Faculty of Computers and Artificial Intelligence Helwan university

Jan 2022

# skin disease smart detection using yolov5 especially version yoloV5s

Mohamed Elredeny

**ARTICLE INFO**

*Keywords:*
Skin diseases
deep learning
object detection
yolov5
vitiligo
melanoma

**ABSTRACT**

Through their system, users can upload their images to check their skin and just press a button and through our application that is connected with an API, their model detects this image through different stages according to object detection principles classification to detect whether the uploaded images contained any of the mentioned diseases or not then localization to know exactly where the disease is located then create a bounding box around them.

Our system works with best mAP reached of value 0.961 of correct classification and localization as well.

Through our system, users can see their last detections, they can also find a community through which they can ask more about their disease and see doctors' recommendations.

asymmetrical shapes, irregular border, changes in color, growth in diameter and evolving size, color or shape. 25% of Melanomas develop from moles, so when previous discussed characteristics develop in mole, it could be a sign of Melanoma. Not only Melanoma occurs due to DNA damage resulting from exposure to ultraviolet rays from the sun but also family history can

asymmetrical shapes, irregular border, changes in color, growth in diameter and evolving size, color or shape. 25% of Melanomas develop from moles, so when previous discussed characteristics develop in mole, it could be a sign of Melanoma. Not only Melanoma occurs due to DNA damage resulting from exposure to ultraviolet rays from the sun but also family history can play a role in having this disease as well, Whereas Vitiligo, there's lack in Melanin in

play a role in having this disease as well, Whereas Vitiligo, there’s lack in Melanin in

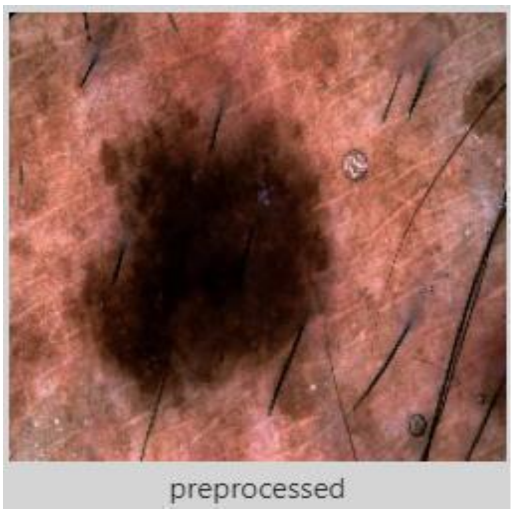
## 2.2. Image data augmentation

When we talk about the data augmentation, we ask ourselves a question How do I get more data, if I don’t have “more data”? deep learning models need data and the more data your model learn the better detection your model will produce. There were a lot of techniques

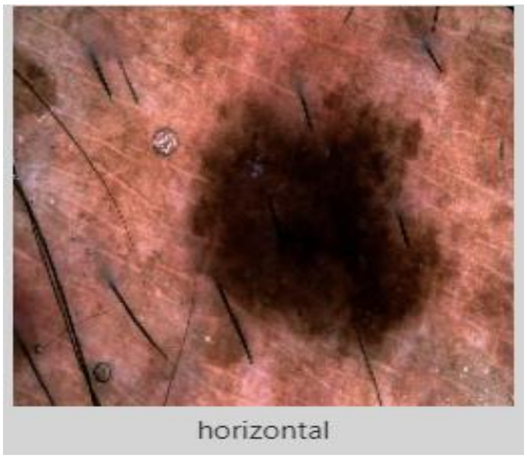
we may use to perform augmentation for our dataset to increase its size and we implement the following.

### 2.2.1 Image Flipping

we are trying to extend our dataset and give our model a correct data to learn so this technique was very useful for us, through image flipping Technique we have flip our pure image:



1.horizontal:



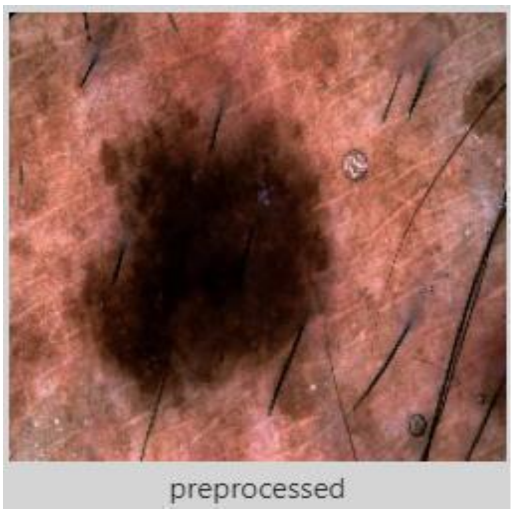
2.2. Image data augmentation

When we talk about the data augmentation, we ask ourselves a question How do I get more data, if I don’t have “more data”? deep learning models need data and the more data your model learn the better detection your model will produce. There were a lot of techniques

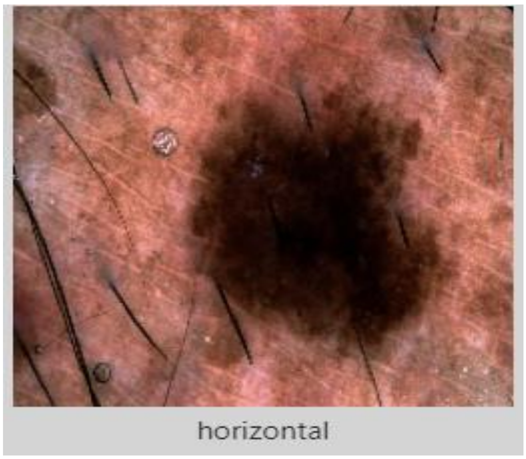
we may use to perform augmentation for our dataset to increase its size and we implement the following.

2.2.1 Image Flipping

we are trying to extend our dataset and give our model a correct data to learn so this technique was very useful for us, through image flipping Technique we have flip our pure image:



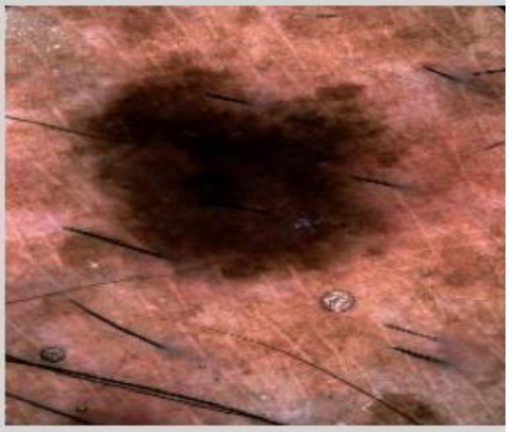
1.horizontal:



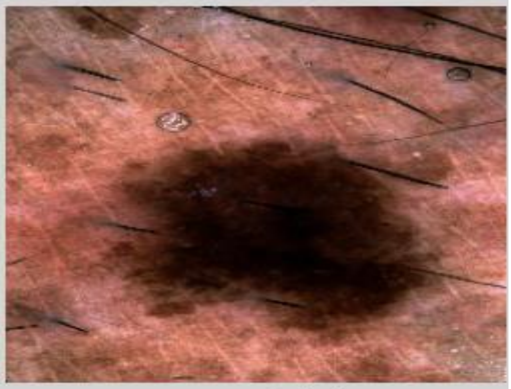
2.2.3 Rotate 90

in the rotate 90 technique we have implemented three ways for our original image.

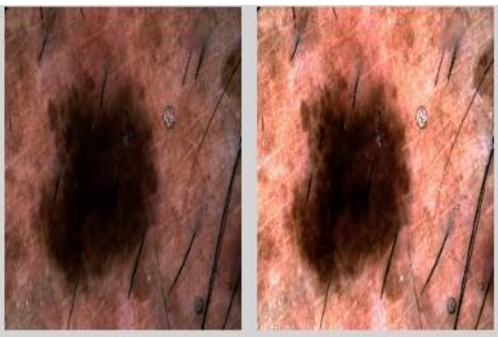
1. Clockwise


clockwise

2. Counter-Clockwise


counter-clockwise

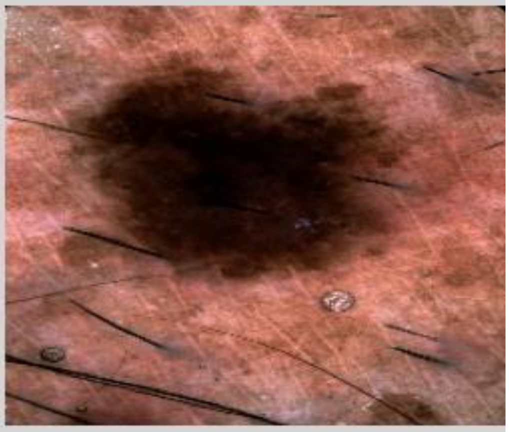
4. Brightness


-25% 25%

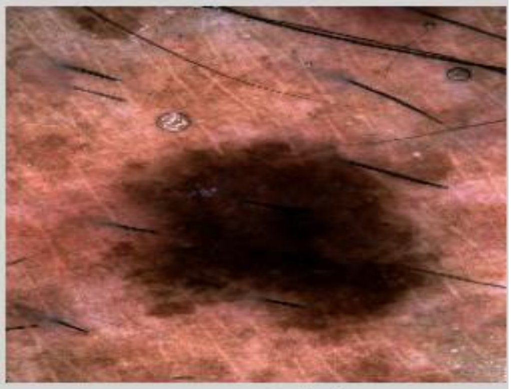
2.2.3 Rotate 90

in the rotate 90 technique we have implemented three ways for our original image.

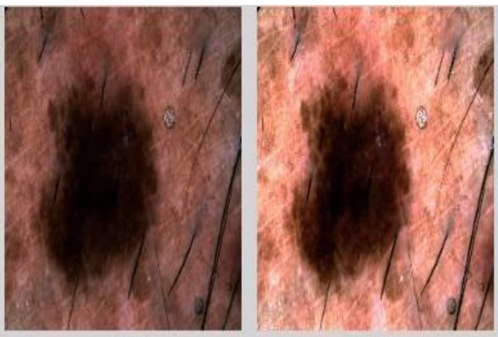
1. Clockwise


clockwise

2. Counter-Clockwise

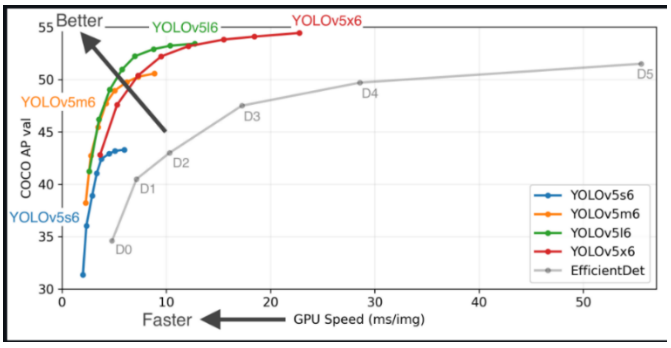

counter-clockwise

4. Brightness


-25% 25%

## 3. Methodologies

### 3.1. YOLO-V5



The YOLO-V5 network has been introduced by Glenn Jocher (the founder and CEO of Ultralytics ) shortly after the release of YOLOv4 . Its implementation has been done in Pytorch, in contrast with the previous developments that used the Darknet framework. This makes it easier to understand, train with it and deploy this model, so YOLO-V5 is said to be the state of the art among all known YOLO implementations according to the Ultralytics GitHub page (https://github.com/ultralytics/yolov5).

YOLO-V5 is be able to achieve fast detection at 140FPS running on a Tesla P100 in comparison to YOLOv4 which bench-marked at a measly 50 FPS stated on an article published on the Roboflow blog (https://blog.roboflow.ai/yolov5-is-here).

The largest contribution of YOLOv5 is to translate the Darknet research framework to the PyTorch framework. The Darknet framework is written primarily in C and offers fine grained control over the operations encoded into the network. In many ways the control of the lower-level language is a boon to research, but it can make it slower to port in new research insights, as one writes custom gradient calculations with each new addition.

As YOLO v5 is a single-stage object detector, it has three important parts like any other single-stage object detector which are Model Backbone, Model Neck and Model Head.

Model Backbone is mainly used to extract important features from the given input image. In YOLO v5 the CSP — Cross Stage Partial Networks are used as a backbone to extract rich in informative features from an input image. CSPNet has shown significant improvement in processing time with deeper networks.

Model Neck is mainly used to generate feature pyramids. Feature pyramids help models to generalized well on object scaling. It helps to identify the same object with different sizes and scales. Feature pyramids are very useful and help models to perform well on unseen data. There are other models that use different types of feature pyramid techniques like FPN, BiFPN, PANet, etc. In YOLO v5 PANet is used as neck to get feature pyramids.

The model Head is mainly used to perform the final detection part. It applied anchor boxes on features and generates final output vectors with class probabilities, objectness scores, and bounding boxes. In YOLO v5 model head is the same as the previous YOLO V3 and V4 versions.

### 3.2. Cross Stage Partial Network (CSPNet)

The problem of excessive inference calculations is caused by the duplication of gradient information in network optimization. CSPNet integrates the gradient changes into the feature map from beginning to end, which reduces the amount of calculation while ensuring accuracy. CSPNet is a processing idea that can be combined with ResNet, ResNeXt and DenseNet. It represents a from the perspective of network structure design to solve the problem of requiring a large amount of calculation in the inference process of previous work.

CSPNet proposed mainly to solve three problems including, Enhancing the learning ability of CNN can maintain accuracy while reducing weight, Reduce computing bottlenecks and Reduce memory costs

Each stage of a DenseNet contains a dense block and a transition layer, and each dense block is composed of k dense layers. The output of the ith dense layer will be concatenated with the input of the ith dense layer, and the concatenated outcome will become the input of the (i + 1)th dense layer. The equations showing the above-mentioned mechanism can be expressed as:

$$
\begin{aligned}
x_1 &= w_1 * x_0 \\
x_2 &= w_2 * [x_0, x_1] \\
&\vdots \\
x_k &= w_k * [x_0, x_1, \ldots, x_{k-1}]
\end{aligned}
$$

where ∗ represents the convolution operator, and [x0, x1, ...] means to concatenate x0, x1, ..., and wi and xi are the weights and output of the ith dense layer, respectively. If one makes use of a backpropagation algorithm to update weights, the equations of weight updating can be written as:

$$\begin{aligned} w_1' &= f(w_1, g_0) \\ w_2' &= f(w_2, g_0, g_1) \\ w_3' &= f(w_3, g_0, g_1, g_2) \\ &\vdots \\ w_k' &= f(w_k, g_0, g_1, g_2, \ldots, g_{k-1}) \end{aligned}$$

where f is the function of weight updating, and gi represents the gradient propagated to the i th dense layer. We can find that large amount of gradient information are reused for updating weights of different dense layers. This will result in different dense layers repeatedly learn copied gradient information.

A stage of CSPDenseNet is composed of a partial dense block and a partial transition layer. In a partial dense block, the feature maps of the base layer in a stage are split into two parts through channel x0 = [x 0’’ , x 0 ].

$$\begin{aligned} x_k &= w_k * [x_{0''}, x_1, \ldots, x_{k-1}] \\ x_T &= w_T * [x_{0''}, x_1, \ldots, x_k] \\ x_U &= w_U * [x_{0'}, x_T] \end{aligned}$$

$$\begin{aligned} w_k' &= f(w_k, g_{0''}, g_1, g_2, \ldots, g_{k-1}) \\ w_T' &= f(w_T, \boxed{g_{0''}, g_1, g_2, \ldots, g_k)} \\ w_U' &= f(w_U, \boxed{g_{0'}, g_T)} \end{aligned}$$

### 3.3. Feature Pyramid Network (FPN) using Path Aggregation Network (PANet)

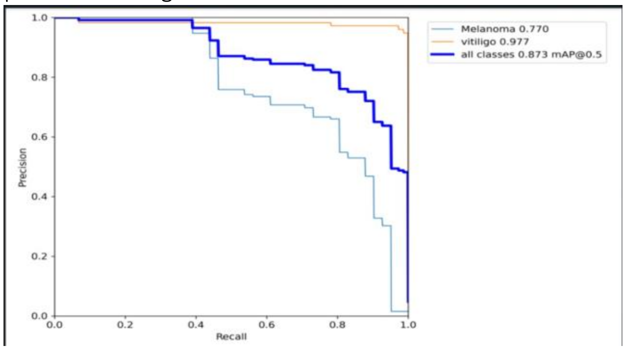
is a feature extractor designed for such pyramid concept with accuracy and speed in mind. It replaces the feature extractor of detectors like Faster R-CNN and generates multiple feature map layers (multi-scale feature maps) with better quality information than the regular feature pyramid for object detection.

FPN composes of a bottom-up and a top-down pathway. The bottom-up pathway is the usual convolutional network for feature extraction. As we go up, the spatial resolution decreases. With more high-level structures detected, the semantic value for each layer increases.

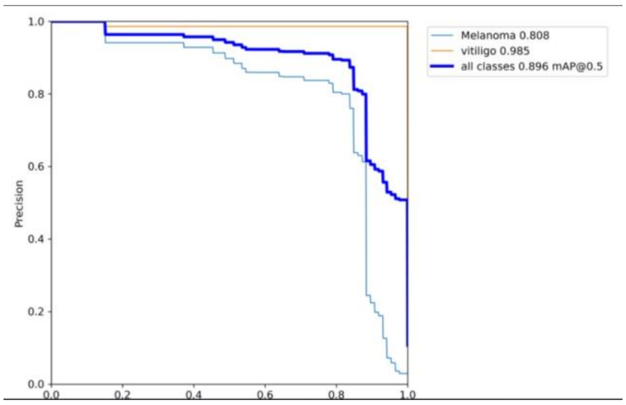
## 4. Experiments and discussion

We have implemented several techniques trying to improve the overall accuracy and we will explain exactly what we have done. We have selected a trussed places to take our dataset images from, we also meant to make our images' objects are big enough to our model and with high quality. We have realized best free place to annotate your data automatically is rob flow, through which you can upload your images and start to draw your bounding boxes around objects easily. after preparing our dataset we have selected the most appropriate techniques for us to implement augmentation to our data.
After Becoming able to start training our model, we Exactly passed through three stages

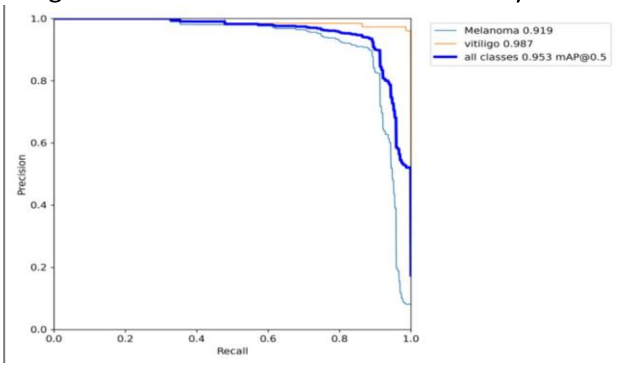
**First Stage:**
We set our images size to be 416px for every entry image and divide our train into 100 epochs with using yolov5s pretrained weights



**Second Stage:**
We set our images size to be 640px for every entry image and divide our train into 200 epochs with using Our first model best results as our weights and added around 300 images to our dataset trying to increase melanoma accuracy



**Last Stage:**
We set our images size to be 640px for every entry image and divide our train into 200 epochs with using Our Second model Best results as our weights and added 900 image to melanoma to increase its accuracy



## 5.Conclusion

before starting an object detection project, you have to put into your consideration that collecting datasets from trusted places is the most important phase your images must be labelled in specific formats in our case YOLO our format is "darken format" objects must be clear and big enough. the more images you learn your model the best map you will find the training phase you have to train your model and change your epochs number patches and images width and height in addition to using your last best weighs from your last model

## 6.Refrences

1. GitHub. (2020). *ultralytics/yolov5*. [online] Available at: https://github.com/ultralytics/yolov5.

2.Nelson, J., Jun 10, J.S. and Read □ ⍰ 〈 ◊2020 4 M. (2020). *YOLOv5 is Here*. [online] Roboflow Blog. Available at: https://blog.roboflow.ai/yolov5-is-here.

3. Failla, C.M., Carbone, M.L., Fortes, C., Pagnanelli, G. and D'Atri, S. (2019). Melanoma and Vitiligo: In Good Company. *International Journal of Molecular Sciences*, 20(22), p.5731.https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6888090/

3. www.medscape.com. (n.d.). *What causes vitiligo in malignant melanoma?* [online] Available at: https://www.medscape.com/answers/1068962-
4. 87927/what-causes-vitiligo-in-malignant-melanoma [Accessed 13 Aug. 2021].https://www.medscape.com/answers/1068962-87927/what-causes-vitiligo-in-malignant-melanoma

5. ***Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi***; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779-788

6. Velasco, J. (2019). A Smartphone-Based Skin Disease Classification Using MobileNet CNN. *International Journal of Advanced Trends in Computer Science and Engineering*, pp.2632–2637.

7. Kasper-Eulaers, M., Hahn, N., Berger, S., Sebulonsen, T., Myrland, Ø. and Kummervold, P.E. (2021). Short Communication: Detecting Heavy Goods Vehicles in Rest Areas in Winter Conditions Using YOLOv5. *Algorithms*, 14(4), p.114.
8. Tian, Y., Yang, G., Wang, Z., Wang, H., Li, E. and Liang, Z. (2019). Apple detection during different growth stages in orchards using the improved YOLO-V3 model. Computers and Electronics in Agriculture, [online] 157, pp.417–426. Available at: https://www.sciencedirect.com/science/article/pii/S016816991831528X [Accessed 20 Nov. 2019].