Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID-19. The COVID-19 resource centre is hosted on Elsevier Connect, the company's public news and information website.

Elsevier hereby grants permission to make all its COVID-19-related research that is available on the COVID-19 resource centre - including this research content - immediately available in PubMed Central and other publicly funded repositories, such as the WHO COVID database with rights for unrestricted research re-use and analyses in any form or by any means with acknowledgement of the original source. These permissions are granted for free by Elsevier for as long as the COVID-19 resource centre remains active.
ETL-YOLO v4: A face mask detection algorithm in era of COVID-19 pandemic

Akhil Kumar a, *, Arvind Kalia a, Aayushi Kalia b

a Department of Computer Science, Himachal Pradesh University, Shimla, India
b CSED, Thapar Institute of Engineering & Technology, Patiala, India

KEYWORDS: COVID-19; Face mask detection; Tiny YOLO v4; SPP network; Mish activation

ABSTRACT

During the last two years, several deep learning-based methods for face mask detection have been proposed by researchers. However, most of the proposed methods struggle with the detection of face masks that are too small an object to detect and further achieve low detection accuracy. Considering the issues of the existing methods, in this work, we have proposed ETL-YOLO v4 with a modified and improved feature extraction and prediction network for tiny YOLO v4 which surpasses all its predecessors and other related work in the literature. To develop ETL-YOLO v4, we have improved the backbone architecture of tiny YOLO v4 by adding a modified-dense SPP network, two additional detection layers with modified and optimized CNN layers that aid in accurate prediction, used Mish as the activation function, and utilized modified anchor boxes. Furthermore, to obtain detection results in images of varied viewpoints, we have added Mosaic and CutMix data augmentation at training time. The proposed ETL-YOLO v4 achieved 9.93% higher mAP, 5.75% higher average precision (AP) for faces with masks, and 16.6% higher average precision (AP) for the face mask region as compared to its original base-line variant.

1. Introduction

At present, with every passing day, a new variant of the COVID-19 virus is appearing and healthcare practitioners are advising to adopt a COVID appropriate behavior where wearing a face mask in outdoor spaces is one of the health guidelines [1]. Governments across the world are pushing their citizens to wear face masks at every possible space to prevent the community spread of new variants of Coronavirus [2]. In a few countries, the government is monitoring its citizens by social policing to check if the people are following COVID appropriate behavior of wearing face masks in public spaces [3]. Considering this ongoing problem and to come up with a solution with artificial intelligence and deep learning there exists a dire need of computer vision algorithms that are capable enough of detecting persons wearing face masks and not wearing face masks. Researchers in the past two years have proposed several face mask classification and detection methods that are capable of detecting faces with masks and without masks. However, the methods in the literature largely miss to improve the existing state-of-the-art and come up with a detector that is capable of performing detection of face masks in distant and occluded images. Moreover, the other problem is achieving low detection accuracy by the proposed face mask detectors.

In recent years, the YOLO algorithm series [4–7] has shown fascinating results in multiple sub-areas of object detection. Several researchers have utilized the full-scale and tiny variants of the YOLO algorithm to solve detection problems such as general object
Algorithms. The tiny series of YOLO utilize lesser computation resources and are capable of integrating with mobile devices and embedded systems. Considering this advantage of tiny YOLO variants, few researchers have proposed face mask detection systems based on tiny YOLO v3 and v4 [14,15]. However, these systems struggle to perform detection in occluded and distant images, and in scenarios where a face mask is too small an object to detect.

In order to address the issues in the existing work and to develop an effective face mask detection algorithm, in this work we have proposed ETL-YOLO v4 which is based on addition of extra convolutional layers and detection layers in the existing tiny YOLO v4 algorithm. The ETL-YOLO v4 is fine-tuned and refined in such a manner that it can achieve high detection accuracy and precision in images of different complexities and viewpoints which very much solve the problems of the existing methods. The proposed ETL-YOLO v4 algorithm for face mask detection is a lightweight detector with only twenty-nine convolutional layers and can train with 2.5 GB/RAM and 1.7 GB/GPU. The advantage of the ETL-YOLO v4 is that it can be deployed to Jetson Nano [16] to perform real-time detections by connecting to a ZED or CCTV camera. The proposed ETL-YOLO v4 is a useful tool in this COVID-19 pandemic times where it can be utilized by the governments and other healthcare agencies to monitor whether the people are following the norms of wearing a face mask or not in social places, hospitals, schools, workplaces and universities to stop the community spread of the disease.

This work primarily contributes:

1. Proposed ETL-YOLO v4 algorithm which is an upscale version of tiny YOLO v4 for detection of face masks in COVID-19 pandemic era. The proposed algorithm overcome the shortcomings of tiny YOLO v4 in terms of detection accuracy.
2. The proposed ETL-YOLO v4 algorithm is improved by adding a dense SPP network and two additional YOLO detection layers to achieve a dense and rich feature map that enables the algorithm to detect small and distant face masks with high accuracy and precision.
3. Training the algorithm with Mish activation function in place of Leaky ReLU employed by original tiny YOLO v4 to keep the training loss unbound.
4. Added Mosaic and CutMix augmentation at training time of the algorithm so that varied viewpoints and complexities are generated in the images at training time to enable the algorithm to learn with more volume of data consisting of varying orientations.
5. Comparison has been drawn with original tiny YOLO v4, YOLO v3, EfficientNet YOLO-v4. In comparison, the proposed ETL-YOLO v4 outperforms the original tiny YOLO v4 by achieving 9.93% higher mAP, 5.75% higher average precision (AP) for faces with masks, and 16.6% higher average precision (AP) for the presence of a mask on the face region. Furthermore, the proposed ETL-YOLO v4 surpassed the performance of full-scale YOLO v3, EfficientNet-YOLO v4, YOLO v1, and YOLO v2, and other related work in the literature.

This work is organized into following sections. Section 2 presents the related work in the face mask detection domain; Section 3 presents the architecture of original tiny YOLO v4 and improvements embodied to develop the proposed algorithm; Section 4 focuses on experiment set-up and results analysis and; Section 5 presents the conclusion and future scope of the work.

2. Related work

Recently a few studies in the field of face mask classification and detection have been proposed by the researchers using deep learning-based methods. The hotspot in all of the researches focuses upon the detection of faces with masks and without masks. However, the researches in the literature miss the important aspect of detection of the face mask region which might open a new dimension in this research area where identities behind the face mask can be determined using face detectors.

Loey et al. [17] based on the combination of machine learning and deep learning proposed a hybrid method for face mask classification. The authors combined the feature extraction network of ResNet-50 and used the strategy of transfer learning to pass on the features to the SVM classifier to check the presence or absence of face masks in images. The authors tested the proposed method on RMFD dataset and achieved a classification accuracy of 99.6% with the hybrid model. As a step towards upsampling the task of face mask classification to face mask detection, Loey et al. [18] combined the ResNet-50 feature extraction network with the YOLO v2 detection network. Authors tested their hybrid technique on a custom dataset consisting of persons wearing medical masks. The proposed hybrid ResNet-50 and YOLO v2 technique achieved an average precision of 81% for images containing faces with masks. Chowdary et al. [19] utilized the transfer learning strategy of InceptionNet v3 and proposed a CNN method for face mask detection. The proposed technique attained an accuracy of 99.6% while training on the SMFD dataset. Roy et al. [20] proposed a face mask detection work by creating a face mask dataset with annotations for faces with masks and faces without masks. The proposed dataset consists of around 3000 images for two class labels. To gauge the validity of the proposed dataset, the authors employed tiny YOLO v3, YOLO v3, SSD, and Faster R-CNN object detectors. On training and testing on the MOAX dataset, YOLO v3 attained a mAP value of 63.99% outperforming other tested detectors by a margin of 3–13%. Nagrath et al. [21] combined MobileNet v2 feature extractor with SSD object detector to propose a face mask detector. The authors tested the proposed technique on a custom dataset of samples for persons wearing masks and not wearing face masks. The proposed combination of MobileNetv2-SSD achieved a classification accuracy of 92.94% on the employed dataset.

Khandelwal et al. [22] employed the MobileNetv2 object classifier to propose a technique for face mask classification. The authors
utilized the capability of the MobileNetv2 classifier to binarize the passed images into two categories i.e. faces with masks and faces without masks. The authors tested the proposed technique on a small dataset consisting of 380 samples for faces with masks and 460 samples for faces without masks. On the employed dataset, the MobileNetv2 based technique achieved an accuracy of 97.6%. Li et al. [23] utilized deep learning-based convolutional neural networks to determine the head pose of a person under consideration and further classify the mask-wearing status. The proposed technique is capable of determining the front view and side view based on the head pose view. Authors using deep learning utilized HGL features to determine the head pose of a person. The proposed technique produced fascinating results by attaining accuracy of 93.64% for the front view and 87.17% for the side view. In all the scenarios face mask was present on the face area therefore, along with the head pose, front, and side view, the proposed technique can detect the presence of a face mask in different viewpoints of the face. Inamdar and Mehendale [24] utilized deep learning-based convolutional neural networks to propose a system for face mask detection. The authors utilized a small dataset consisting of 35 samples where 10 samples were with masks, 15 samples were without masks, and 10 samples were having masks worn incorrectly. On the self-created dataset, the authors achieved a classification accuracy of 98.6% with the proposed system. Kumar et al. [25] proposed a novel face mask detection dataset and further tested and proposed four new variants of the tiny YOLO algorithm. On the novel dataset, the proposed tiny YOLO v3 achieved a mAP value of 53.15%, and the proposed tiny YOLO v4 achieved a mAP value of 60.25%.

As indicated in the literature, primarily most of the researches address only the classification and detection of faces with masks and without masks. The related studies largely miss to pave efforts for detection of face mask region in images for faces with masks and further proposes a state-of-the-art technique capable of detecting face mask region which is noticeably a small object to localize and detect, and a challenging task in itself.

### 3. Method

In this section, the original and proposed ETL-YOLO v4 algorithm are described. In order to achieve better detection results and to come up with a face mask detection algorithm better than the full-scale variants of YOLO and other work in the literature, we have paved efforts to improve the backbone architecture of tiny YOLO v4, modified the activation function, and added data augmentation at

| Type          | Filters | Size/Stride | Output         |
|---------------|---------|-------------|----------------|
| Convolutional | 32      | 3 × 3/2     | 208 × 208 × 32 |
| Convolutional | 64      | 3 × 3/2     | 104 × 104 × 64 |
| Convolutional | 64      | 3 × 3       | 104 × 104 × 64 |
| Route         |         |             | 104 × 104 × 32 |
| Convolutional | 32      | 3 × 3       | 104 × 104 × 32 |
| Convolutional | 32      | 3 × 3       | 104 × 104 × 32 |
| Route         |         |             | 104 × 104 × 64 |
| Convolutional | 64      | 1 × 1       | 104 × 104 × 64 |
| Maxpool       | 2 × 2/2 | 52 × 52 × 128 |
| Convolutional | 128     | 3 × 3       | 52 × 52 × 128  |
| Route         |         |             | 52 × 52 × 64   |
| Convolutional | 64      | 3 × 3       | 52 × 52 × 64   |
| Convolutional | 64      | 3 × 3       | 52 × 52 × 64   |
| Route         |         |             | 52 × 52 × 128  |
| Convolutional | 128     | 1 × 1       | 52 × 52 × 128  |
| Maxpool       | 2 × 2/2 | 52 × 256    |
| Convolutional | 256     | 3 × 3       | 26 × 26 × 256  |
| Route         |         |             | 26 × 26 × 128  |
| Convolutional | 128     | 3 × 3       | 26 × 26 × 128  |
| Convolutional | 128     | 3 × 3       | 26 × 26 × 128  |
| Route         |         |             | 26 × 26 × 256  |
| Convolutional | 256     | 1 × 1       | 26 × 26 × 256  |
| Maxpool       | 2 × 2/2 | 26 × 26 × 512 |
| Convolutional | 512     | 3 × 3       | 13 × 13 × 512  |
| Convolutional | 256     | 1 × 1       | 13 × 13 × 256  |
| Convolutional | 512     | 3 × 3       | 13 × 13 × 512  |
| Convolutional | 27      | 1 × 1       | 13 × 13 × 27   |
| YOLO          |         |             | 13 × 13 × 256  |
| Route         |         |             | 13 × 13 × 128  |
| Convolutional | 128     | 1 × 1       | 13 × 13 × 256  |
| Upsample      | 2       |             | 26 × 26 × 128  |
| Route         |         |             | 26 × 26 × 384  |
| Convolutional | 256     | 3 × 3       | 26 × 26 × 256  |
| Convolutional | 27      | 1 × 1       | 26 × 26 × 27   |

**Table 1**

Network architecture of tiny YOLO v4 algorithm.
training time. The details of the original tiny YOLO v4 and proposed ETL-YOLO v4 algorithm are present in subsequent sub-sections.

3.1. Tiny YOLO v4 algorithm

The tiny YOLO v4 algorithm [7] is a lightweight and mini version of YOLO v4 and it is constituted of twenty-one convolutional layers with two YOLO detection layers. The tiny YOLO v4 is 1/10th the size of the YOLO v4 algorithm in terms of CNN layers and trainable parameters. The advantage of tiny YOLO v4 is its ability to fast training and produce highly accurate detection results as compared to tiny variants of YOLO v1, v2, and v3. Furthermore, this advantage of tiny YOLO v4 makes it capable of integrating with mobile devices and other low computation resources-based systems. However, the tiny YOLO v4 algorithm struggles to detect small objects.

Contrary to other variants of YOLO, the tiny YOLO v4 utilizes the CSPDarkNet-29 feature extraction network and for detection, it uses two YOLO detection layers which produce a feature map of $13 \times 13 \times 27$ and $26 \times 26 \times 27$. For evaluating loss at training time, it employs the C-IoU loss function and uses GreedyNMS for Non-Max Suppression. The entire training of the algorithm is performed with the Leaky ReLU activation function and before YOLO detection layers, it uses linear activation function. The detailed network architecture of tiny YOLO v4 is shown in Table 1.

3.2. Proposed ETL-YOLO v4 algorithm

The tiny YOLO v4 algorithm in recent times has shown fascinating results in general object category detection on the MS COCO dataset [26]. Few researchers by proposing new variants of tiny YOLO v4 for face mask detection have scaled up the performance in terms of speed and detection [27]. However, the issue of inaccurate detection in distant images and the ability to detect small objects still surface in the published work. In order to address the issues of the related work and to scale up the performance of tiny YOLO v4 few necessary changes are required in the feature extraction and detection network of the algorithm therefore, we propose a new variant of the tiny YOLO v4 with the name ETL-YOLO v4 for face mask detection which is capable of producing high precision and detection accuracy.

The necessary changes embodied in the feature extraction and detection of the tiny YOLO v4 are first, we improved the feature extraction network by adding a dense SPP network and secondly, by adding two additional YOLO detection layers with wisely chosen convolutional layers that enables the algorithm to detect small objects such as a face mask on the face area with very high precision. Furthermore, to prevent the training loss, we have added Mish activation function in place of Leaky ReLU activation in the feature extraction network. To get detection results in varied complexities, we have incorporated the Mosaic and CutMix data augmentation of the YOLO v4 algorithm at the training time that tiny YOLO v4 lack. The added improvements enhanced the performance of tiny YOLO v4 by leaps and bounces and enabled it to perform better than full-scale YOLO v3 and original tiny YOLO v4.

3.2.1. Improved feature extraction and detection network

The full-scale YOLO v3 and YOLO v4 utilize three YOLO detection layers with anchor values of 9 which allows these to achieve benchmark results. Furthermore, YOLO v4 has an SPP network for extraction of a rich feature map, uses Mish as the activation function, and allows augmentations namely, Mosaic, CutMix, and LetterBox at training time which allows it to outperform other object detectors and make it a state-of-the-art. Considering these additional components of full-scale YOLO v3 and YOLO v4, we have embodied a few changes in the CNN architecture of the tiny YOLO v4 network which it lacks. To improve the backbone architecture of the tiny YOLO v4 algorithm, we have added a dense SPP network [28] constituted of four maxpool layers of size $3 \times 3$, $5 \times 5$, $7 \times 7$, and $9 \times 9$. The embodiment of this SPP network converts the feature map of size $13 \times 13 \times 512$ produced by the 15th convolutional layer into a feature map of $13 \times 13 \times 2560$ thus, allowing a larger number of features passed on to the next layers of the network and extract useful features from the large pool of related features. Furthermore, we have fine-tuned the second YOLO detection layer by providing a suitable filter size for each convolutional layer. Instead of using small-sized convolutional layers as used in tiny YOLO v4,
we have used layers of size 256 and 512 thus, producing a larger feature map. In order to improve the detection ability of the algorithm and make it capable of detecting the face mask region which is noticeably a small object to detect, we have added two additional YOLO detection layers with each having a wisely chosen filter size for the convolutional layers to extract and detect objects under consideration at distance and small in size. For the third YOLO detection layer, we have added convolutional layers of sizes 512, 128, 256, 512, and 27. Whereas, for the fourth detection layer, the convolutional layers are of sizes 27, 64, 128, 256, and 27. With these improvements, the first YOLO layer produces a feature map of size $13 \times 13 \times 27$; the second YOLO detection layer produces a feature map

| Table 2 | Network architecture of proposed ETL-YOLO v4. |
|---------|---------------------------------------------|
| Type    | Filters | Size/stride | Output                |
| Convolutional | 32      | $3 \times 3/2$ | $208 \times 208 \times 32$ |
| Convolutional | 64      | $3 \times 3/2$ | $104 \times 104 \times 64$ |
| Convolutional | 64      | $3 \times 3$   | $104 \times 104 \times 64$ |
| Route    |          |              | $104 \times 104 \times 32$ |
| Convolutional | 32      | $3 \times 3$   | $104 \times 104 \times 32$ |
| Convolutional | 32      | $3 \times 3$   | $104 \times 104 \times 32$ |
| Route    |          |              | $104 \times 104 \times 64$ |
| Convolutional | 64      | $1 \times 1$   | $104 \times 104 \times 64$ |
| Route    |          |              | $104 \times 104 \times 128$ |
| Maxpool  |          | $2 \times 2/2$ | $52 \times 52 \times 128$ |
| Convolutional | 128     | $3 \times 3$   | $52 \times 52 \times 128$ |
| Route    |          |              | $52 \times 52 \times 64$  |
| Convolutional | 64      | $3 \times 3$   | $52 \times 52 \times 64$  |
| Convolutional | 64      | $3 \times 3$   | $52 \times 52 \times 64$  |
| Route    |          |              | $52 \times 52 \times 128$ |
| Convolutional | 128     | $1 \times 1$   | $52 \times 52 \times 128$ |
| Route    |          |              | $52 \times 52 \times 256$ |
| Maxpool  |          | $2 \times 2/2$ | $26 \times 26 \times 256$ |
| Convolutional | 256     | $3 \times 3$   | $26 \times 26 \times 256$ |
| Route    |          |              | $26 \times 26 \times 128$ |
| Convolutional | 128     | $3 \times 3$   | $26 \times 26 \times 128$ |
| Convolutional | 128     | $3 \times 3$   | $26 \times 26 \times 128$ |
| Route    |          |              | $26 \times 26 \times 256$ |
| Convolutional | 256     | $1 \times 1$   | $26 \times 26 \times 256$ |
| Route    |          |              | $26 \times 26 \times 512$ |
| Maxpool  |          | $2 \times 2/2$ | $13 \times 13 \times 512$ |
| Convolutional | 512     | $3 \times 3$   | $13 \times 13 \times 512$ |
| Maxpool  |          | $3 \times 3$   | $13 \times 13 \times 512$ |
| Route    |          |              | $13 \times 13 \times 512$ |
| Maxpool  |          | $5 \times 5$   | $13 \times 13 \times 512$ |
| Route    |          |              | $13 \times 13 \times 512$ |
| Maxpool  |          | $7 \times 7$   | $13 \times 13 \times 512$ |
| Route    |          |              | $13 \times 13 \times 512$ |
| Maxpool  |          | $9 \times 9$   | $13 \times 13 \times 512$ |
| Route    |          |              | $13 \times 13 \times 512$ |
| Convolutional | 256     | $1 \times 1$   | $13 \times 13 \times 2560$ |
| Convolutional | 512     | $3 \times 3$   | $13 \times 13 \times 256$ |
| Convolutional | 27      | $1 \times 1$   | $13 \times 13 \times 256$ |
| YOLO     |          |              | $13 \times 13 \times 256$ |
| Route    |          |              | $13 \times 13 \times 256$ |
| Convolutional | 256     | $1 \times 1$   | $13 \times 13 \times 256$ |
| Upsample |          |              | $13 \times 13 \times 256$ |
| Route    |          |              | $13 \times 13 \times 256$ |
| Convolutional | 512     | $3 \times 3$   | $26 \times 26 \times 512$ |
| Convolutional | 27      | $1 \times 1$   | $26 \times 26 \times 512$ |
| YOLO     |          |              | $26 \times 26 \times 512$ |
| Route    |          |              | $26 \times 26 \times 512$ |
| Convolutional | 128     | $1 \times 1$   | $26 \times 26 \times 128$ |
| Upsample |          |              | $26 \times 26 \times 128$ |
| Route    |          |              | $26 \times 26 \times 128$ |
| Convolutional | 512     | $3 \times 3$   | $26 \times 26 \times 512$ |
| Convolutional | 27      | $1 \times 1$   | $26 \times 26 \times 512$ |
| YOLO     |          |              | $26 \times 26 \times 512$ |
| Route    |          |              | $26 \times 26 \times 512$ |
| Convolutional | 64      | $1 \times 1$   | $52 \times 52 \times 27$ |
| Upsample |          |              | $52 \times 52 \times 27$ |
| Route    |          |              | $52 \times 52 \times 27$ |
| Convolutional | 256     | $3 \times 3$   | $104 \times 104 \times 256$ |
| Convolutional | 27      | $1 \times 1$   | $104 \times 104 \times 256$ |
| YOLO     |          |              | $104 \times 104 \times 256$ |
of size $26 \times 26 \times 27$; and the third and fourth YOLO detection layers produces a feature map of size $52 \times 52 \times 27$ and $104 \times 104 \times 27$ thus, enabling the algorithm to detect objects under consideration i.e. faces with masks and specifically, presence of a face mask on the face area with high accuracy and precision. The original tiny YOLO v4 can only produce a feature map of size $13 \times 13 \times 27$ and $26 \times 26 \times 27$ thus, struggles to detect small objects and objects at distance. Each YOLO detection layer in the network is used for processing and predicting bounding boxes, objectness score, anchors, and class predictions. Since we have used four detection layers and there are four classes in the dataset, the filter size is set to 27 before each YOLO layer by computing using formula $(\text{classes} + 5) \times 3$. The detailed network configuration of the proposed ETL-YOLO v4 algorithm is presented in Table 2. Furthermore, the detailed description of added dense SPP network is illustrated in Fig. 2.

In the proposed ETL-YOLO v4, we have used Greedy NMS as Non-Max Suppression and C-IoU loss function [29] which allows the convolutional neural network to achieve faster convergence and regression. Furthermore, the proposed face mask detection task is a bounding box regression problem therefore, C-IoU loss fits best in for the case. The formula for C-IoU loss is given in Eq. (1).

$$L_{CIOU} = 1 - IoU + \frac{\rho^2(B, B^{\text{gt}})}{c^2} + \alpha \nu$$  \hspace{1cm} (1)

In the above equation, $CIOU$ is the complete-IoU loss; $IoU$ is the intersection over union; $B$ represents the predicted box; $B^{\text{gt}}$ represents the ground truth box; $\rho$ is the Euclidean distance between $B$ and $B^{\text{gt}}$; $c$ is the diagonal length between the boxes; $\alpha$ is the positive trade-off parameter; $\nu$ is the measure of the consistency of aspect ratio.

3.2.2. Modified activation function

In the feature extraction backbone and detection layers, tiny YOLO v4 employs the Leaky ReLU activation function. However, for faster classification and detection, rich extraction of the feature maps and to aid regularization and avoid overfitting authors of YOLO v4 have used the Mish activation function. Following the strategy of YOLO v4 and to achieve better performance indicators, in the proposed ETL-YOLO v4, we have incorporated the Mish activation function in place of Leaky ReLU activation in the feature extraction and detection network.

Mish activation function is a self-regularizing non-monotonic activation function and have outperformed activation functions namely, Swish, GELU, ReLU, ELU, Leaky ReLU, SELU, SoftPlus, SReLU, ISRU, and RReLU in object classification tasks [30]. Mathematically, Mish activation function can be represented by Eq. (2).

$$f(x) = x\tanh(softplus(x))$$ \hspace{1cm} (2)

The advantage of using Mish activation over Leaky ReLU is its ability to be unbounded above i.e. it avoids saturation due to capping and can achieve any positive value which aids better gradient flow while training the CNN model. However, the Leaky ReLU is bounded and has an order of continuity as zero which may cause problems in gradient-based optimization.
3.2.3. Augmentation at training time

The YOLO v4 object detector has achieved very high detection accuracy on the MS COCO dataset and is a current state-of-the-art. The reason behind the outmatching performance of the YOLO v4 algorithm is not its feature extraction and detection network but the way it prepares and utilizes the data at training time. The YOLO v4 at training time utilizes data augmentation techniques namely, Mosaic, CutMix, and LetterBox. However, the same misses in the training network of the tiny YOLO v4 algorithm. Considering this advantage of YOLO v4, we have embodied Mosaic [7] and CutMix [31] data augmentation at training time in the training network of the proposed ETL-YOLO v4 algorithm. Adding these data augmentation techniques aid in obtaining different viewpoints and amalgamation of multiple images and dataset object categories in a single image.

In the training architecture of the proposed ETL-YOLO v4, firstly we have embodied the Mosaic augmentation which allowed us to combine four training images into one in a certain ratio. Adding Mosaic augmentation allows the training model to learn how to identify different dataset object categories at a smaller scale than normal. Furthermore, it reduces the need for a large mini-batch size at training time. The other augmentation technique utilized at the training time is CutMix data augmentation. The CutMix augmentation allows to cut and paste random patches between training images. It allows mixing the ground truth labels in proportion to the area of patches in the images. CutMix augmentation aids in increasing localization ability by making the model more intuitive and focusing on less discriminative parts of the object being classified. The sample images for Mosaic and CutMix augmentation utilized by the proposed ETL-YOLO v4 face mask detection algorithm at training time are illustrated in Fig. 3.

Fig. 3(a) shows a combined image of four different images of the dataset whereas, Fig. 3(b) shows a patch pasted in lower left and upper left in the image.

4. Experiments and analysis

To carry out this work, the proposed ETL-YOLO v4 and other tested algorithms are implemented using TensorFlow and Keras deep learning libraries on an Intel i5–8th Gen CPU with 12 GB/RAM and 4 GB/GPU. At training time, the hyperparameters for ETL-YOLO v4 and other tested variants are set as input size to 416 × 416, batch size to 32, sub-divisions to 16, momentum to 0.9, decay to 0.5, and learning rate to 0.0261. The algorithms employed are trained for 8000 iterations by keeping the confidence threshold for each class of the dataset to 0.25 and the IoU threshold to 50%. While training, the proposed ETL-YOLO v4 algorithm utilized 2.5 GB/RAM and 1.7 GB/GPU memory. For training the proposed algorithm for 8000 iterations, the utilized computation platform took around 5 h for training.

4.1. Dataset

In order to train and test the proposed ETL-YOLO v4 algorithm and other variants of YOLO employed in this work, we have used the face mask detection dataset (FMD) [25]. The employed dataset is constituted of 52,635 images with approximately 50,000 bounding boxes for class labels, with mask, without mask, mask incorrect, and mask area. Furthermore, for training, test, and validation, the dataset is split in the ratio of 80:10:10. The employed dataset has augmented images with viewpoints such as, flip, shear, zoom, HSV, and rotate with distinct annotation for each viewpoint. The advantage of using the face mask detection dataset over other datasets is its
number of images, augmentation and annotation, and coverage of face mask region which other datasets lack. The sample images of the dataset are illustrated in Fig. 4.

4.2. K-means++ based anchor boxes

The idea of anchor boxes was first introduced in Faster R-CNN [32]. Anchors boxes are one of the most important criteria that help in tuning the CNN-based object detectors to perform detection on a dataset. Anchor boxes help in determining the small, large, and irregular objects in an image and further aid in detection. The YOLO v4 and tiny YOLO v4 compute anchor boxes using the K-means clustering method. The K-means clustering is based on the selection of the centroids around which the clustering takes place. If the initial selection of the centroid is not correct, the computed clustering values result in incorrect clusters which leads to the collection of non-uniform data points coming together. To overcome this drawback and to collect the data points with the same properties, in this work, we have utilized one step ahead method based on K-means++ clustering. In the K-means++ clustering method, firstly the centroid is picked and then a cluster of data with similar properties is gathered. Unlike K-means, the K-means++ clustering is independent of the initialization of centroid. Furthermore, utilizing K-means++ clustering support in the selection of the optimal number of anchor boxes and aid in preventing convergence of CNN network at training time which is an important parameter for local optimization for a CNN-based object detector.

In the K-means++ clustering method as proposed in [33,34], an anchor box is randomly selected as the centroid of the current first cluster and further, the shortest distance between the selected cluster center and other anchor boxes is calculated. In this process, the anchor boxes based on the small distance are classified into the category it belongs to. As a next step, the probability of other anchor boxes is identified so that the next cluster center can be determined, and the anchor box with the highest probability is considered as the next center. The K-means++ clustering is computed based on the probability formula given in Eq. (3).

\[
P = \frac{D(x)}{\sum_{i=1}^{n} D(x_i)}
\]  

(3)

In the above equation, \(D(x)\) represents the shortest distance measured by evaluation Intersection over Union (IoU) between each anchor box and current centroid. This process is repeated until each object is reassigned to other clusters and then K clusters are extracted out for the dataset.

Based on the strategy and advantage of K-means++ clustering, in this work, we have used a K size of 12 and determined the anchor boxes for the employed dataset. The K size of 12 is chosen considering the number of YOLO detection layers utilized in the proposed algorithm. In the proposed ETL-YOLO v4, we have utilized 4 YOLO detection layers, therefore, each utilizing 12 values for anchor boxes. The values of anchor boxes used in present work are: \([(8 \times 12), (15 \times 22), (21 \times 36)], (36 \times 40), (28 \times 59), (44 \times 74), (54 \times 115), (83 \times 86), (84 \times 163), (145 \times 131), (138 \times 245), (238 \times 304)]\).

4.3. Evaluation parameters

To test the effectiveness of the proposed ETL-YOLO v4 algorithm, we have used the following performance indicators: precision (P), recall (R), F1 score, average precision (AP), and mean average precision (mAP). The chosen performance indicators have been widely used in optical recognition tasks to evaluate detection accuracy. The formulae for the stated performance indicators are presented in Eqs. (4–8).

\[
\text{precision} \quad (P) = \frac{TP}{TP + FP} \quad \text{(4)}
\]

\[
\text{recall} \quad (R) = \frac{TP}{TP + FN} \quad \text{(5)}
\]

\[
\text{F1 score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad \text{(6)}
\]

Fig. 4. Dataset images with persons wearing and not wearing face masks.
Performance comparison of tiny YOLO v4 and ETL-YOLO v4.

In Table 3, the performance of the proposed ETL-YOLO v4 algorithm is compared with tiny YOLO v4 and other variants of the YOLO algorithm. The performance comparison is based on average precision (AP) and mean average precision (mAP) for each class of the face mask detection dataset. The proposed ETL-YOLO v4 algorithm achieved an AP value of 86.97% over 70.37% attained by the original tiny YOLO v4. The AP value achieved by the proposed algorithm is 16.6% higher as compared to its original counterpart which signifies it to be more capable of detecting small objects like masks on the face region with much high precision. Moreover, the proposed algorithm attained an intersection over union (IoU) value of 61.02% on the employed dataset which signifies that the proposed ETL-YOLO v4 is capable of detecting overlapping objects with much high accuracy. The performance comparison of tiny YOLO v4 and ETL-YOLO v4 are presented in Table 3. The detection behavior of the proposed ETL-YOLO v4 on real-time images is shown in Fig. 5. Fig. 5(a) shows the detection results for faces with masks, Fig. 5(b) presents the results for faces without masks, Fig. 5(c) illustrates the detection results for persons wearing face masks in a public area, Fig. 5(d) presents the results for detection of face masks in times of COVID-19 to determine whether the healthcare workers and patients are wearing a face mask or not, Fig. 5(e) presents the detection results on a Mosaic augmented image where four images are patched together in a single image, and Fig. 5(f) illustrates the detection on a CutMix augmented images where a patch of an image is embedded in an image. From the detection results, it is quite evident that in almost every real-time scenario, the proposed ETL-YOLO v4 algorithm localized and detected the faces with masks and face masks with much high accuracy and precision. Moreover, it is capable of detecting distant and small face masks as illustrated in Fig. 5(c), (e), and (f). The fascinating results obtained by the proposed ETL-YOLO v4 algorithm make it capable of an important tool in present times of the COVID-19 pandemic where it can be embedded to surveillance cameras at public places and hospitals to identify if the people are following the COVID appropriate behavior of wearing a face mask or not.

4.4. Evaluation results

To get insightful and intuitive results, we combined the above work and have evaluated the proposed ETL-YOLO v4 algorithm based on precision, recall, F1 score, mAP, and AP for each class of the face mask detection dataset and drew comparisons with the original tiny YOLO v4 algorithm. The comparative results show that the proposed ETL-YOLO v4 achieved a 9.93% higher value for mean average precision (mAP) which signifies high detection accuracy by the proposed algorithm. Furthermore, in terms of average precision (AP) for each class of the dataset, the proposed algorithm achieved a value of 86.69% for images with masks over 83.94% obtained by original tiny YOLO v4 signifying higher precision in detecting images containing faces with masks. For images without masks, the proposed algorithm achieved an AP value 11.5% higher compared to tiny YOLO v4 signifying better detection precision to identify faces without masks. Specifically, for mask area i.e. presence of a face mask on the face area which is in itself a challenge in object detection, the proposed ETL-YOLO v4 algorithm achieved an AP value of 86.97% over 70.37% attained by the original tiny YOLO v4. The AP value achieved by the proposed algorithm is 16.6% higher as compared to its original counterpart which signifies it to be more capable of detecting small objects like masks on the face region with much high precision. Moreover, the proposed algorithm attained an intersection over union (IoU) value of 61.02% on the employed dataset which signifies that the proposed ETL-YOLO v4 is capable of detecting overlapping objects with much high accuracy. The performance comparison of tiny YOLO v4 and proposed ETL-YOLO v4 are presented in Table 3. The detection behavior of the proposed ETL-YOLO v4 on real-time images is shown in Fig. 5. Fig. 5(a) shows the detection results for faces with masks, Fig. 5(b) presents the results for faces without masks, Fig. 5(c) illustrates the detection results for persons wearing face masks in a public area, Fig. 5(d) presents the results for detection of face masks in times of COVID-19 to determine whether the healthcare workers and patients are wearing a face mask or not, Fig. 5(e) presents the detection results on a Mosaic augmented image where four images are patched together in a single image, and Fig. 5(f) illustrates the detection on a CutMix augmented images where a patch of an image is embedded in an image. From the detection results, it is quite evident that in almost every real-time scenario, the proposed ETL-YOLO v4 algorithm localized and detected the faces with masks and face masks with much high accuracy and precision. Moreover, it is capable of detecting distant and small face masks as illustrated in Fig. 5(c), (e), and (f). The fascinating results obtained by the proposed ETL-YOLO v4 algorithm make it capable of an important tool in present times of the COVID-19 pandemic where it can be embedded to surveillance cameras at public places and hospitals to identify if the people are following the COVID appropriate behavior of wearing a face mask or not.

4.5. Comparison with other algorithms

To find out the validity and effectuality of the proposed ETL-YOLO v4 algorithm for the face mask detection task further, we compared it with tiny YOLO v4 and other full-scale variants of the YOLO algorithm. These variants also include a hybrid face mask detector based on the EfficientNet feature extractor and YOLO v4 [35]. All the tested variants are trained from the scratch on the face mask detection dataset which is used to train and test the proposed ETL-YOLO v4. The performance comparison of the proposed algorithm and other variants have been performed based on mean average precision (mAP) and average precision (AP) for each class of

Table 3

| Algorithm          | Class      | AP    | Precision | Recall | F-1 Score | mAP   |
|--------------------|------------|-------|-----------|--------|-----------|-------|
| tiny YOLO v4       | with mask  | 83.94%| 79%       | 65%    | 72%       | 57.71%|
|                    | without mask| 53.16%|           |        |           |       |
|                    | mask incorrectly | 23.99%|           |        |           |       |
|                    | mask area    | 70.37%|           |        |           |       |
| Proposed ETL-YOLO v4 | with mask  | 89.69%| 85%       | 72%    | 78%       | 67.64%|
|                    | without mask| 64.66%|           |        |           |       |
|                    | mask incorrectly | 29.25%|           |        |           |       |
|                    | mask area    | 86.97%|           |        |           |       |
Fig. 5. Detection results with proposed ETL-YOLO v4. (a) Face with mask and mask area. (b) Face without mask. (c) Face mask detection results for a public area. (d) Face mask detection results in COVID-19 pandemic. (e) Face mask detection results for Mosaic augmented images. (f) Face mask detection results for CutMix augmented images.
The comparisons of detection accuracy are shown in Fig. 6 and Table 4 considering an intersection over union (IoU) value of 50%. As illustrated in Fig. 6, the proposed ETL-YOLO v4 outperformed tiny YOLO v4 in terms of AP for each class of the dataset. Furthermore, it surpassed the performance of YOLO v3 and EfficientNet YOLO v4 as well by leaps and bounces by obtaining 2–26% higher value for AP for faces with masks, 3% higher AP value for faces without masks, 6% higher AP value for mask incorrectly as compared to YOLO v3, and 1–86% higher AP value for face mask region. In terms of mean average precision (mAP) as shown in Table 4, the proposed algorithm achieved a 9.93% higher value as compared to tiny YOLO v4, 1.8% higher value as compared to YOLO v3, 27.06% higher value as compared to EfficientNet-YOLO v4, and 12–15% higher value as compared to YOLO v1 and v2. The important point of consideration here is that YOLO v3 has seventy-five convolutional layers and EfficientNet-YOLO v4 has ninety-nine convolutional layers in its feature extraction and detection network whereas, the proposed ETL-YOLO v4 algorithm has only twenty-nine convolutional layers in its network and still have the ability to perform better and achieve higher detection accuracy. This is due to the wisely chosen filters for convolutional layers in the YOLO detection layers, addition of two extra YOLO detection layers and addition of SPP network that aid in producing a larger feature map and further pass on to the subsequent layers which performs feature extraction, localization, classification and detection. The results indicate that the proposed algorithm is a better face mask detection algorithm capable of detecting faces with masks and specifically, the face mask region which is a too small object to detect and an enthralling challenge in object detection as compared to other tested algorithms.

In order to get more intuitive and justifiable results, we tested the proposed ETL-YOLO v4 algorithm on a benchmark face mask detection dataset with the name MOXA. We selected 300 test images from the MOXA dataset and re-annotated those with the class labels of the employed dataset that is, with mask, without mask, mask incorrectly and mask area. Firstly, we tested the proposed ETL-YOLO v4 on test images of the MOXA dataset with weights trained on the employed face mask detection dataset and then performed a direct comparison with algorithms tested by authors in [20]. The results of the direct comparison of the proposed algorithm with the algorithms tested in [20] are shown in Table 5. The detection results as shown in Table 5 indicates that the proposed ETL-YOLO v4 algorithm for face mask detection achieved a better accuracy in terms of mean average precision (mAP) by 1.15–18.62% as compared to F-RCNN 300 Inception v2, SSD 300 MobileNetv2, and YOLO v3 having a deeper CNN feature extraction and detection network which requires a higher training time and need of high-end computation resources.

Fig. 6. ETL-YOLO v4 comparison with other algorithms based on average precision.
Table 4
Performance comparison of ETL-YOLO v4 with other algorithms based on mAP.

| Algorithm        | mAP  |
|------------------|------|
| YOLO v1          | 52.40% |
| YOLO v2          | 55.34% |
| YOLO v3          | 65.84% |
| EfficientNet-YOLO v4 | 40.04% |
| tiny YOLO v4     | 57.71% |
| Proposed ETL-YOLO v4 | 67.64% |

Table 5
Performance comparison of ETL-YOLO v4 on MOXA dataset.

| Work         | Algorithm       | Dataset | mAP   |
|--------------|-----------------|---------|-------|
| Roy et al. [20]    | tiny YOLO v3    | MOXA    | 56.27% |
|                 | SSD 300 MobileNet v2 |         | 46.52% |
|                 | F-RCNN 300 Inception v2 |     | 60.50% |
|                 | YOLO v3        |         | 63.99% |
| **Ours**      | Proposed ETL-YOLO v4 |   | 65.14% |

5. Conclusion and future work

Based on the ongoing pandemic of COVID-19 where wearing face masks is a mandate and to find a solution for people following COVID appropriate behavior of wearing face masks using artificial intelligence, deep learning, and object detection, the present work proposes ETL-YOLO v4 algorithm for face mask detection. The proposed algorithm is capable of detecting faces with masks, without masks, and specifically, face masks on the face region with high detection accuracy. The proposed ETL-YOLO v4 outperformed its original counterpart by a value of 9.93% for mAP. Furthermore, it outperformed the performance of tiny YOLO v4 for faces with masks by achieving a value higher by 5.53%, and specifically, for a small object such as a face mask, it achieved a 16.6% higher value for average precision (AP). The proposed ETL-YOLO v4 algorithm surpasses the performance of full-scale YOLO v3 by achieving a 1.8% higher value for mAP despite having a comparatively very small feature extraction network. The high detection accuracy achieved by the proposed algorithm is due to the necessary embodiments made in the architecture of tiny YOLO v4 such as, the inclusion of a dense SPP network for extraction of the rich feature map, the addition of two extra YOLO detection layers that allows detection at random scales, using Mish as the activation function over Leaky ReLU for faster training convergence and addition of Mosaic and CutMix augmentation at training time to provide more data to the YOLO network. The proposed work has high implications in the present time of the COVID-19 pandemic to develop detectors for surveillance of people to check if they are following the mandate of wearing a face mask or not. Furthermore, the proposed work can be extended to the use of conditional generative adversarial networks. The proposed algorithm can detect faces with masks and pass the mask regions for the training of generative adversarial networks with different viewpoints and create new samples of synthetic face masks.

Funding

Not applicable.

Declaration of Competing Interest

The authors report that they have no competing interests.

Availability of data and material

The face mask detection dataset used in this work is available on request at: https://drive.google.com/drive/folders/1_lBlgHXewEadtLFKtw8hnug_cOr6erbj?usp=sharing.

Acknowledgements

Not applicable.
References

[1] J. Howard, A. Huang, Z. Li, Z. Tufekci, V. Zdimal, H.M. Westhuizen, A.V. Delft, A. Price, L. Fridman, L.H. Tang, V. Tang, G.L. Watson, C.E. Bax, R. Shaikh, F. Questier, D. Hernandez, L.F. Chu, C.M. Ramirez, A.W. Rimoin, An evidence review of face masks against COVID-19, Proc. Natl. Acad. Sci., USA (2021), https://doi.org/10.1073/pnas.2021546118.

[2] L. Pepe, M. Cioffi, Data say about wearing face masks, Nature 586 (2021) 186–189, https://doi.org/10.1038/d41586-020-02801-4.

[3] M. Lucia, K. Vanja, V. Matjaž, H. Ciara, M. Helena, T. Zoran, B. Norbert, P. Mirjam, P. Barbara, G. Srecko, Face masks during the covid-19 pandemic: A simple protection tool with many meanings, Front. Public Health 8 (2021), https://doi.org/10.3389/fpubh.2020.606635.

[4] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: unified, real-time object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016. (https://doi.org/10.1109/CVPR.2016.91).

[5] J. Redmon, A. Farhadi, YOLO9000: better, faster, stronger, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017. (https://doi.org/10.1109/CVPR.2017.690).

[6] J. Redmon, A. Farhadi, YOLOv3: an Incremental Improvement, arXiv (2018). (https://arxiv.org/abs/1804.02767).

[7] A. Bochkovskiy, C.Y. Wang, H.Y. Lio, YOLOv4: optimal speed and accuracy of object detection, arXiv (2020). (https://arxiv.org/abs/2004.10934).

[8] R. Huang, J. Pedoenn, C. Chen, YOLO-LITE: a real-time object detection algorithm optimized for non-gpu computers, arXiv (2018). (https://arxiv.org/pdf/1811.05858.pdf).

[9] R.C. Hendry, Automatic license plate recognition via sliding-window darknet-YOLO deep learning, Image Vis. Comput. 87 (2019) 47–56, https://doi.org/10.1016/j.imavis.2019.04.007.

[10] Z. Yi, S. Yongliang, Z. Jun, An improved tiny-yolov3 pedestrian detection algorithm, Optik 183 (2019) 17–23, https://doi.org/10.1016/j.ijleo.2019.02.038.

[11] S. Singh, U. Ahuja, M. Kumar, K. Kumar, M. Sachdeva, Face mask detection using YOLOv3 and faster R-CNN models: COVID-19 environment, Multimed. Tools Appl. 80 (2021) 19753–19768, https://doi.org/10.1007/s11042-021-10711-8.

[12] J. Yu, W. Zhang, Face mask wearing detection algorithm based on improved yolo-v4, Sensors 21 (9) (2021) 3263, https://doi.org/10.3390/s21093263.

[13] R. Liu, Z. Ren, Application of yolo on mask detection task, arXiv (2021). (https://arxiv.org/ftp/arxiv/papers/2102/2102.05402.pdf).

[14] G. Cheng, S. Li, Y. Zhou, A mask detection system based on yolov3-tiny, Front Soc. Sci. Tech. 2 (11) (2020) 33–41, https://doi.org/10.25236/FSTT.2020.021106.

[15] Z. Zhao, K. Hao, X. Ma, X. Liu, T. Zheng, J. Xu, S. Cui, SAI-YOLO: a lightweight network for real-time detection of driver mask-wearing specification on resource-constrained devices, Comput. Intell. Neurosci. 2021 (2021), 4529107, https://doi.org/10.1155/2021/4529107.

[16] M. Loqvi, J. Cano, Accelerating deep learning applications in space, in: Proceedings of the 22nd Annual Conference on Artificial Intelligence (AAAI), New York, USA, 2020.

[17] M. Lofqvist, J. Cano, Accelerating deep learning applications in space, in: Proceedings of the 22nd Annual Conference on Artificial Intelligence (AAAI), New York, USA, 2020.

[18] M. Loey, G. Manogaran, M.H.N. Taha, N.E.M. Khalifa, A hybrid deep transfer learning model with machine learning methods for face mask detection in the era of COVID-19, Optik 239 (2021), 166744, https://doi.org/10.1016/j.ijleo.2019.02.038.

[19] M. Loey, G. Manogaran, M.H.N. Taha, N.E.M. Khalifa, Fighting against covid-19: A novel deep learning model based on yolo-v2 with resnet-50 for medical face mask detection, Sustain. Cities Soc. 65 (2020), 102600, https://doi.org/10.1016/j.scs.2020.102600.

[20] G.J. Chowdary, N.S. Punn, S.K. Sonbhadra, S. Agarwal, Face mask detection using transfer learning ofyolo with darknet, Optik 183 (2019) 17–23, https://doi.org/10.1016/j.ijleo.2019.02.038.

[21] M. Inamdar, N. Mehendale, Real-Time face mask identification using face masknet deep learning network, SSRN (2020), https://doi.org/10.2139/ssrn.3653289.

[22] A. Kumar, A. Kalia, K. Verma, A. Sharma, M. Kaushal, A hybrid tiny YOLO v4-SPP module based improved face mask detection vision system, J. Ambient Intell. Hum. Comput. 80 (2021) 509, https://doi.org/10.1007/s12652-021-00354-x.

[23] H. Kaiming, Z. Xiangyu, R. Shaoqing, S. Jian, Spatial pyramid pooling in deep convolutional networks for visual recognition, arXiv (2014). (https://arxiv.org/abs/1406.4729).

[24] Z. Zheng, P. Wu, W. Li, L. Li, R. Ye, D. Ren, Distance-IoU loss: faster and better learning for bounding box regression, in:Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), New York, USA, 2020. (https://doi.org/10.1609/aaai.v34i07.6999).

[25] D. Misra, A self-regularized non-monotonic activation function, arXiv (2019). (https://arxiv.org/abs/1908.08681).

[26] S. Yun, D. Han, S.J. Oh, S. Chun, J. Choe, Y. Yoo, CutMix: Regularization strategy to train strong classifiers with localizable features, arXiv (2019). (https://arxiv.org/abs/1905.05287).

[27] S. Ren, K. He, R. Girshick, J. Sun, Towards real-time object detection with region proposal networks, arXiv (2015). (https://arxiv.org/abs/1506.01497).

[28] S. Nowozin, Optimal decisions from probabilistic models: the intersection-over-union case, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, OH, USA, 2014. (https://doi.org/10.1109/CVPR.2014.7).

[29] F. Ahmed, D. Tarlow, D. Batra, Optimizing expected intersection-over-union with candidate-constrained CRFs. in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 2015. (https://doi.org/10.1109/ICCV.2015.215).

[30] H. Takimoto, Y. Sato, A.J. Nagano, K.K. Shimizu, A. Kanagawa, Using a two-stage convolutional neural network to rapidly identify tiny herbivorous beetles in the field, Ecol. Inform. 66 (2021), 101466, https://doi.org/10.1016/j.ecoinf.2021.101466.