011, pp. 120-127.

[10] M. R. R. Karmarkar, and V. N. Honmane, "Object Detection System For The Blind With Voice Guidance". The International Journal of Engineering Applied Sciences and Technology, 2021, pp. 2455-2143.

[11] A. P. Jana, and A. Biswas, "Yolo based Detection and Classification of Objects in video records". The 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), May 2018, pp. 2448-2452.

[12] Sr. D. Zhang et al., Y. "Using YOLO-based pedestrian detection for monitoring UAV". The Tenth International Conference on Graphics and Image Processing, 2018 Vol. 11069, pp. 1141-1145.

[13] K. Priyankan and & T. G. I. Fernando, "Mobile Application to Identify Fish Species Using YOLO and Convolutional Neural Networks". The International Conference on Sustainable Expert Systems, 2021, pp. 303-317.