Volume: 12 Issue: VII Month of publication: July 2024

DOI: https://doi.org/10.22214/ijraset.2024.63641

Real Time Dress Code Adherence Recognition in an Academic Setting Using a Deep Learning Model

Ayobami Olawale Fakunle[1], Ayobami Olatunde Fawole[2], Busari Olukayode Ayodeji[3], Olorunfemi Micheal Ayeni[4]
[1]*Department of Mechatronics Engineering, The Polytechnic, Ibadan, Oyo State, Nigeria*
[2,4]*Department of Electrical Engineering, The Polytechnic, Ibadan, Oyo State, Nigeria*
[3]*Department of Computer Engineering Technology, The Polytechnic, Ibadan, Oyo State, Nigeria*

Abstract: Machine learning is finding application in many fields as a tool. Its increasing adoption fueled by rapid advancements in algorithms and hardware. Deep learning techniques have shown great capabilities in image recognition, face recognition and other vision tasks. The proposed model describes the use of a deep learning method for the soft biometrics' classification of clothing according to a predefined dress code standard in an academic setting. The Yolov4 architecture is used in this work for detection and classification. A custom dataset of images is gathered at a higher institution of learning by volunteer students which are subsequently box annotated for location of clothed figures. These are used for training and testing of the dress code detection model. The proposed model indicates detection by drawing bounding boxes and classifies by gender into appropriately dressed APD and not appropriately dressed NAPD. The results indicate that the proposed deep learning model is an efficient and successful network configuration for dress code detection and classification.

Keywords: Soft biometrics, computer vision, deep learning, image classification, cloth recognition

I. INTRODUCTION

The resurgence of Artificial intelligence in the past decade due to multiple converging inflection points in both massively parallel computing hardware and improved algorithms has brought about great strides in NLP natural language processing, computer vision, and automation among other areas. AI as an applied tool in non-core computer scientists has gained massive popularity and rightfully so because of the widely favorable results from recent machine learning techniques compared to traditional programming.

Dress code compliance is a requirement on many occasions, industry dress code for safety purposes, dress code to ensure lab safety, public spaces not marked for recreational purposes, higher instructions for distraction mitigation or where specific dress code is required

The application of AI in Computer vision for image recognition opens up ability to extend the capability of tapping into a massive amount of visual data which are already available or easy to acquire.

Deep learning especially provides a tool that reduces the need for a human in the role of feature extraction which is slow, prone and to error, reduces ability to self-learn.

In the problem of apparel segmentation, deep learning has shown promising results. The absence of annotated data, the usage of datasets with unbalanced classes, and high category similarity, however, are frequently mentioned as being difficult issues by authors.

This work provides a foundation for an automated method of identifying individuals not following the dress code as stipulated in the guidelines of an institution in real time through a deep learning-based computer vision method. This is a Multitask objective involving localization + classification.

The model proposed primarily categorizes appropriateness of clothing as detailed in the handbook of a higher institution of learning in still images and MP4 video formats.

II. LITERATURE REVIEW

Extracting human attributes using a convolutional neural network approach [1] Authors proposed employing a convolutional neural network to categorize soft biometric features. To perform gender differentiation, distinguish between type of upper clothing (long or short sleeves), and lower clothing (long or short), they employed independent classifiers. Despite the fact that the method produced a model with strong generalization ability, the authors observed difficulty in locating suitable image datasets in terms of size, quality, and variability. It's slow inferencing is also not suitable for real time recognition purposes

Deep Learning Based Protective Equipment Detection on Offshore Drilling Platform [2] Their two staged method using YOLOv3 + ResNet50 achieves a higher accuracy in the protective equipment detection on offshore drilling platforms compared with other deep learning models. The detection accuracy of these proposed method for helmets and work-wear uniforms are 94.8% and 95.4%, respectively. The paper did not show a metric for inferencing time or number of detections in relation to suggest good enough performance for real-time use. However, from the work of [3] its deducted that the method has limitations considering the time of inference and precision.

A Robust Real-time Component for Personal Protective Equipment Detection in an Industrial Setting [3] A brand-new monitoring system component for safety analysis was proposed. This component uses the data stream from a regular CCTV camera to determine the appropriate use of personal protective elements in real-time. A system component using YOLO v3 is used. The methodology is accurate enough to be useful and can function in real-time. Mean Average Precision (80.19% mAP) and 80 FPS. It was limited to a single PPE (Hard Hats) which makes it unsuitable for more arbitrary definitions required for clothing recognition

YOLOv4: Optimal Speed and Accuracy of Object Detection [4] A cutting-edge detector was presented that can be trained on low resource workstations such as those found in single GPU workstations and free/low cost cloud systems and still outperformed every other alternative detector in terms of speed (FPS) and accuracy greater than 90% mAP (MS COCO AP50:::95 and AP50). It is called YOLOv4

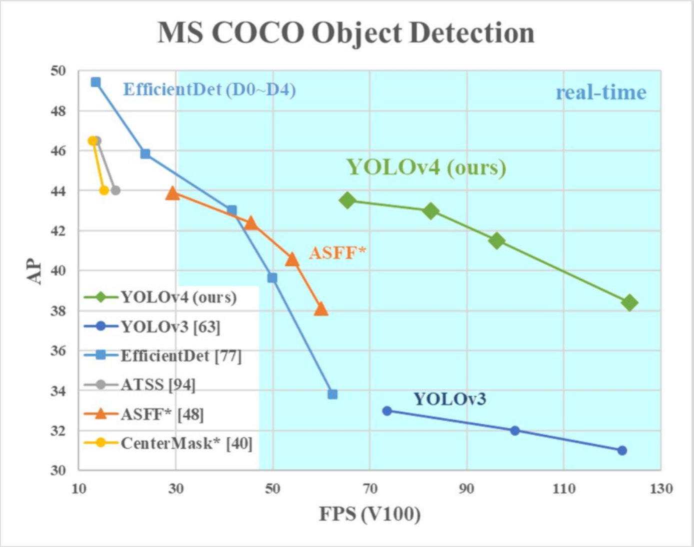


Fig. 1 Comparison of the proposed YOLOv4 and other state-of-the-art object detectors. YOLOv4 runs twice faster than EfficientDet with comparable performance. Improves YOLOv3's AP and FPS by 10% and 12%, respectively

III. METHODOLOGY

The YOLO v4 network is used in this study to recognize dress code conformity. As a one-shot technique for classification of detection into predetermined properties, YOLO is used.

The performance of the YOLOv4 based model is evaluated using the mAP (Mean Average Precision) and FPS (frames per second) metrics, which are used in several object detection models (e.g., Faster R-CNN, R-CNN, SSD).

Optimizations are made for increased performance by finding the minimum input image size that produces zero to minimal drop in mean average precision mAP while increasing the frames per second FPS.

A. Data Generation

The bespoke dataset created from photos gathered on the polytechnic campus serves as the source of the dataset for detection.

There are a total of four (4) classes for detection using the Yolov4 based model. The classes taken from this dataset are APD (*Appropriately Dressed*) Female, NAPD (*Not Appropriately Dressed*) Female, APD Male and NAPD Male. Some of the data is also taken from videos of random places with people walking around. Images of varied background and location were taken i.e., images mostly taken in studios, photos taken outdoors with colorful backgrounds and random photos taken from mobile phones on the campus. A total of 1800 images are used for detection out of which 80% i.e., 1440 images are used for training the YOLO v4 based model and remaining 364 images are used for validation.

Following the thorough collection of these images, there was a visual review to weed out any pictures that were misplaced or of poor quality. The photographs that were taken in other settings are screened to ensure they mimic the conditions that can be observed in an academic setting.

Each batch of the images is annotated by an individual using the *labelimg* data labelling tool for annotation of the image datasets

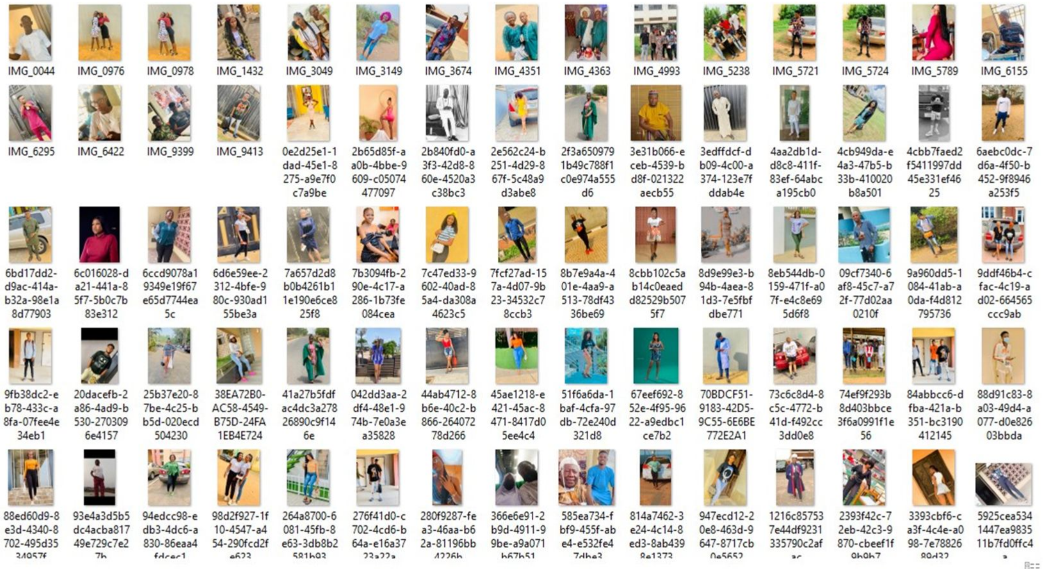

IMG_0044
IMG_0976
IMG_0978
IMG_1432
IMG_3049
IMG_3149
IMG_3674
IMG_4351
IMG_4363
IMG_4993
IMG_5238
IMG_5721
IMG_5724
IMG_5789
IMG_6155
IMG_6295
IMG_6422
IMG_9399
IMG_9413
0e2d25e1-1dad-45e1-8275-a9e7f0c7a9be
2b65d85f-aa0b-4bbe-9609-c05074477097
2b840fd0-a3f3-42d8-860e-4520a3c38bc3
2e562c24-b251-4d29-867f-5c48a9d3abe8
2f3a6509791b49c788f1c0e974a555d6
3e31b066-eceb-4539-bd8f-021322aecb55
3edffdcf-db09-4c00-a374-123e7fddab4e
4aa2db1d-d8c8-411f-83ef-64abca195cb0
4cb949da-e4a3-47b5-b33b-410020b8a501
4cbb7faed2f5411997dd45e331ef4625
6aebc0dc-7d6a-4f50-b452-9f8946a253f5
6bd17dd2-d9ac-414a-b32a-98e1a8d77903
6c016028-da21-441a-85f7-5b0c7b83e312
6ccd9078a19349e19f67e65d7744ea5c
6d6e59ee-2312-4bfe-980c-930ad155be3a
7a657d2d8b0b4261b11e190e6ce825f8
7b3094fb-290e-4c17-a286-1b73fe084cea
7c47ed33-9602-40ad-85a4-da308a4623c5
7fcf27ad-157a-4d07-9b23-34532c78ccb3
8b7e9a4a-401e-4aa9-a513-78df4336be69
8cbb102c5ab14c0eaedd82529b5075f7
8d9e99e3-b94b-4aea-81d3-7e5fbfdbe771
8eb544db-0159-471f-a07f-e4c8e695d6f8
09cf7340-6af8-45c7-a72f-77d02aa0210f
9a960dd5-1084-41ab-a0da-f4d812795736
9ddf46b4-cfac-4c19-ad02-664565ccc9ab
9fb38dc2-eb78-433c-a8fa-07fee4e34eb1
20dacefb-2a86-4ad9-b530-2703096e4157
25b37e20-87be-4c25-bb5d-020ecd504230
38EA72B0-AC58-4549-B75D-24FA1EB4E724
41a27b5fdfac4dc3a27826890c9f146e
042dd3aa-2df4-48e1-974b-7e0a3ea35828
44ab4712-8b6e-40c2-b866-26407278d266
45ae1218-e421-45ac-8471-8417d05ee4c4
51f6a6da-1baf-4cfa-97db-72e240d321d8
67eef692-852e-4f95-9622-a9edbc1ce7b2
70BDCF51-9183-42D5-9C55-6E6BE772E2A1
73c6c8d4-8c5c-4772-b41d-f492cc3dd0e8
74ef9f293b8d403bbce3f6a0991f1e56
84abbcc6-dfba-421a-b351-bc3190412145
88d91c83-8a03-49d4-a077-d0e82603bbda
88ed60d9-8e3d-4340-8702-495d35
93e4a3d5b5dc4acba81749e729c7e2
94edcc98-edb3-4dc6-a830-86eaa4
98d2f927-1f10-4547-a454-290fcd2f
264a8700-6081-45fb-8e63-3db8b2
276f41d0-c702-4cd6-b64a-e16a37
280f9287-fea3-46aa-b62a-81196bb
366e6e91-2b9d-4911-99be-a9a071
585ea734-fbf9-455f-abe4-e532fe4
814a7462-3e24-4c14-8ed3-8ab439
947ecd12-20e8-463d-9647-8717cb
1216c857537e44df9231335790c2af
2393f42c-72eb-42c3-9870-cbeef1f
3393cbf6-ca3f-4c4e-a098-7e78826
5925cea5341447ea983511b7fd0ffc4

Fig. 2 Example pictures from the generated dataset

B. *Data Processing*

To improve the generalization of the classifier by including extra copies of the training examples that have undergone non-classifying modifications, the training set's size is enhanced. Because the class is invariant to so many transformations and the input can be easily modified with numerous geometric operations, object identification is a classification job that is particularly suited to this type of dataset augmentation.

For detection purpose, objects in images are labelled with appropriate classes. In this case there are a total of four (4) classes: APD (*Appropriately Dressed*) Female, NAPD (*Not Appropriately Dressed*) Female, APD Male and NAPD Male. For annotations of images an opensource, python-based software called *LabelImg* is used in this work. It presents a simple graphics user interface GUI and multiple annotation formats including the YOLO format used in this work. According to the classes, people in photographs are labeled, and the corresponding labels per photograph are saved in a .txt file format. Labels may be readily made with LabelImg's user-friendly software and stored in YOLO format. The class label and bounding box coordinates of the image are described in the label file. The text file's values are saved in YOLO format and are listed in alphabetical order by class., x, y, w and h, Where

$$X = \frac{AbsoluteX}{WidthofTotalImage}$$

$$Y = \frac{AbsoluteY}{HeightofTotalImage}$$

$$W = \frac{AbsoluteWidth}{WidthofTotalImage}$$

$$H = \frac{AbsoluteHeight}{HeightofTotalImage}$$

Equation -Yolo Annotation Format

The absolute values of x,y,w and h are given below

$$\text{Absolute x} = (X_{min}+(\text{Absolute width}/2))$$
$$\text{Absolute y} = (Y_{min}+(\text{Absolute height}/2))$$
$$\text{Absolute width} = \text{abs}(X_{max}-X_{min})$$
$$\text{Absolute height} = \text{abs}(Y_{max}-Y_{min})$$

The images and the accompanying *.txt* files containing the annotations are placed in folder named *obj*. The *obj* folder is zipped and uploaded to google drive for cloud training. A python script is used to divide the uploaded data into two (2) sets: training and validation set in the ratio 80% to 20%.

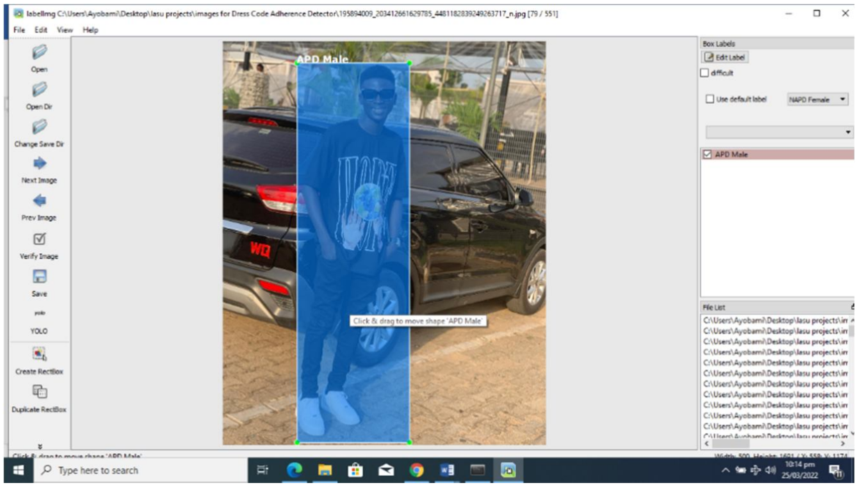


Fig. 3 Annotation of images NAPD male into classes using LabelImg

The following labelling practices were observed to create a high-quality dataset for the training validation and testing of the model:

1) The Label was placed around the entirety of an object. a little bit of non-object buffer is included rather than excluded with a rectangular label. This is to achieve the aim of having boxes that tightly captures the object of interest without cutting off any part of it.

2) Occluded objects are labeled in entirety as if the entire object is visible, label them entirely. This enables the trained model to understand the true bounds of objects

3) objects partially out of frame objects are labeled

C. Model Training

Training a neural network require a lot of computational power. Therefore, GPU's which are inherently good at matrix multiplication (which make up the bulk of training process) due to their parallel architecture. Given the present state of the GPU market (non-availability and inflated prices due to cryptocurrency mining)

The model training is done in the cloud leveraging the google collab platform, a cloud-based resource provided by google for researchers. The collab resource used is the free runtime which provides a virtual system with the following Specifications

1) 2 Intel Xeon CPU clusters

2) NVidia Quadro K80 GPU or Tesla T4 (randomly offered)

3) 12GB System RAM

4) 12GB GPU RAM

A graphics processing unit GPU such as the NVidia GeForce GTX 1080 Ti GPU with 11 GB of RAM or above would have been sufficient for training on a local machine.

For the YOLOv4 pretrained model, the model is started with weights trained on the COCO dataset. After preparing the dataset, it is necessary to modify the configuration file in order to train the network using the necessary model as the COCO dataset initially has eighty (80) classes. We have four (4) classes in our situation.

The yolo4 configuration file has to be configured for the custom model the changes are done according to the guidelines by the originators of the YOLO models. Therefore, the formula filters = (classes + 5) * 3 is used to compute the number of filters needed for four (4) classes, giving us a total of 27 filters. Therefore, the class value and filter numbers are set to 4 and 21, respectively, at the three (3) yolo detection layers. Input photos for training are divided into 64 batches with 16 subdivisions. In this situation, a resolution of 416x416 is utilized since the input resolution to the YOLO network is designed to be multiples of 32.

The max_batches line is changed to (classes*2000 this should not less than the number of training images and not less than 6000), for four (4) classes max_batches=8000.

Learning rate is set to 0.001 with a decay of 0. 0005 and a momentum of 0.9. As recommended by the authors of YOLOv4 the network is trained until the point after which the average loss will not decrease any further or remains constant. During training the weights of the model are saved temporarily every 100 iterations and a permanent weight is saved every 1000 iterations.

After training the network on the training set, the weights are used validate the model on Val set. To do this mean average precision (mAP) is calculated. The mAP is calculated at the first 1000 iteration weight of the model and subsequently after every 100 iterations. The last saved weights usually provide the better results. The mAP is given by comparing the original ground truths with the predicted outputs. This is a public link to a google drive containing a copy of the collab notebook https://colab.research.google.com/drive/1naJHOVwT1lmEx8B2HMMfxn9r2f-YpIo9?usp=sharing

IV. EXPERIMENTAL RESULTS

After training the network, the model is put to test with images not from the initial sets for training and validation. The dress code detection is done by the custom YOLOv4 based detector. To analyze the performance of the trained model, the weight with the best result is chosen. Each individual image is passed through the dress code detector model one after the other. Precision and frames per second (FPS) are the metrics used to measure the performance of the system. 20 images were chosen to be used for testing based on them containing features attributed with each class of the dataset. A mean average precision of 89% was achieved. The mAP gives the average precision of each class present in the image. The validation set of 364 images got a mAP of 82%

Test images were taken using a mobile phone. The output of some of the test images are shown below

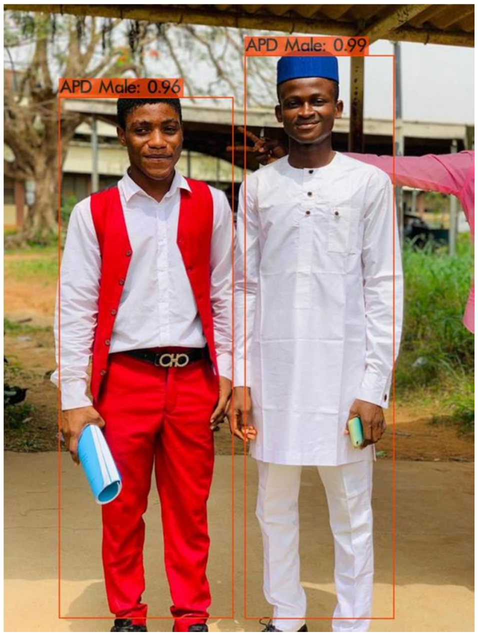


Fig. 4a Appropriately dressed males (APD)

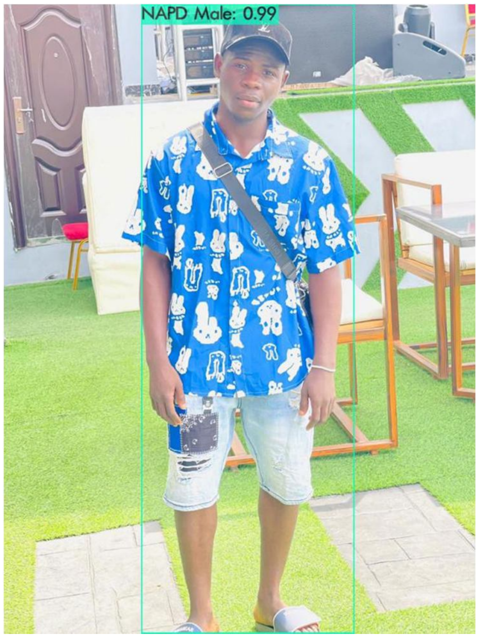


Fig. 4b Not Appropriately dressed males (NAPD)

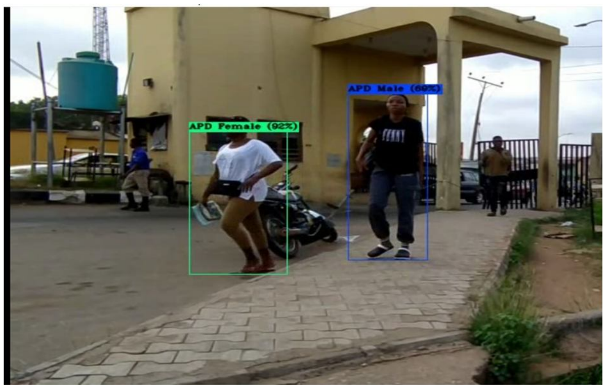


Fig. 8 Output from a video recording at the school entrance

From the test images (20 in number) the model could get most of it correctly with expected bounding boxes. For the test images from a mobile phone taking in different lighting conditions and background the model could get 89%. In some of the pictures some bounding boxes did not totally define the class. The percentage of accuracy of the model predictions on the test images is shown in **Error! Reference source not found.**.

TABLE 1
PREDICTED OUTPUT CONFIDENCE FOR TEST IMAGES.

Test Images	(APD) Female	(APD) Male	(NAPD) Female	(NAPD) Male
1	10	86		
2		98		
3		89		
4		90		
5		97		
6	66			
7	79	14		
8	76	33		
9	89			
10	92			
11			88	
12			89	
13			90	
14			95	
15			88	
16	20			89
17				88
18	10			95
19	12			98
20				92

It should be noted that there is a lesser distinction between the ADP male and ADP female classes than others, this is as observed by humans too. There is natural similarity in appropriately dressed members of both sexes

TABLE 2
CONFUSION MATRIX FOR THE TEST IMAGE

	ADP Male	ADP Female	NADP Male	NADP Female	Missing
ADP Male	5	1	-	-	-
ADP Female	1	3	-	-	-
NADP Male	-	-	4	-	-
NADP Female	-	-	-	5	-

- Precision: measures how accurate is your predictions. i.e. the percentage of your predictions are correct.

$$Precision = \frac{17}{17+2} = 0.894$$ Equation1 Calculating Model precision

- Recall: it is a measurement of how good a model finds all the positives.

$$Recall = \frac{17}{17} = 1$$ Equation -2 Calculating Model Recall

- F1 score is used to assess the quality of the model

$$F1 = \frac{2*(0.894*1)}{(0.894+1)} = 0.95$$ Equation -3 F1 calculation

Table 3
Mean Average Precision Map@0.50iou For The Model Weight At Different Training Iterations

Iterations	Mean Average Precision mAP
100	0.41
1000	0.70
2000	0.87
3000	0.85
4000	0.89
5000	0.90
6000	0.92
7000	0.92
8000	0.92

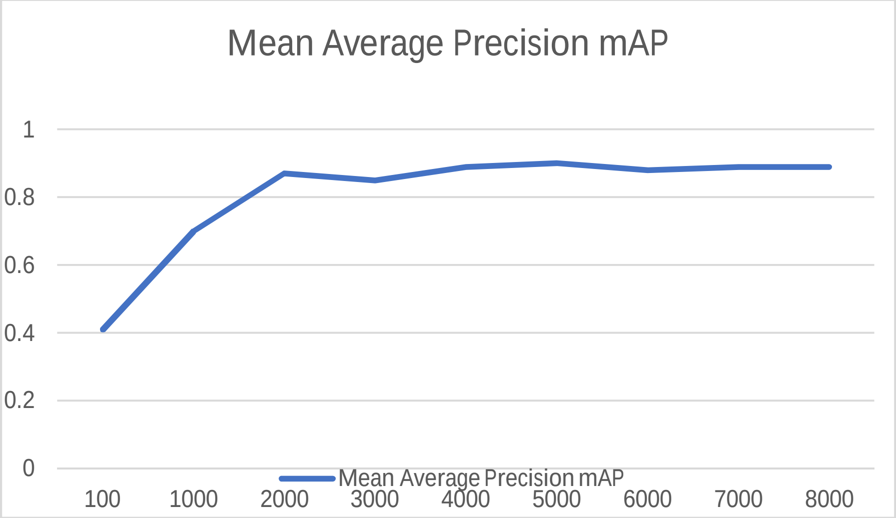
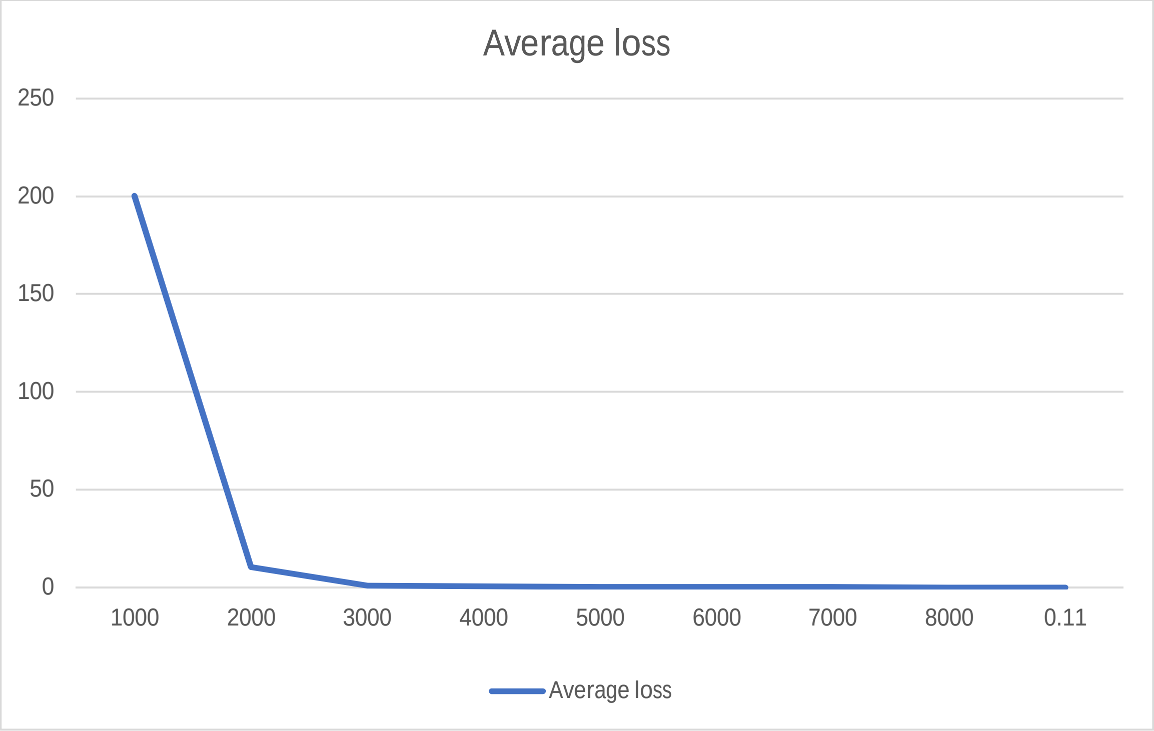


Fig. 9 mAP changes @ various iterations

Table 4
Average Loss At Various Training Iterations A Model It Is Generally Accepted That A Model Is Trained Once The Average Loss Trends Below 0.3

Iterations	Average loss
100	200.36
1000	10.54
2000	1.23
3000	0.91
4000	0.84
5000	0.52
6000	0.31
7000	0.12
8000	0.11

Fig. 10 Graph showing average loss vs number of iterations.



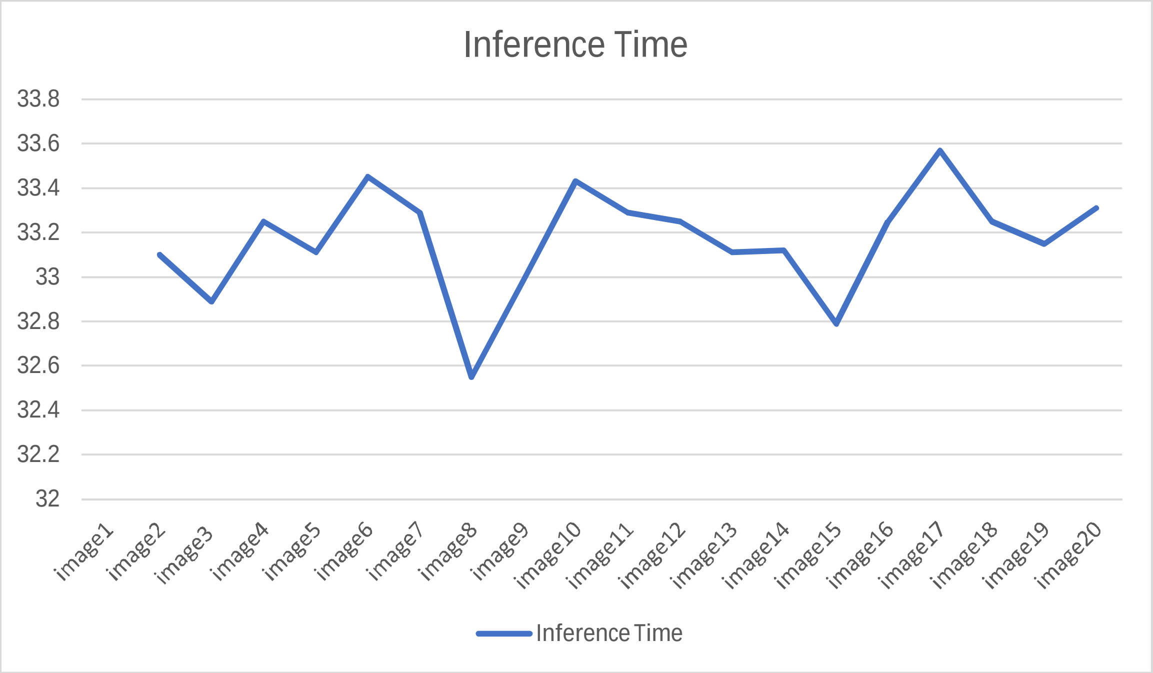


Fig.11 Inference time of test images

From Fig.11 plotted from the inferencing time for the test images, the average inferencing time per image is 33ms, when the training instance uses a NVidia Tesla T4 GPU processing about 30 frames per second this is fast enough for real-time use.

V. CONCLUSION

The database of images and yolov4 format annotations used for the training are provided through a link to a publicly available google drive link this is free to use in a opensource format as all images present are fully consented by the individuals there within.

As the trained models as a mAP of 89 the precision is sufficient for use as a component in an automated system for flagging inappropriately dressed students in an academic setting. The adoption of a yolov4 based model is justified by the training iterations required to see an appreciable increase in precision, starting at the 0.87 mAP at about the 2000th iteration. The neural network also converged with an average loss of 0.3 at the 6000th iteration.

By limiting the number of classes in the model to four (4) the training time is drastically reduced to about 62 hours. This is more of a necessity as the cloud based google collab notebook places a limit on the resources allocation making training a larger more complex network much less practical. The limit also had the effect of optimizing the inferencing time to about 33ms or about 30 FPS making the model well suited for real-time use. This was put to test using a recorded video and also a live webcam footage.

Following the definition adopted in this work *>25 FPS* (better than 5 FPS as adopted by Redmon et al. (2016) [5] it is clear that this approach is capable of processing videos in real-time applications and can be utilized as the base component of a monitoring system for dress code compliance.

This work can be further improved by a more granular approach involving the annotation of images on an item basis e.g., hair, trousers, top while removing the distinction in sexes. Thus, more classes can be employed, requiring more computing power for training which can be achieve with access to a local workstation with RTX 3090 and above GPU.

Further optimization could be possible by using the lite version of the YOLO model. These may allow the use of resource constrained edge devices such as the google mobile TPU, Nvidia Tegra platforms, raspberry pi for cost effective implementations.

REFERENCES

[1] H. A. Perlin and H. S. Lopes, "Extracting human attributes using a convolutional neural network approach," Pattern Recognition Letters, vol. 68, no. special Issue on Soft Biometrics, pp. 250 - 259, 2015.

[2] G. Faming, J. Xiaofeng, G. Wenjuan, Y. Xiangbing and G. Chenyu, "Deep Learning Based Protective Equipment Detection on Offshore Drilling Platform," Symmetry 13, p. 954, 2021.

[3] P. Torres, A. Davys, T. Silva and L. Schirmer, "A Robust Real-time Component for Personal Protective Equipment Detection in an Industrial Setting," In Proceedings of the 23rd International Conference on Enterprise Information Systems (ICEIS 2021) - Volume 1,, pp. pages 693-700, 2021.

[4] A. Bochkovskiy, C.-Y. Wang and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," Alexey Bochkorskiy, 2020.

[5] N. Carion, F. Massa, G. Synnaeve and N. Ununier, "End-to-End Object Detection with Transformer," Facebook, 2020.

[6] G. Ross, "Fast r-cnn," ICCV, 2015.

[7] R. Shaoqing, H. Kaiming, G. Ross and S. Jian, "Faster r-cnn: Towards real-time object detection with region proposal networks," NIPS, 2015.

[8] R. Joseph and F. Ali, "Yolo9000: better, faster,stronger.," CVPR, 2017.

[9] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks.," in ICML, 2019.

[10] T. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan and S. Belongie, "Feature pyramid networks for object detection," in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, 2017.

[11] A. L. Samuel, "some studies in machine learning using the game of checkers," IBM Journal of Research and development, vol. 3, no. 3, pp. 210-229, 1959.

[12] A. R. Muddam, "Clothing Recognition Using Deep Learning Techniques," Asian Institute of Technology, Thailand, 2019.

[13] E. Shelhamer, J. Long and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation arXiv:1605.06211 [cs.CV]," Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3431-3440, 2015.

[14] K. E. Koech, " Object Detection Metrics With Worked Example," Towards Data Science , 26 08 2020. [Online]. Available: https://towardsdatascience.com/on-object-detection-metrics-with-worked-example-216f173ed31e. [Accessed 02 03 2022].

[15] Deci, "The Object Detection Landscape: Accuracy vs Run Time," Deci, 24 May 2021. [Online]. Available: https://deci.ai/resources/blog/object-detection-landscape-accuracy-vs-runtime/. [Accessed 02 March 2022].

[16] M. Tan, P. Ruoming and V. L. Quoc, "Efficientdet: Scalable and efficient object detection," in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020.

[17] Y. T. Liu, "The Ultimate Guide to Video Object Detection," Towards Data Science, 13 May 2020. [Online]. Available: https://towardsdatascience.com/ug-vod-the-ultimate-guide-to-video-object-detection-816a76073aef. [Accessed 2 March 2022].

[18] L. Wei, A. Dragomir, E. Dumitru, S. Christian, R. Scott, F. Cheng-Yang and C. B. Alexander, "SSD: Single shot multibox detector," ECCV, 2016.

[19] N. Romero, M. Gutoski, L. Hattori and H. Lopes, "Soft biometrics classification using denoising convolutional autoencoders and support," in Anais do 13 Congresso Brasileiro de Inteligência Computacional., Curitiba,, 2017.

[20] A. d. S. Inácio, A. Brilhador and H. S. Lopes, "Semantic Segmentation of Clothes in the Context of Soft Biometrics Using Deep learning Methods," in Anais do 20 Congresso Brasileiro de Inteligência Computacional, Curitiba, 2020.

[21] J. G. Shanahan and D. Liang, "Introduction to Computer Vision and Real Time Deep learning Based object detection," in KDD 20,, Virtual Event, USA, 2020.

[22] S. Khan, H. A. S. Rahmani and A. Shah, A guide to Convolutional Neural Networks for Computer Vision, Morgan and Claypool., 2018.

[23] I. Goodfellow, B. Yoshua and A. Courville, Deep learning, MIT Press online: https://www.deeplearningbook.org/, 2016.

[24] S. (Ren, K. He, R. Girshick and J. and Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," Advances in neural information processing systems,, pp. 91-99, 2015.

[25] J. D. S. G. R. a. F. A. Redmon, "You only look once: Unified, real-time object detection," In Proceedings of the IEEE conf. has on computer vison and pattern recognition , pp. 779-788, 2016.

[26] M. A. Roy and M. A. Roy, "Clothing Recognition Using Deep Learning Techniques," Asian Institute of Technology, Thailand, 2019.

[27] V. I. A. B. A. S. S. Seferbekov, "feature pyramid network for multi-class land segmentation,”," in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, pp. 272-275, 2018.

[28] Q. Tian, S. Chanda, A. Kumar and D. Gray, "Improving Apparel Detection with Category Grouping and Multi-grained Branches," Amazom, 2021.

[29] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu and A. C. Berg, "SSD: Single shot multibox detector," in in Proceedings of the 14th European Conference on Computer Vision, Amsterdam, 2016.

[30] J. Wu, N. Cai, W. Chen, H. Wang and G. and Wang, "Automatic detection of hardhats worn by construction personnel: A deep learning approach and benchmark dataset," Automation in Construction, p. 106:102894, 2019.

[31] S. Huynh, "How to install LabelImg in Windows with Anaconda?," Medium, 13 May 2020. [Online]. Available: https://medium.com/@sanghuynh_73086/how-to-install-labelimg-in-windows-with-anaconda-c659b27f0f. [Accessed 23 March 2022]. BIBLIOGRAPHY