YOLOv4-Mosaic^2 No Pain No Gain.

1st Jehad Said Mahrous
Info. Tech & Comp. science
Nile university
J.Said2117@nu.edu.eg

2nd Lamiaa Omar Abdelaziz Hassan
Info. Tech & Comp. science
Nile university
l.omar2119@nu.edu.eg

3rd Mahmoud Mostafa Tayee
Info. Tech & Comp. science
Nile university
mamostafa@nu.edu.eg

***Abstract*—Object detection is one of the most famous and important problems in the filed of computer vision and deep learning and this challenge along with other challenges in field of computer vision and natural language processing using deep learning have been on the spot light of the whole research community for almost the last decade. There are many functions that can improve the accuracy of the target detection model. Usually, it is necessary to test the combination of these functions on a large number of data sets and prove the results theoretically. This paper starts from the data enhancement algorithm and carries on the comparison and the theory analysis to it. A new data enhancement algorithm is obtained. After using the new data enhancement algorithm to replace Mosaic algorithm of Yolov4 [1],stands for "you only look once", mean average precision on the training data has increased to be 97% in comparison with 89% which was derived from the YOLOv4 with its original architecture.**

Source code is at YOLOv4$_{Mosaic^2NoPainNoGain}$.

***Index Terms*—YOLOv4, Object detection, Mosaic, Mosaic^2**

I. Introduction

Over the past decade, Deep learning has drawn much greater attention and become hot technology in the Artificial intelligence area. Object detection aims to determine the class and location of a specific object in an image. It's widely used in many tasks around us every where nowadays. As it allows us to identify and locate objects in an image or video and With this kind of identification and localization, object detection can be used to count objects in a scene and determine and track their precise locations, all while accurately labeling them.

Today in this paper it's used to detect if the object, a human being, is wearing a mask or not. As corona virus sickness has become a big public health issue in 2019. The use of a face mask is among the most efficient methods for preventing the transmission of the Covid-19 virus. Wearing the face mask alone can cut the chance of catching the virus by over 70%. Consequently, World Health Organization (WHO) advised wearing masks in crowded places as precautionary measures. So, we needed a reliable mask monitoring system. To monitor population and check if they are wearing masks or not is a very exhaustive operation and to facilitate it, this paper propose a face mask detection software based on AI and image processing techniques.

II. Related work

YOLOv4 is the fourth version in the You Only Look Once family of models. YOLOv4 makes real-time detection a priority and conducts training on a single GPU. YOLOv4 was a real-time object detection model published in April 2020 that achieved state-of-the-art performance on the COCO dataset [2]. It works by breaking the object detection task into two pieces, regression to identify object positioning via bounding boxes and classification to determine the object's class. This implementation of YoloV4 uses the Darknet [3] framework.

By using YOLOv4, we are implementing many of the past research contributions in the YOLO family along with a series of new contributions unique to YOLOv4 including new features. All of the YOLO models are object detection models. Object detection models are trained to look at an image and search for a subset of object classes. When found, these object classes are enclosed in a bounding box and their class is identified. Object detection models are typically trained and evaluated on the COCO dataset which contains a broad range of 80 object classes. From there, it is assumed that object detection models will generalize to new object detection tasks if they are exposed to new training data.

The main goal of YOLOv4 is designing a fast operating speed of an object detector in production systems and optimization for parallel computations, rather than the low computation volume theoretical indicator (BFLOP).

A. YOLOv4 architecture

All object detectors take an image in for input and compress features down through a convolutional neural network backbone.

A modern detector is usually composed of two parts, a backbone which is pre-trained on ImageNet [4] and a head which is used to predict classes and bounding boxes of objects. But Fig. 1 shows the architecture of YOLOv4 and it consists of 3 main parts not just 2. The extra part is the Neck which is used to combine the feature extracted from the backbone in different levels to be used for the detection operation.

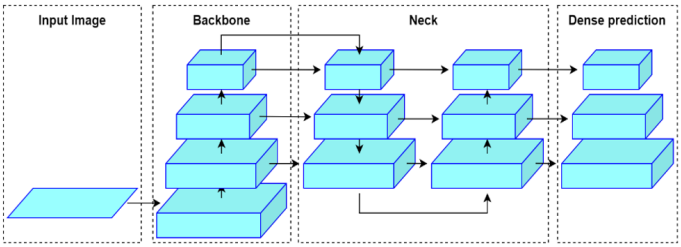

Fig. 1. YOLOv4 architecture.

Backbone is the deep learning architecture that basically acts as a feature extractor. All of the backbone models are basically classification models.
The backbone network for an object detector is typically pre-trained on ImageNet classification. Pre-training means that the network's weights have already been adapted to identify relevant features in an image, though they will be tweaked in the new task of object detection.

The authors considered the following backbones for the YOLOv4 object detector.

- CSPResNext50 [5].
- CSPDarknet53 [6].
- EfficientNet-B3 [7].

The CSPResNext50 and the CSPDarknet53 are both based on DenseNet [8]. DenseNet was designed to connect layers in convolutional neural networks with the following motivations: to alleviate the vanishing gradient problem (it is hard to backprop loss signals through a very deep network), to bolster feature propagation, encourage the network to reuse features, and reduce the number of network parameters.
In CSPResNext50 and CSPDarknet53, the DenseNet has been edited to separate the feature map of the base layer by copying it and sending one copy through the dense block and sending another straight on to the next stage. The idea with the CSPResNext50 and CSPDarknet53 is to remove computational bottlenecks in the DenseNet and improve learning by passing on an unedited version of the feature map.
EfficientNet was designed by Google Brain to primarily study the scaling problem of convolutional neural networks. There are a lot of decisions you can make when scaling up your ConvNet including input size, width scaling, depth scaling, and scaling all of the above. The EfficientNet paper posits that there is an optimal point for all of these and through search, they find it.
EfficientNet outperforms the other networks of comparable size on image classification. The YOLOv4 authors posit, however, that the other networks may work better in the object detection setting and decide to experiment with all of them.
Based on their intuition and experimental results, the final YOLOv4 network implements CSPDarknet53 for the backbone network. Fig. 2 shows a study on the different backbone neural networks and the difference in their parameters and emphasize on the FPS as the most significant criteria to choose according to the terminology of YOLOv4 authors.

Backbone model	Input network resolution	Receptive field size	Parameters	Average size of layer output (WxHxC)	BFLOPs (512x512 network resolution)	FPS (GPU RTX 2070)
CSPResNext50	512x512	425x425	20.6 M	**1058 K**	31 (15.5 FMA)	62
CSPDarknet53	512x512	725x725	**27.6 M**	950 K	52 (26.0 FMA)	**66**
EfficientNet-B3 (ours)	512x512	**1311x1311**	12.0 M	668 K	11 (5.5 FMA)	26

Fig. 2. Parameters of backbone neural networks.

In object detection, multiple bounding boxes need to be drawn around images along with classification, so the feature layers of the convolutional backbone need to be mixed and held up in light of one another. The combination of backbone feature layers happens in the neck.

And now we will present some of the neck manipulations used by YOLOv4 architecture and not changed in our architecture also.

1) Path Aggregation Network(PANet) [9]: PANet is present in the neck of the YOLOv4 model and it is mainly incorporated in the model to enhance the process of instance segmentation by preserving spatial information.
The reason why PANet is chosen for instance segmentation in YOLOv4 is because of its ability to preserve spatial information accurately which helps in proper localization of pixels for mask formation and Fig. 3 shows the PANet architecture.

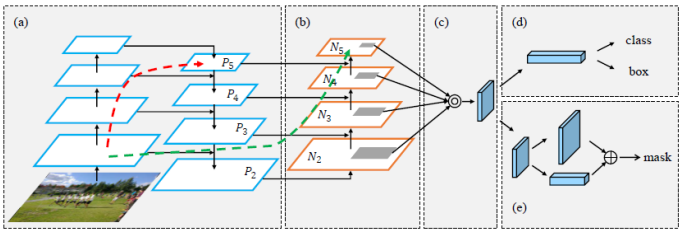

Fig. 3. PANet Architecture.

As the image passes through the various layers of the neural network, the complexity of the features increases and the spatial resolution of the image decreases concurrently. Due to this, the pixel-level masks cannot be identified accurately by the high level features.

It takes an additional bottom-up path to the top-down path taken by FPN. This helps in shortening that path by using clean lateral connections from the lower layers to the top ones. This is called the "shortcut" connection which is only about 10 layers.
PANet uses features from all the layers, and lets the network decide which ones are useful. It performs ROI Align operation on each feature map to extract the features for the object. This is followed by an element-wise max fusion operation to enable the network to adapt new features.
PANet conventionally adds the neighbouring layers together for doing mask predictions using the adaptive feature pooling. However, this approach is slightly twisted when PANet is employed in YOLOv4, such that instead of adding the neigh-

bouring layers, a concatenation operation is applied on them which improves the accuracy of predictions.

2) Spatial Pyramid Pooling(SPP) [10]: CNNs present a very important technical issue that limits some of their capabilities. This is a fact that the input image of a CNN should have a fixed size. In order to give an image of arbitrary size as input to a CNN, the image should be resized either by cropping or warping.

The solution to this problem lies in the Spatial Pyramid Pooling (SPP) layer. It is placed between the last Conv layer and the first FC layer and removes the fixed-size constraint of the network.

Spatial Pyramid Pooling (SPP) is a pooling layer that removes the fixed-size constraint of the network, i.e. a CNN does not require a fixed-size input image. Specifically, we add an SPP layer on top of the last convolutional layer. The SPP layer pools the features and generates fixed-length outputs, which are then fed into the fully-connected layers. In other words, they perform some information aggregation at a deeper stage of the network hierarchy (between convolutional layers and fully-connected layers) to avoid the need for cropping or warping at the beginning.

The input of the SPP layer is a set of d feature maps of arbitrary size. Then, the SPP layer maintains spatial information of these feature maps in local spatial bins. The output size of the SPP layer is fixed because the number and the size of these bins are always fixed as shown in Fig. 4.

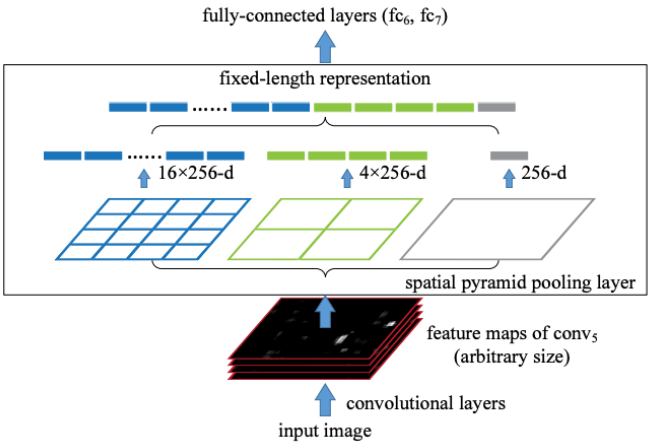

Fig. 4. PANet Architecture.

3) Spatial Attention Module(SAM) [11]: A Spatial Attention Module is a module for spatial attention in convolutional neural networks. It generates a spatial attention map by utilizing the inter-spatial relationship of features. Different from the channel attention, the spatial attention focuses on where is an informative part, which is complementary to the channel attention.

To compute the spatial attention, average-pooling and max-pooling operations are applied along the channel axis and concatenate them to generate an efficient feature descriptor is applied. On the concatenated feature descriptor, a convolution layer to generate a spatial attention map which encodes where to emphasize or suppress.

Channel information of a feature map is aggregated by using two pooling operations, generating two 2D maps. Each denotes average-pooled features and max-pooled features across the channel. Those are then concatenated and convolved by a standard convolution layer, producing the 2D spatial attention map as shown in “Fig. 9”.

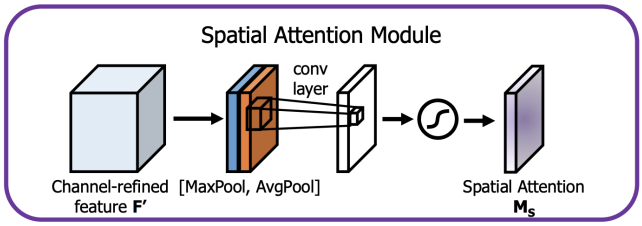

Fig. 5. PANet Architecture.

YOLOv4 deploys the same head as YOLOv3 [12] for detection with the anchor based detection steps, and three levels of detection granularity.

B. Bag of Specials

YOLOv4 deploys strategies called a ”Bag of Specials”, so termed because they add marginal increases to inference time but significantly increase performance, so they are considered worth it.

The authors experiment with various activation functions. Activation functions transform features as they flow through the network. With traditional activation functions like ReLU, it can be difficult to get the network to push feature creations towards their optimal point. So the research has been done to produce functions that marginally improve this process. Mish is an activation function designed to push signals to the left and right as shown in Fig. 6.

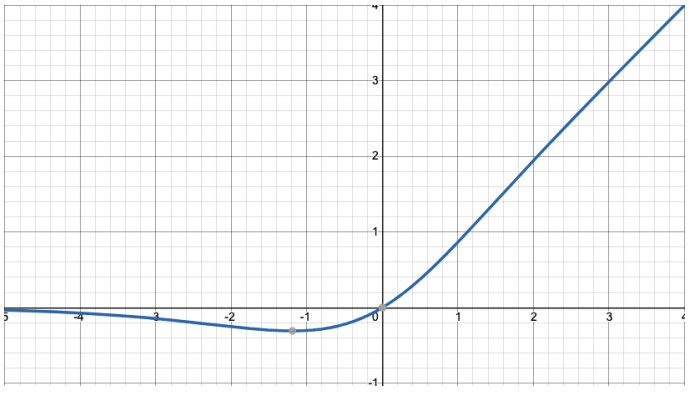

Fig. 6. Mish activation function.

The reason why Mish function is used in YOLOv4 is because of its low cost and its various properties like it’s smooth and non-monotonic nature, unbounded above, bounded below property improves its performance when

compared with other popularly used functions like ReLU.

For batch normalization, the authors use Cross mini-Batch Normalization (CmBN) with the idea that this can be run on any GPU that people use. Many batch normalization techniques require multiple GPUs operating in tandem.

YOLOv4 uses DropBlock [13] regularization. In DropBlock, sections of the image are hidden from the first layer. DropBlock is a technique to force the network to learn features that it may not otherwise rely upon. For example, you can think of a dog with its head hidden behind a bush. The network should be be able to identify the dog from its torso as well as its head.
DropBlock is a structured form of dropout directed at regularizing convolutional networks. In DropBlock, units in a contiguous region of a feature map are dropped together. As DropBlock discards features in a correlated area, the networks must look elsewhere for evidence to fit the data and this can be seen in Fig. 7.

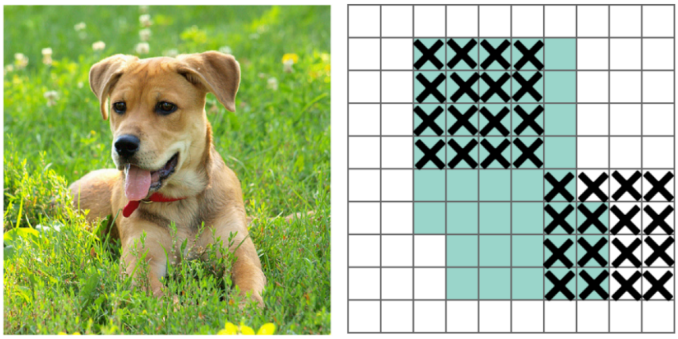

Fig. 7. The dog is an input image to a convolutional neural network and its mask with corresponding blocks which point to be dropped

C. Bag of Freebies

YOLOv4 employs a ”Bag of Freebies” so termed because they improve performance of the network without adding to inference time in production. Most of the Bag of Freebies have to do with data augmentation.The authors of YOLOv4 use data augmentation to expand the size of their training set and expose the model to semantic situations that it would not have otherwise seen. Fig. 8 some data augmentation techniques used in YOLOv4 and among all research community not just YOLOv4.

Many of these strategies were already know to the computer vision community, and YOLOv4 is simply verifying their effectiveness. The new contribution is mosaic data augmentation which tiles four images together, teaching the model to find smaller objects and pay less attention to surrounding scenes that are not immediately around the object.

YOLOv4 represented mosaic which is a new data augmentation method that mixes 4 training images. Thus 4 different

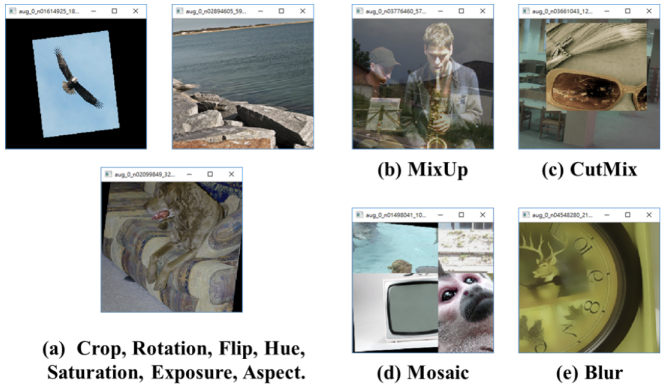

Fig. 8. Data Augmentation.

contexts are mixed, while CutMix mixes only 2 input images. This allows detection of objects outside their normal context. In addition, batch normalization calculates activation statistics from 4 different images on each layer. This significantly reduces the need for a large mini-batch size.

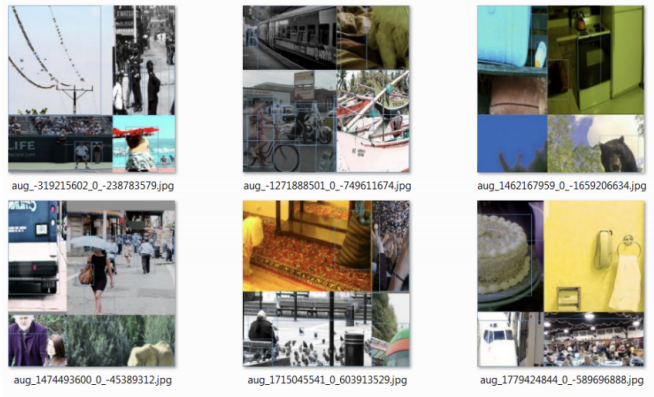

Fig. 9. YOLOv4's Mosaic | single mosaic

III. Methodology

In this paper, different improvements have been made over two dimensions which are batch size and data augmentation.

A. Batch size

Batch Size is among the important hyperparameters in deep Learning. It is the hyperparameter that defines the number of samples to work through before updating the internal model parameters.
In this experiment, we investigated the effect of batch size on training dynamics. There is a lot of research into how different Batch Sizes affect the training from the cost of inference and also from accuracy of the models.

we analyze the results obtained with models trained with different mini-batch sizes, and the results are shown in Table. I. And according to the results provided that the 1st experiment

TABLE I
INFLUENCE OF DIFFERENT MINI-BATCH SIZE ON DETECTOR TRAINING.

	Batch size	mAP	
		Inference	Test
YOLOv4(Baseline)	64	89%	87%
1st experiment	32	91%	89%

with batch size of 32 is better in training the network than 64 which was used originally by YOLOv4.

Fig. 10. OUR mosaic | double mosaic | Mosaic^2

B. Mosaic^2

We have presented the mosaic which was originally introduced by YOLOv4, but here we have added one more step for the proposed method with adding the mosaic 2 times not just one time as YOLOv4 did and the what we wanted from using such a technique is as follows:

- Out of context images which was also provided by mosaic but here it's more challenging or we can say more generic and less prone to be biased towards a specific context form any of the images, which makes our model generalizes better.
- Multiple objects within one image, as here the image is not just a single image with few objects but now there are multiple objects within each image and even half objects which makes it more challenging for the model to learn, detect and localize these objects.
- Challenges for the image to detect small objects and objects with insignificant key points, this point has already been here with the challenges of the last two points but deserved to be a separate one and even have our attention till making us name the paper after it, "No Pain No Gain", as these challenges provide a whole new dimensions for our model to learn not just the naive features and to localize and detect the clear objects but to pay attention for even the Small details which are hidden within these objects to better detect them.

Fig. 10 shows 2 examples of double mosaic which is our proposed methodology throughout data augmentation, and this figure shows 2 examples, side by side, in each example some of the challenges are clear like having part of human faces not the whole face unlike the original dataset which gives the data with clear and in position human faces which doesn't happen by nature especially when detecting these images with surveillance cameras which people don't pose for. Also having human parts but not with the mouse area or with part of it presents a bigger challenge.
To summarize this point, we can't even list the amount of poses that this methods proposed with such a small dataset to be a richer with images and have a more image with more normal and abnormal images to work on it which make the model stand still and effective against abundance of strange poses.

IV. DATASET

This dataset contains 1510 images belonging to 2 classes, The classes are:

- With mask
- Without mask

And their bounding boxes in the YOLO format labeled text files.

V. RESULTS AND EVALUATION

As presented before, there are 3 different scenarios that we evaluated and these are:

- Single Mosaic + 64 batch size(original YOLOv4 | baseline).
- Single Mosaic + 32 batch size.
- Double Mosaic + 32 batch size.

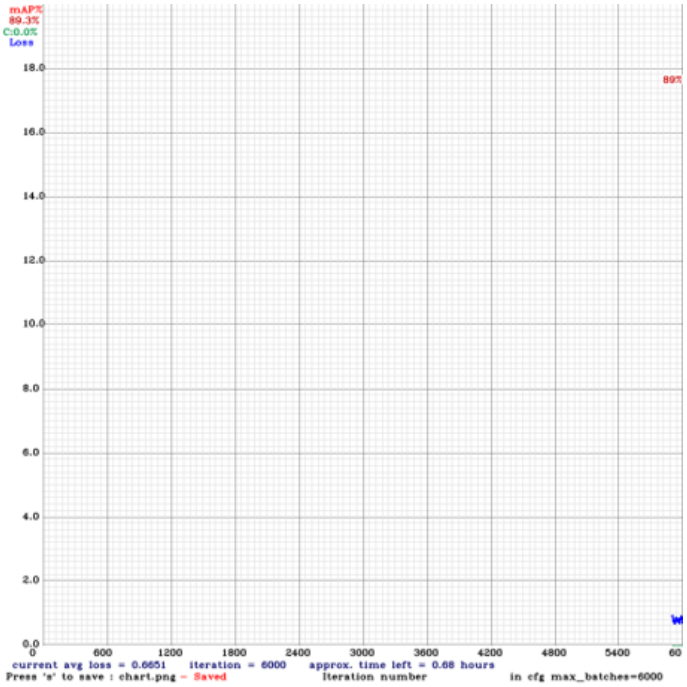

Fig. 11. training of original YOLOv4 | Single Mosaic + 64 batch size | baseline)

TABLE II
INFLUENCE OF DIFFERENT MINI-BATCH SIZE & MOSAIC ON DETECTOR TRAINING.

	Batch size	Mosaic	mAP	
			Inference	Test
YOLOv4(Basline)	64	Single	89	87%
1st Experiment	32	Single	91%	89%
OUR Methodology	32	Double	97%	94

First, we would present the original results of YOLOv4 as our baseline and to compare our results with it and from Fig. 11 we can see only the last iterations of the training and that's because of using colab to train and the run-time disconnected multiple times and we didn't save the graph on the first time but we learned our lesson and saved it the other 2 times.

Throughout this graph, we can see that the training mAP for the original YOLOv4 with no change on its parameters so it's a baseline to compare our results with, we can see that the training mAP has reached 89%.

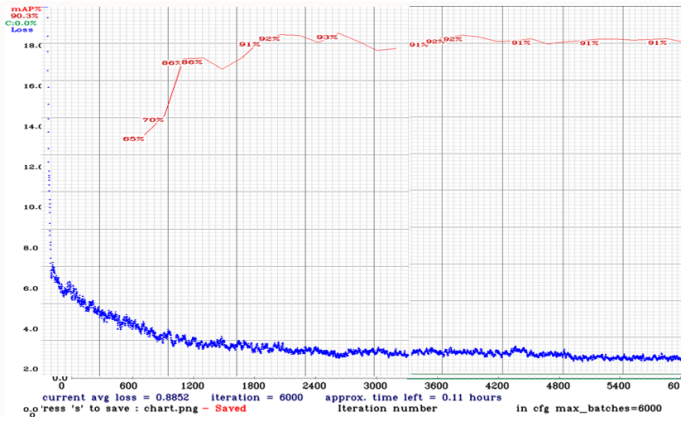

Fig. 12. training of 1st Experiment | Single Mosaic + 32 batch size)

And for our 1st experiment we can see Fig. 12 and we could recognize from the first glimpse the difference in training mAP from the baseline as here it has arrived to 91% which is higher than the baseline with 2%.

And for our proposed methodology, we can look at Fig. 13 which gives us the thrill to something great as the training has reached 97% for mAP which is promising and achieved what we were expecting or even better.

And to analyze these difference not just only on training but also in testing these results with test data that splitted from our original data we may look at Table. II.

VI. QUALITATIVE RESULTS

Figures Fig. 14 and Fig. 15 are 2 examples of our method running on detection of mask on one of the authors.

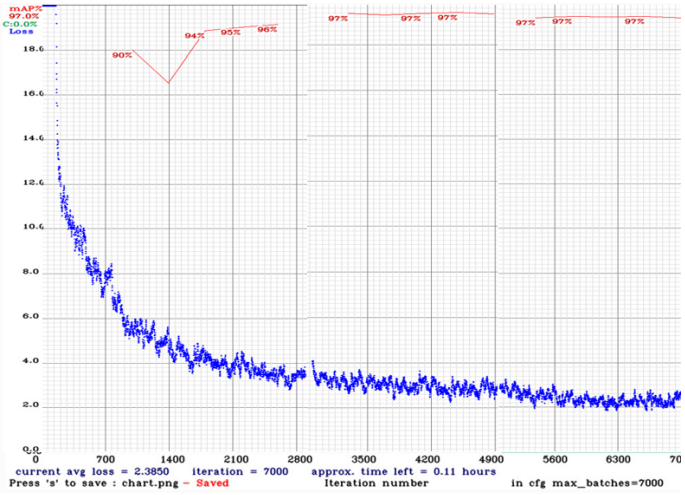

Fig. 13. training of 2nd Experiment | Double Mosaic + 32 batch size | Our Methodology)

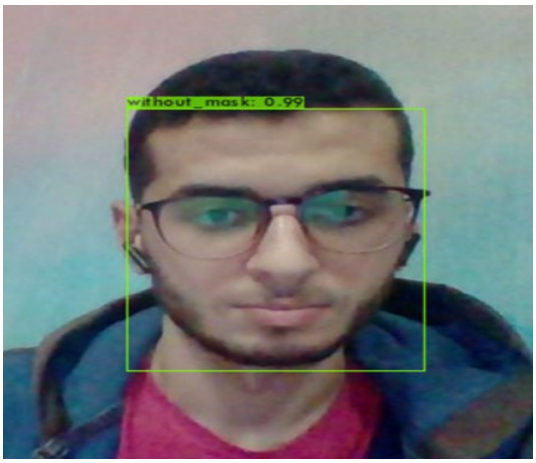

Fig. 14. OUR method detection for a non-masked human)

Fig. 15. OUR method detection for a masked human

VII. Conclusion

Using 32 batch size rather than 64 has done a good change in results but need to a further look on that point for multiple contexts but using double mosaic has once proven 2 facts:

- No Pain No Gain, that was the title for our paper and we would like to emphasize on that point one more time to say that make the challenges for our model made it comes to a new point that it wouldn't have arrived to without these challenges.
- Having more data in deep learning would make us reach frontiers that we only dream right now to reach but also working with available data and manipulating it to suit us better and give better results with the understanding for the whole picture is challenging but proves everyday that it makes a huge difference.

At the end, challenge your model to the edge.

Acknowledgment

We want to acknowledge the work done in the deep learning course that make us arrive to that point right now, to once prove again that making hopes on people and directing them to test their limits just make them realize that they don't have limits and they were just imaginary limits and they start pursuing a new frontiers.
For Prof. Mohamed A. Naiel.

References

[1] Bochkovskiy, A., Wang, C., Liao, H.M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. ArXiv, abs/2004.10934.

[2] Tsung-Yi Lin et al., 2014. Microsoft COCO: Common Objects in Context. CoRR, abs/1405.0312. Available at: http://arxiv.org/abs/1405.0312.

[3] Joseph Redmon. (2013–2016). Darknet: Open Source Neural Networks in C.

[4] Deng, J. et al., 2009. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255.

[5] Chien-Yao Wang, Hong-Yuan Mark Liao, I-Hau Yeh, Yueh-Hua Wu, Ping-Yang Chen, and Jun-Wei Hsieh. (2019). CSPNet: A New Backbone that can Enhance Learning Capability of CNN.

[6] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection.

[7] Mingxing Tan, and Quoc V. Le. (2020). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks.

[8] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger 2016. Densely Connected Convolutional Networks. CoRR, abs/1608.06993.

[9] Liu, S., Qi, L., Qin, H., Shi, J., and Jia, J. 2018. Path aggregation network for instance segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 8759–8768).

[10] He, K., Zhang, X., Ren, S., and Sun, J. 2015. Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE transactions on pattern analysis and machine intelligence, 37(9), p.1904–1916.

[11] Woo, S., Park, J., Lee, J.Y., and Kweon, I. 2018. Cbam: Convolutional block attention module. In Proceedings of the European conference on computer vision (ECCV) (pp. 3–19).

[12] Redmon, J., and Farhadi, A. 2018. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.

[13] Ghiasi, G., Lin, T.Y., and Le, Q. 2018. Dropblock: A regularization method for convolutional networks. Advances in neural information processing systems, 31.