ENHANCEMENT OF AI-BASED IMPLEMENTATION USING A ONE-STAGE DETECTOR ALGORITHM FOR THE DETECTION OF COUNTERFEIT PRODUCTS

Eduard Daoud, Nabil Khalil and Martin Gaedke
Technische Universität Chemnitz, Germany

ABSTRACT

Counterfeit products are a major problem that the market has been facing for a long time. According to the Global Brand Counterfeiting Report 2018 "Amount of Total Counterfeiting, globally has reached to 1.2 Trillion USD in 2017 and is Bound to Reach 1.82 Trillion USD by the Year 2020", a solution to this concern has already been researched and published by the authors in previous research papers published in e-society 2020 and IADIS journal, but the issue with the previously mentioned solution was that the object detection performance and accuracy needed to be improved. In this paper, a comparison between the current YOLO (You Only Look Once) algorithm used in the new implementation and the SSD (Single Shot Detector) algorithm, the faster R-CNN (Region-Based Convolutional Neural Networks) used in the old implementation, is made in the context of the present task to discuss and prove why YOLO is a more suitable option for the counterfeit product detection task.

KEYWORDS

Anti-Counterfeiting, Machine Learning, Deep Learning, Image Recognition, Object Detection

1. INTRODUCTION AND CURRENT PROBLEM

Counterfeited products have been a huge problem to the whole world market for a long time, there have been many types of research focused on solving this specific problem and only one solution was found proposing the use of machine learning and object detection to allow end-user to be able to verify and authorize products using only an image that contains the product quality certificate logos and marks. This solution was published by Eduard, et al. (2020) and it uses SSD by Liu, et al. (2016) and Faster R-CNN by Ren, et al. (2016) machine learning algorithms for detecting logos and marks in images. Logos and marks in product images are likely to be small, and these images will not always be in a high resolution, and that is one of the difficulties that make it hard for the SSD algorithm to detect small objects efficiently. Before we start, we will explain the reason why YOLOv4 by Alexey & Chien-Yao (2020) is a suitable option. Two different architectures are used in object detection tasks; the first one is the two-stage detectors which include the famous regional-based detectors like R-CNN by Girshick, et al. (2013), Fast R-CNN by Girshick (2015), and Faster R-CNN which are used for different tasks. The other object detection architecture is the one-stage detector, which includes SSD and YOLOv4, even though SSD and YOLO are using the same architecture, the way the network in each algorithm is constructed and the way they work is different. The differences between the one-stage detector and the two-stage detectors as well as the difference between SSD and YOLOv4 will be explained in the next sections.

2. STATE OF THE ART ANALYSIS

According to Alexey & Chien-Yao (2020), there are two popular types of object detectors, two-stage detectors, and one-stage detectors.

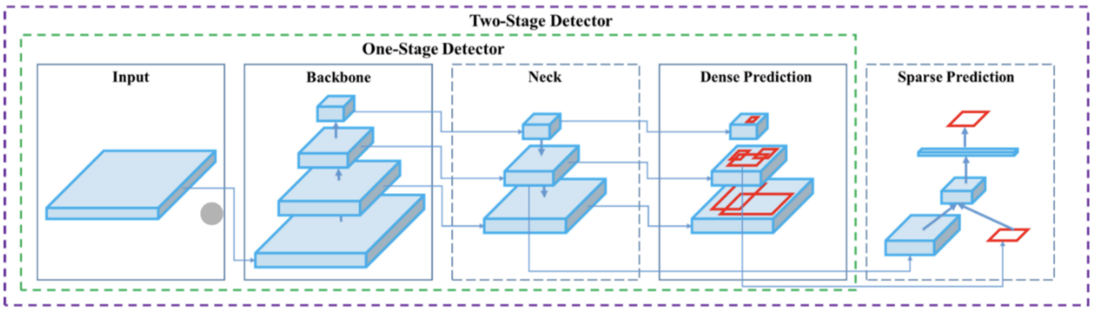


Figure 1. The difference between the two-stage and one-stage detectors according to Alexey & Chien-Yao (2020)

The image above illustrates the difference between the two-stage and the one-stage detector. The two-stage detector focuses on Faster R-CNN implementation by Girshick (2015) and the single-stage detector focuses on SSD and YOLOv4 implementation by Alexey & Chien-Yao (2020). The CertificateOk platform resulting from the research of Eduard, et al. (2020) uses the Faster R-CNN implementation and the proposed new implementation uses the YOLOv4 implementation as a framework. As in Manuel, et al. (2020) & Shilpa, et al. (2021), the Faster R-CNN contains two main elements: The region proposal network (RPN), and the Fast R-CNN header network. At the beginning of the detection, a CNN is used to extract the features on the image as a whole; in previous versions of R-CNN, feature extraction was applied for each region of interest (ROI) separately, which increases the inference time. Then the RPN utilizes the sliding windows' technique on the output of the feature map to generate proposals on each location with anchors (reference boxes) that were on the feature map and predict whether this location contains an object or not. The anchors that the RPN uses come in different shapes and sizes to be able to correctly predict whether there is an object in the given region of interest or not. After that, all the detected regions will be passed through a network to calculate the score of having an actual object in that region. The final step of the first stage is having the top regions of interest cropped using the ROI pooling layers. In the second stage, the results of the ROI pooling will be sent to the fully connected Fast R-CNN network for classification and localization, which is referred to in Figure 1 as Sparse Prediction.

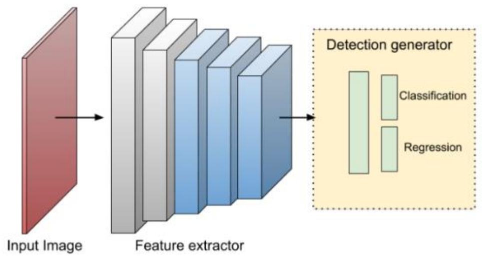


Figure 2. One-stage detector according to Manuel, et al. (2020)

On the other hand, there is the one-stage detector which includes SSD and YOLO algorithms. As shown in Figure 1 and Figure 2 the one-stage detectors detect the bounding boxes as well as the predicted class in one step. The one-stage detector, single-shot detector, or SSD by Liu, et al. (2016) uses VGG-16 as a feature extractor, which is the same feature extractor used by Faster R-CNN by Girshick (2015), VGG-16 is a convolutional neural network model first proposed by K. Simonyan and A. Zisserman of the University of Oxford in their work "Very Deep Convolutional Networks for Large-Scale Image Recognition". SSD inference starts by dividing the image into multiple grids and then for each grid it makes prediction boxes, with the anchor of these boxes being in the middle of the grid. And predicts the existence of all the object classes that we have. As shown in Figure 3 SSD predicts with different convolutional sizes and sends the predicted classes to the final overall prediction to be able to improve accuracy.

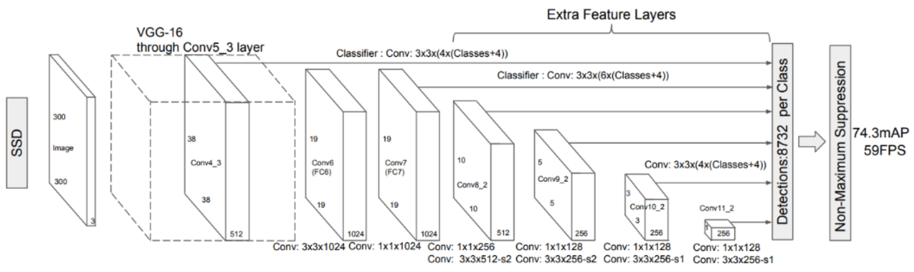


Figure 3. Single-shot detector as in Liu, et al. (2016)

One of the challenges and problems with SDD, as described in Liu, et al. (2016), is that the detection step leads to numerous prediction boxes. This problem is solved by non-maximum suppression, which is a technique for combining multiple boxes belonging to the same object Jan, et al. (2017).

The second one-stage detector and the one that will be the focus of this research is 'you only look once' or YOLOv4 by Alexey & Chien-Yao (2020). YOLO by Joseph, et al. (2016) was introduced in 2016 as a state-of-the-art real-time detection algorithm, and since then the development and improvement of YOLO have never stopped. The way YOLO works is a little different from the way SSD by Liu, et al. (2016) works, YOLO starts by dividing the image into an S × S grid, for each grid a B number of bounding boxes are created along with the confidence in these boxes, if the object center was detected in one of the grids, that grid will be in charge of detecting the whole object, each bounding box inside the grid will contain 5 predictions (x, y, h, w, and c) while X and Y represent the center of this box, H and W are the height and the width and C is the confidence, in the end for the whole YOLO predictions can be put as (S×S×(B×5 + C)), and for each grid, only one prediction will be constructed with each class probability, this can be expressed like (B×5+C) Joseph, et al. (You Only Look Once:Unified, Real-Time Object Detection, 2016).

As an example, from our case - detection fake product -, shown in Figure 4 we detect the TUV SUD logo so the grid that contains the logo will result in Pc =1 which means that there is an object in this grid and the (x, y, h, w) for this logo and the classes probabilities which will be zero for the first and last class and 1 for the second class which indicates that the detected object belongs to the second class.

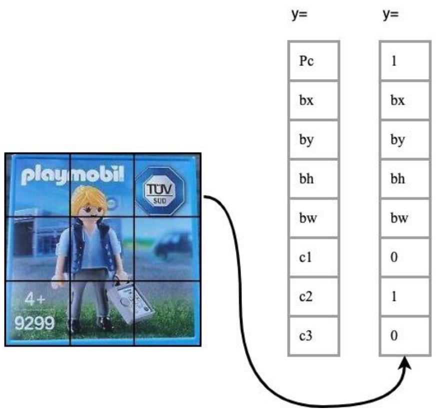


Figure 4. Detect a TÜV SÜD logo

YOLOv2 by Joseph & Ali (2016) included some improvements like 'Higher Resolution Classifier', 'Multi-Scale Training', 'Fine-Grained Features' and Darknet-19 architecture that include 19 convolutional layers as a feature extractor, alongside other improvements that increased the accuracy and the performance of YOLOv2 over YOLOv1 and made the detection of smaller objects much better.

YOLOv3 by Redmon & Farhadi (2019) featured huge improvements in the feature extractor as it uses Darknet-53 has 53 convolutional layers which is much deeper than Darknet-19 which was used in YOLOv2. YOLOv3 makes predictions like Feature Pyramid Networks or FPN Tsung-Yi, et al. (2017), FPN constructs two pathways, bottom-up which increases the semantic value but with lower resolution, and top-down which offers higher resolution but low semantic value. Figure 5 shows the way FPN works, the top level is a low-resolution high-semantic representation that helps detect larger objects, and the lower level is a high-resolution layer that helps detect smaller objects, and in the middle layers connections between the reconstructed layers and feature maps is made for better location accuracy Tsung-Yi, et al. (2017).

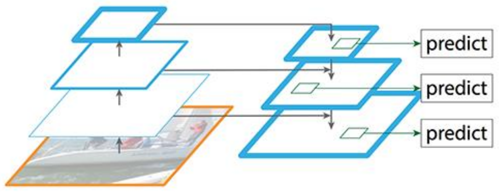


Figure 5. Feature pyramid networks as in Tsung-Yi, et al. (2017)

While YOLOv3 uses multiple layers like FPN and also uses the high-resolution layer, SSD by Liu, et al. (2016) only uses the upper layers for inference and does not reuse the higher-resolution feature maps as shown in Figure 6 and that makes SSD a bad choice for detecting small objects like a logo or quality marks Tsung-Yi, et al. (2017).

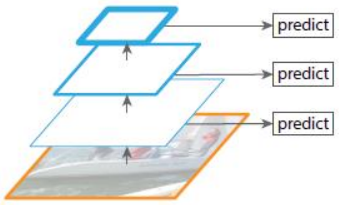


Figure 6. Pyramidal feature hierarchy as in Tsung-Yi, et al. (2017)

YOLOv4 by Alexey & Chien-Yao (2020) improves the mean average precision (mAP) by 10% over YOLOv3, and it includes changes in different aspects of the algorithm, first YOLOv4 uses CSPDarknet53 as a feature extractor which is a feature extractor that can learn, it takes the input and divides that input into two parts, one part will go through the dense layer and the other will be concatenated at the end to get the final dense layer result Chien-Yao, et al. (2020), YOLOv4 also added a spatial pyramid pooling (SPP) Kaiming, et al. (2015) after the CSPDarknet53 to be able to generate fixed-length features regardless of the size of the feature maps. A key change in YOLOv4 is that it uses a modified Path Aggregation Network (PANet) Shu, et al. (2018) instead of the FPN that was used in YOLOv3, PANet tries to enhance the way FPN works and increase its accuracy and performance. PANet uses “FPN backbone, bottom-up augmentation, adaptive feature pooling, Box branch, and a fully connected fusion” Shu, et al. (2018).

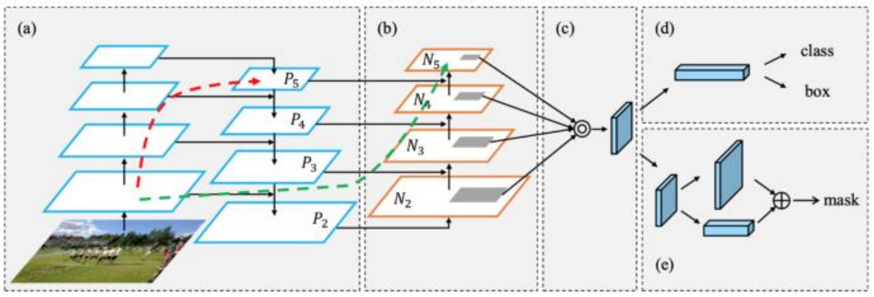


Figure 7. Path aggregation network architecture as in Shu, et al. (2018)

As shown in Figure 7 part (a) and (b) shows the “FPN backbone” and the “bottom-up path augmentation” this helps preserve the spatial information while keeping the length of the layers short. Part (c) is the “adaptive feature pooling” it uses features from all layers and lets the network choose the useful features; it also performs “ROI Align operation” on feature maps. (d) and (e) represent the “Fully-connect Fusion” which is used to make accurate mask prediction with a high location sensitivity. YOLOv4 uses PANet with a small modification, instead of adding layers and forming a new layer YOLOv4 preform concatenation between the input layer and the previous one.

Based on this evidence and insights, this paper implements a new performant and user-friendly solution. In the next section, we present the concept behind the new solution and the technical architecture of the application and then go into more detail about the implementation phase.

3. CONCEPT AND TECHNICAL ARCHITECTURE

The approached application is developed in a way that allows the end-user to easily verify products using smartphones, cameras, or a picture from the internet. This allows a high majority of end-consumers to detect fake products.

The application will receive the image that contains the logo and possible marks in an HTTP request; after that, the image will be sent to OpenCV YOLOv4 trained model for detection. After detection is completed, the results will be sent to the user in a JSON Response so that the user can easily view the results on any device.

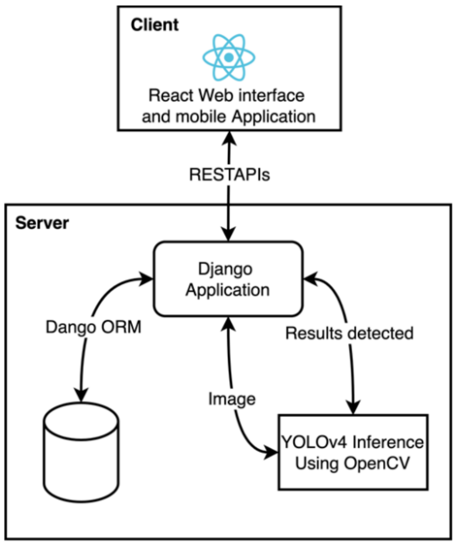

Client
React Web interface and mobile Application
RESTAPIs
Server
Django Application
Dango ORM
Results detected
Image
YOLOv4 Inference Using OpenCV

Figure 8. The concept of the proposed application

As shown in Figure 8, the client-side can be a ReactJS application on the web or React Native application on smartphones. The user will send an image from the client-side to the web servers via Rest APIs along with other information (metadata, request user info). The server-side has two main parts, the Django application, which will receive the request and extract the necessary data from it, and the OpenCV deep learning inference using YOLOv4. To verify the concept as a whole and the proposed application, which focuses in particular on the detection of counterfeit products, we implement the solution in the next section and train the model with the logo of the certification body and the quality mark of TÜV SÜD AG as one of the largest European certification bodies and VDE e.V. as one of the largest technical-scientific associations in Europe in the field of safety and quality in electrical, information and medical technology.

4. IMPLEMENTATION

Certain steps are necessary to build a high-efficiency machine learning solution that can identify labeled logos/marks correctly from product images. The first step is to decide which computer vision algorithm to use, in our case YOLOv4 seems the most reasonable algorithm because of its high ability to detect a small object and its very high performance compared to other algorithms. After that, the dataset should be created including both correct and wrong logos/marks to use this dataset later in the training process.

We are going to use the dataset that was used in the previous implementation by Eduard, et al. (2020) of CertificateOk 1 and improve it a bit by adding more samples, the dataset that was used contained around 600 images of different shapes and different logo/marks dimensions. After creating the dataset, a suitable labeling tool was used like YOLO_Label2, which is an open-source labeling tool that creates label files easily and efficiently. After labeling the whole dataset it is recommended to split the dataset by 80% for training and 20% for testing and validation. We used the transfer learning technique which is using pre-trained weights file in our case YOLOv4 pre-trained weights to speed up the training process and enhance the accuracy and performance.

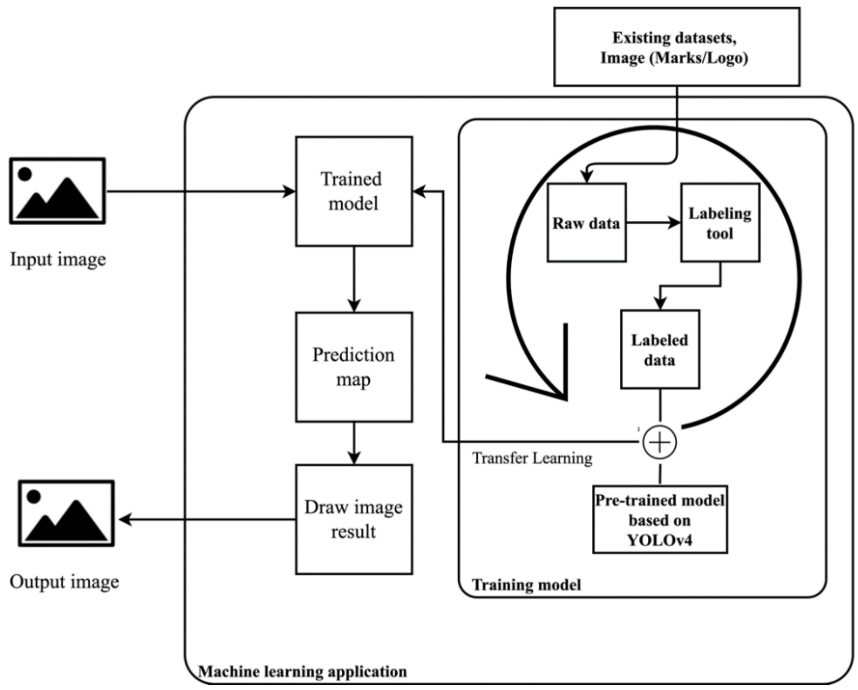


Figure 9. Implementation to YOLOv4 model adapted from Eduard, et al. (2020)

The training step is a continuous step, which means it can be repeated using saved weights to increase accuracy until the model achieves its highest mean Average Precision (mAP).

The trained model weights will be used by OpenCV3 deep learning inference since OpenCV has a good integration with darknet4 which is an open-source neural network that is used in building YOLOv4. The backend will be implemented using Django5 which is an open-source python web framework and integrating OpenCV with Django OpenCV-python library will be used. On the client-side ReactJS will be used for creating a customizable application with high stability. In the following section, we evaluate our approach before we headline the summary and an outlook on our work.

5. EVALUATION AND CHALLENGES

The proposed solution to fight counterfeit products has several advantages over normal counterfeit counters, it empowers the end-user to be able to verify and report products without the need for any special tools, only an image of the product that contains the logo/marks of the manufacturer is considered enough for our machine learning application to decide if the product is counterfeit or original.

[1] https://app.certificateok.de/Consumer accessed at 9th of January 2022
[2] https://github.com/developer0hye/Yolo_Label accessed at 9th of January 2022
[3] https://opencv.org/ accessed at 9th of January 2022
[4] https://pjreddie.com/darknet/ accessed at 9th of January 2022
[5] https://www.djangoproject.com/ accessed at 9th of January 2022

One of the biggest challenges during the implementation of any AI application is collecting and creating a good dataset to use in training. Our focus was improving the implementation of the CertificateOk application by switching from Faster R-CNN and SSD to YOLOv4, and as discussed in section 2 (State-of-the-art analysis) YOLOv4 showed much better results with smaller objects.

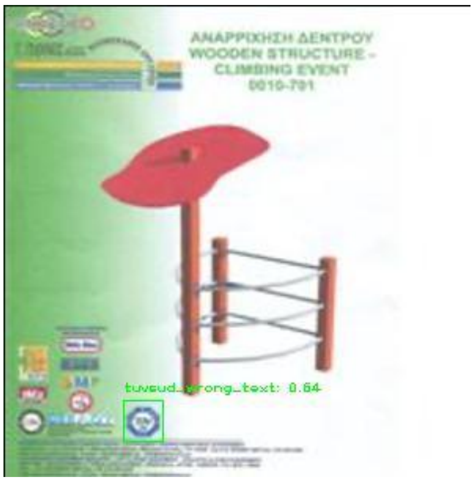
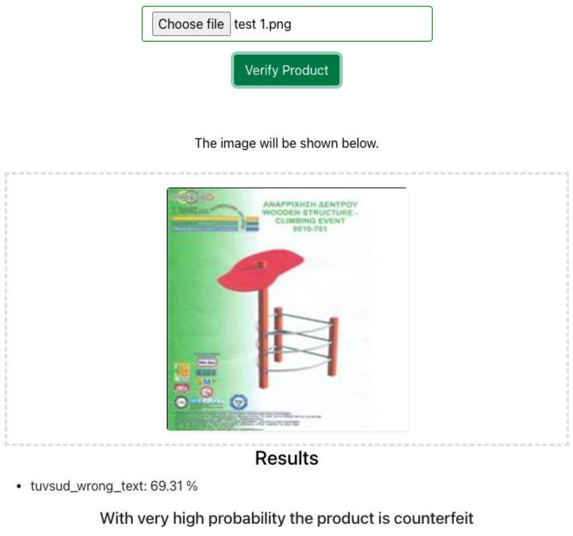


Figure 10. Detection example using old implementation

The image in Figure 10 shows a product image with TUV SUD fake logo with a low resolution and small logo size; the old implementation was SSD and Faster R-CNN could not detect the logo/marks of that size on multiple images with the same sizes. To compare the performance between the old implementation and the new one, both were deployed on the same Ubuntu server,

Figure 11 demonstrates the query with the same file and under the same conditions (Hardware environment and internet connection) completely different response experience.



The image will be shown below.

Results

- tuvsud_wrong_text: 69.31 %

With very high probability the product is counterfeit

Figure 11. Detection example using the new implementation

The old implementation receives the response after 15 seconds, while the new implementation 6 receives the final response of 600~900 MS, which is a huge improvement over the old implementation.

[6] http://aiapp.certificateok.de:8008/ accessed at 9th of January 2022

6. CONCLUSION AND OUTLOOK

In conclusion, this paper highlighted the difference between one-stage detectors with a focus on SSD Liu, et al. (2016) and YOLO with its different versions and two-stage detectors with a focus on Faster R-CNN Ren, et al. (2016). The highlight of that research is that YOLOv4 Alexey & Chien-Yao (2020) perform better on **lower resolution and small objects** than SSD or Faster R-CNN specifically because of the following the way FPN make a prediction which was first adopted in YOLOv3 Redmon & Farhadi (2019) and later in YOLOv4 PANet was used which make YOLOv4 even better in detecting smaller objects Alexey & Chien-Yao (2020).

In future work, the focus could be on collecting more data samples and working closely with big certificate issuers to cover a variety of options. There is also the possibility to improve the existing application by giving the certificate issuer the ability to upload their logos/marks, and it gets added to the model training process, that way in the future certificate issuer can change or update their logos and have that directly reflected on CertificateOk system.

In the end, the possibilities of using AI and machine learning in preventing counterfeit products are huge and there is always room for improvement, and empowering end-users that will make the marked a safer and a more transparent place for customers.

REFERENCES

Alexey, B. & Chien-Yao, W. a. H.-Y. M. L., 2020. YOLOv4: Optimal Speed and Accuracy of Object Detection, s.l.: s.n.

Chien-Yao, W. et al., 2020. CSPNet: A new backbone that can enhance learning capability of cnn, s.l.: IEEE.

Eduard, D., Martin, G., Dang, V. & Hung, N., 2020. IMPROVING FAKE PRODUCT DETECTION USING AI- BASED TECHNOLOGY, s.l.: 18th International Conference e-Society.

Girshick, R., 2015. Fast R-CNN, s.l.: s.n.

Girshick, R., Donahue, J., Trevor, D. & Jitendra, M., 2013. Rich feature hierarchies for accurate object detection and semantic segmentation, s.l.: s.n.

Jan, H., Rodrigo, B. & Bernt, S., 2017. Learning non-maximum suppression, Saarbrücken, Germany: Max Planck Institut für Informatik.

Joseph, R. & Ali, F., 2016. YOLO9000: Better, Faster, Stronger, s.l.: s.n.

Joseph, R., Santosh, D., Ross, G. & Ali, F., 2016. You Only Look Once: Unified, Real-Time Object Detection, s.l.: University of Washington.

Kaiming, H., Xiangyu, Z., Shaoqing, R. & Jian, S., 2015. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition, s.l.: s.n.

Liu, W. et al., 2016. SSD: Single Shot MultiBox Detector. s.l., arXiv preprint, arXiv:1512.02325.Manuel, C.-G., Jesús, T.-M. & Pedro, L.-B. a. J. G.-G., 2020. On the Performance of One-Stage and Two-Stage Object.

Redmon, J. & Farhadi, A., 2019. Yolov3: An incremental improvement. s.l., arXiv preprint arXiv:1804.02767.

Ren, S., He, K., Girshick, R. & Sun, J., 2016. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. s.l., In Advances in neural information processing systems (pp. 91-99)..

Shilpa, S., Mamta, K. & Trilok, K., 2021. A Real-Time Integrated Face Mask Detector to Curtail Spread of Coronavirus, s.l.: s.n.

Shu, L. et al., 2018. Path aggregation network for instance segmentation, s.l.: IEEE.

Tsung-Yi, L. et al., 2017. Feature Pyramid Networks for Object Detection. s.l., IEEE Conference on Computer Vision and Pattern Recognition (CVPR).