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.
Chapter 4

Combating COVID-19 using object detection techniques for next-generation autonomous systems

Hrishikesh Shenai, Jay Gala, Kaustubh Kekre, Pranjal Chitale and Ruhina Karani
D. J. Sanghvi College of Engineering, Mumbai, India

4.1 Introduction

Object detection is a rapidly growing technology in the field of computer vision, which deals with the detection of objects belonging to a particular class like vehicles, humans, etc. in digital media (videos and images). It is considered as a combination of object localization and image classification techniques as it deals with identifying as well as locating the position of objects belonging to a certain predefined class in the image. The result of object detection algorithms can be displayed either by bounding boxes or by highlighting the pixels belonging to the object found. Approaches based on either deep learning or machine learning are used for object detection. In machine learning-based approaches, each object class is described by some unique features that help in the classification process—for example, spherical balls would appear as a circle in a 2D image. Such predefined features are extracted using feature extraction techniques like the Viola–Jones framework based on Haar features, after which classification is carried out using classification techniques such as support vector machines (SVM) to identify the class. On the other hand, deep learning techniques can perform end-to-end object detection without the need to specify any features, thus saving time. These deep learning methods are generally based on convolutional neural networks (CNN).

The following sections of the chapter explore the architecture and working of some of the most commonly used deep learning-based object detection algorithms: the region-based convolutional neural network (R-CNN) family...
and the You Only Look Once (YOLO) family. In the current pandemic situation, these techniques can be of real help in various applications, some of which are described in this chapter. The chapter also proposes a pothole detection system as one of the applications built on YOLOv3, along with results tested on different types of potholes.

4.2 Need for object detection

The sense of vision is one of the most important human senses. Humans can understand the visual scene given in an image or video and gain a lot of information from it. This happens because the human brain detects several entities observed by the eyes and understands the context by analyzing the correlation between these entities. Thus for visual understanding, there is a need to detect the different entities present in visual content. Today, in the age of automation, there is a need for Artificial Intelligence (AI) based systems to perform a multitude of tasks related to vision to reduce the dependency on human workers or, in some cases, to aid them. The AI-based systems need to work similarly to the human brain to detect objects to analyze them, interpret the scene, and perform any further tasks based on the interpretation. Thus object detection is an essential phase of AI-based vision systems. The world is plagued by the COVID-19 pandemic and such AI-based systems can play a major role in helping to mitigate the effects of this pandemic by reducing the dependency on human workers and thus saving lives. Object detection can find its use in various applications that serve to aid in the COVID-19 crisis. Few of the applications are as follows:

- Face mask detector and social distancing detector: To ensure that the citizens are following the safety guidelines in public places.
- COVID-19 detector based on X-rays: To detect the abnormalities in chest X-rays even before the lab reports confirm the clinical symptoms.
- Module for autonomous systems: To help in various sectors to reduce the dependency on human workers, for example, in the transportation sector, where such a system can play a crucial role in the monitoring of road conditions through the use of street cameras. The pothole detection system proposed in the chapter is an example of such a system.

4.3 Object detection techniques

Object detection deals with detecting all the instances of objects present in an image, where these objects belong to particular classes. The object detection models that are based on deep learning contain two major components: an encoder and a decoder. The input image is fed to the encoder to learn and extract the features that are significant for locating and labeling the objects. The encoder output is fed to the decoder to predict the bounding box position
and the corresponding label for the object. The metric used to grade the
object location predicted by the algorithm is called intersection over union
(IOU). Considering the prediction of the model and the ground truth bound-
ing box, compute the area of intersection of the ground truth bounding box
and the predicted bounding box, then divide it by the union of the two. The
value of IOU is in the range of $0 \leq 1$. IOU 0 indicates no intersection
between the predicted bounding box and ground truth bounding box, while
IOU 1 indicates the predicted bounding box is perfectly overlapping with the
ground truth bounding box. In this chapter, some of the most commonly
used object detection algorithms like the R-CNN family and the YOLO fam-
ily are described. R-CNN utilizes a selective search algorithm to propose the
Region of Interest (RoI) in an image followed by a CNN to detect the pre-
sence of the object of interest in those particular regions, thus following a
two-step process. YOLO is a single step process where a single neural net-
work is applied to the entire image and then the image is divided into smaller
regions and bounding boxes and probability for each region is predicted.

4.3.1 R-CNN family

R-CNN stands for a region-based convolutional neural network. The R-CNN
family of object detectors follow two-stage object detector architecture. In a
two-stage detector, an RoI pooling layer separates the stages. Fig. 4.1 shows
the architecture of a two-stage detector. The candidate object bounding boxes
are proposed by a Region Proposal Network (RPN). This acts as the first
stage of the detector. While in the second stage, the process of feature
extraction is performed by the RoI pool (RoI pooling) operation by

![FIGURE 4.1 General architecture of a two-stage object detector.](image-url)
considering every candidate box for performing further tasks of classification and bounding-box regression. The advantage of using two-stage detectors is high localization and object recognition accuracy.

The main aim of this family is to improve model performance. Fig. 4.2 summarizes the network architecture of object detectors in the R-CNN family (Girshick, Donahue, Darrell, & Malik, 2014; Girshick, 2015; Ren, He, Girshick, & Sun, 2017).

### 4.3.1.1 R-CNN

R-CNN is the first model in the R-CNN family and was proposed by Girshick et al. (2014).

#### 4.3.1.1.1 Network architecture

A region proposal algorithm is used to identify a feasible number of region proposals in the input image and then feature maps are extracted using CNN from each region independently for classification. R-CNN uses the selective search algorithm to extract just 2000 regions from the image, which are referred to as region proposals. A 4096-dimensional feature vector is extracted from each region proposal using AlexNet (proposed by Krizhevsky et al.). AlexNet requires the input to be of size $227 \times 227$; therefore the region proposals are resized irrespective of aspect ratio or size. The last softmax layer present in AlexNet is removed and fully connected (FC) layers are added. The input image is fed to a series of five convolution layers and then...
passed through two FC layers, which are 4096 dimensional to get a 4096-dimensional feature map. This network architecture is shown in Fig. 4.2. The class of the object, which is represented by the feature vector, is detected by using an SVM per object class (one-vs-one strategy). Each SVM outputs a score for that particular class. This score is an indicator of the likelihood that the region proposal belongs to that particular class. The region proposal is assigned a class label corresponding to the SVM with the highest score (Krizhevsky, Sutskever, & Hinton, 2017).

4.3.1.1.2 Advantages

1. The region proposal algorithm provides a computationally less expensive method of identifying a feasible number of region proposals.

4.3.1.1.3 Disadvantages

1. The SVM is trained on the feature vectors that are generated by AlexNet. That means that training the SVM classifier is not possible before CNN is fully trained. This implies that the training process is not parallelizable.
2. R-CNNs require longer training.
3. For real-time systems, R-CNN is not feasible (it takes 47 seconds for detecting objects in each test image).
4. The selective search algorithm is a fixed algorithm (no learning procedure happens at each stage). This may result in bad candidate region proposals.

4.3.1.2 Fast R-CNN

Fast R-CNN was proposed by Ross Girshick (Girshick, 2015) in 2015 as an improved version of R-CNN. R-CNN was a major development in object detection; however, it had some shortcomings. In R-CNN, for each region proposal, a feature map was calculated, making it slow. Since each feature map was saved, memory requirements were high. The process becomes complicated as a result of separate training of Bounding Box Regressor, CNN, and SVM. Fast R-CNN takes the entire image as an input, from which features are extracted and passed through an RoI pooling layer. This layer outputs a set of fixed-sized feature maps that form the inputs for further classification and localization.

4.3.1.2.1 Network architecture

The Fast R-CNN takes the entire image as an input along with some object proposals. The image is first processed by a network using multiple convolutional layers coupled with max pooling layers to extract a convolutional feature map. The extracted feature map is then passed to the RoI pooling layer. The network architecture is shown in Fig. 4.2.
4.3.1.2.2 The RoI pooling layer
This layer helps to achieve a significant speedup in training and testing while maintaining high accuracy. The layer takes two inputs:

1. Fixed sized feature map from the previous CNN layers.
2. An $N \times 5$ matrix, a list of $N$ RoIs. The first column is the index of the image, while the remaining four columns define the positions of the corners of the bounding boxes.

The layer first divides the region proposal window (of size $h \times w$) into several equal-sized ($H \times W$) subwindows. Max pooling is applied in each of the subwindows, across each channel independently, to generate the corresponding output value. The output dimensions depend only on the number of sections into which the proposal is divided. The benefit of this layer is that even if multiple objects are present in a proposal, the same feature map can be used for each of them, thus reducing the overall processing speed. Each feature vector is then given as an input to a set of FC layers. The output of these layers is forked into the two separate layers. The first layer is a softmax classification layer, wherein the object classes are identified. The second layer is a Bounding Box Regressor layer that generates a tuple of four real-valued numbers, which define the positions of the corners of the bounding box for each of the “K” object classes (Girshick, 2015; Grel, 2017).

4.3.1.2.3 Advantages
1. Fast R-CNN has an improved mean average precision (mAP) over R-CNN.
2. It uses an RoI pooling layer to get a fixed-sized feature map, reducing computation time.
3. It uses truncated Singular Value Decomposition (SVD) to reduce the time required for the computation of FC layers.

4.3.1.2.4 Disadvantages
1. Searching for region proposals takes a lot of time and in the overall running of the algorithm; this slows down the entire architecture.

4.3.1.3 Faster R-CNN
Selective search is utilized by the algorithms discussed earlier (R-CNN and Fast R-CNN) to determine the region proposals. It is heavily time-consuming due to its taxing Central Processing Unit (CPU) implementation, which takes around 2 seconds per image, thereby affecting the performance of the overall network for object detection. Ren et al. (2017), in 2015, proposed a computationally efficient object detection algorithm that unifies the RPNs with the Fast R-CNN. This algorithm reduces the computation time, for determining the region proposals, from 2 seconds to 10 milliseconds improving the
performance of the overall network for object detection. In short, the Faster R-CNN algorithm comprises two modules: region proposal by the RPN followed by the Fast R-CNN detector.

4.3.1.3.1 Network architecture

RPN: The RPN takes an input image and then outputs the region proposals along with their objectness score. The RPN and Fast R-CNN share a common set of convolutional layers. The network has to determine whether an object is present in the input image by generating the region proposals by sliding a small network over the convolutional feature map obtained from the last shared convolutional layer. The sliding window is mapped to a lower-dimensional feature subspace (512-d for VGGNet and 256-d for ZFNet). The feature space is fed into the two FC layers—a regression layer (reg) and a classification layer (cls). As the network slides through the convolutional feature map, it checks whether “k” corresponding anchors contain an object and refines the anchor coordinates to give the bounding boxes. Fig. 4.3 shows the RPN (Ananth, 2019; Simonyan & Zisserman, 2015; Szegedy et al., 2015; Xia, Chen, Wang, Zhang, & Xie, 2018; Zeiler & Fergus, 2014).

For k anchors boxes, the regression layer outputs 4k coordinates of the bounding boxes and the classification layer outputs 2k scores that estimate the probability of the presence of an object for each region proposal. An

**FIGURE 4.3** Region proposal network. Reprinted from Xia, D., Chen, P., Wang, B., Zhang, J., & Xie, C. (2018). Insect detection and classification based on an improved convolutional neural network, Agricultural Sensing and Image Analysis, 18, 2018, licensed under CC by 4.0.
anchor is assigned a positive label if it satisfies either of the two conditions: (1) the anchors with the highest IOU, that is, a measure of overlap with the ground truth box or (2) an anchor that has an IOU greater than 0.7. An anchor is assigned a negative label if its IOU value is lower than 0.3. The anchors that do not satisfy either of the conditions are discarded.

Detector: The convolutional layers are shared by both RPN and Fast R-CNN. The region proposals from the RPN are passed through the RoI pooling layer to obtain feature vectors. These feature vectors are then fed into the sibling classification and regression branches, which are different from those in the RPN. The classification layer estimates the probability of the proposal belonging to a particular class and the regression layer outputs the coordinates of the predicted bounding boxes whose size is specific to each class. The network architecture is shown in Fig. 4.2.

4.3.1.3.2 Advantages
1. It eliminates the CPU-based selective search, which makes the overall network faster.
2. It uses a shared convolutional layer for the RPN and Fast R-CNN instead of separate convolutional networks.

4.3.1.3.3 Disadvantages
1. To extract all the objects from an image, more than one pass is required through a single image.
2. It consists of different modules working one after the other, so the overall performance is proportional to the performance of the previous modules.

4.3.2 YOLO family

YOLO is the abbreviation for “You Only Look Once,” which suggests that given an input image, it is possible to detect and localize the objects present in a single glance. This technique treats object detection as a regression task. The YOLO family of object detectors follows one-stage detector architecture, as shown in Fig. 4.4.

One-stage detectors do not have the region proposal step. Thus the bounding boxes are predicted by considering the input images directly and

FIGURE 4.4 General architecture of an one-stage object detector.
so a separate region proposal step is not needed. One-stage detectors are efficient concerning time because of their high inference speed and are thus highly suitable for real-time devices. Fig. 4.5 summarizes the network architecture of object detectors in the YOLO family (Redmon & Farhadi, 2018; Redmon, Divvala, Girshick, & Farhadi, 2016; Seong, Song, Yoon, Kim, & Choi, 2019).

4.3.2.1 YOLOv1

YOLO, as the name suggests, aims to look at the image once to detect and localize the objects present in the image. It was proposed by Joseph Redmon et al. (Redmon et al., 2016) in 2015. It has a simple architecture that consists of a single convolutional network that predicts objects across various classes simultaneously by using feature maps obtained from the entire input image. This design helps to provide end-to-end training with real-time speeds while achieving high accuracies. YOLO splits the input image in an S × S grid. Each cell of this grid is responsible for detecting an object if the object center is present in it. Each cell predicts “B” number of bounding boxes and also generates an objectness score for it. This score reflects the accuracy of the detection of the cell. The objectness score will be zero if no object is in the cell; otherwise, it is the IOU between the ground truth box and the predicted box.

4.3.2.1.1 Network architecture

- YOLO is essentially based on GoogLeNet. It has 24 convolutional layers, which are followed by a couple of FC layers. The inception modules of
GoogLeNet are replaced by reduction layers (1 × 1) along with (3 × 3) convolutional layers (Szegedy et al., 2015).

- The output of the network is a (7 × 7 × 30) tuple. Fig. 4.5 shows this network architecture.

### 4.3.2.1.2 Advantages

1. It is extremely fast. The images can be processed at 45 frames per second, which is much faster than Faster R-CNN.
2. Background errors made by Fast R-CNN is about 13.6%, while that of YOLO is less than 4.75%.

### 4.3.2.1.3 Disadvantages

1. It is not very accurate despite being very fast.
2. The model fails to detect small objects in groups accurately (like a bird flock).
3. The model has difficulty in generalizing objects with unusual configurations or aspect ratios.

### 4.3.2.2 YOLOv2

This version of YOLO focuses mainly on improving localization and recall (a ratio of true object detections to the total number of objects in the data) while also maintaining classification accuracy. It was proposed by Redmon and Farhadi (2017) in 2016.

#### 4.3.2.2.1 Improvements made over YOLOv1

- Batch normalization: It is used on all convolutional layers. Results in a 2% improvement in mAP.
- High-resolution classifier: YOLOv2 uses additional 416 × 416 images for fine-tuning the classification network. This is carried out for 10 epochs on ImageNet. Results in 4% improvement in mAP.
- Convolutions with anchor boxes: All FC layers are removed and anchor boxes are used to predict bounding boxes. With anchor boxes, recall is 88% and a mAP of 69.2%.
- Direct location prediction: Logistic activation $\sigma$ is used for location binding. This makes the value fall from 0 to 1. This results in 5% improvement in mAP.
- Fine-grained features: Large objects can be detected using a 13 × 13 feature map. While to detect smaller objects properly, the maps from the earlier layer are mapped into a 13 × 13 × 2048 feature map and then concatenated with the original 13 × 13 feature maps. This results in 1% improvement in mAP.
4.3.2.2 Network architecture

- Darknet-19 acts as a backbone for YOLOv2. It has 19 convolutional and five max pooling layers.
- Here the output shape is \((13, 13, (k(1 + 4 + 20)))\). As the number of classes is 20 while the number of anchor boxes is denoted by \(k\). Fig. 4.5 shows this network architecture.

4.3.2.2.3 Advantages

1. It has considerable improvement in the IOU score.
2. The mAP was also improved.

4.3.2.2.4 Disadvantages

1. It is not suitable for a real-time production system due to its slowness.
2. It cannot detect smaller objects and objects with unusual aspect ratios.

4.3.2.3 YOLOv3

YOLOv2 had certain drawbacks such as (1) not good at detecting small objects, (2) assuming that the object classes are mutually exclusive. Redmon and Farhadi (2018) in 2018 proposed a new architecture for the YOLO known as YOLOv3, which tries to solve the problems faced by YOLOv2 architecture by collectively accumulating good ideas from other architectures. YOLOv3 boosts the accuracy of the overall network.

4.3.2.3.1 Improvements made over YOLOv2

- Bounding box prediction: For every bounding box, logistic regression is used to determine the “objectness score.”
- In case of overlap between the preceding bounding box and a ground truth object, we get an objectness score of “1.” Only one bounding box gets assigned for each of the ground truth objects.
- Class prediction: YOLOv3 eliminates the softmax classification and uses the multilabel classification for predicting the classes for the bounding boxes. During training, a binary cross-entropy loss is used for the class predictions.
- Prediction across scales: YOLOv3 predicts bounding boxes by extracting the features at three different scales, much similar to the Feature Pyramid Network. Each bounding box prediction comprises four bounding offsets, 1 objectness score and 80 class scores. K-means clustering is used for determining the bounding boxes (Lin et al., 2017).
4.3.2.3.2 Network architecture
YOLOv3 uses a hybrid Darknet-53, which consists of 53 layers of Darknet and an additional 53 layers leading to a total of 106 layers. Fig. 4.5 shows this network architecture.

4.3.2.3.3 Advantages
1. Average precision is improved for the small objects and is better than Faster R-CNN.
2. mAP was increased significantly and localization errors were reduced.
3. Predictions at different scales improved.

4.3.2.3.4 Disadvantages
1. Average precision for the large and medium objects can be improved.
2. mAP score between 0.5 and 0.95 IOU can still be increased.

4.4 Applications of objection detection during COVID-19 crisis
Object detection is useful in various applications, which will be crucial in combating the COVID-19 crisis. Few of the applications are as follows.

4.4.1 Module for autonomous systems (pothole detection)
The pandemic has affected various sectors that are dependent on physical labor. With lockdowns being imposed for curbing the spread of this deadly disease, the sectors that depend on physical human labor have been affected the most. One such area is the transportation sector, where the road conditions have to be monitored and maintained continuously. An AI-based autonomous system would play a crucial role in the monitoring of road conditions through the use of surveillance cameras where the road anomalies can be detected using the object detection module. This module would detect potholes using encoded representation of images. As most of the cities have now become “smart cities,” most of the roads are covered by surveillance cameras spanning across the city. The YOLO family is used here since pothole detection is a real-time system and the R-CNN family, though more accurate, is slower as compared to the YOLO family.

4.4.1.1 Architecture
Dataset used: Dataset consists of 1500 images taken with a mobile phone mounted on the dashboard of a car. Images are captured with different lighting conditions and climatic conditions and are annotated according to YOLO standards using an annotation tool. The train test split is 80–20.
Network architecture: YOLOv3 from the YOLO family is used as the network architecture. The architecture is based on the pretrained weights of a hybrid Darknet-53. It is tuned to obtain the required mAP value. Parameters like batch size and subdivisions can be tweaked to match the training system’s capabilities. The filter size is $18 \times 18$ (Redmon & Farhadi, 2018).

### 4.4.1.2 Results

The proposed system gives an average IoU value of 0.72 while the YOLOv2 model has a value of 0.61 (both trained on the above-mentioned dataset). Fig. 4.6A and B show pothole detection achieved with the help of YOLOv3 through experiments conducted.

### 4.4.2 Social distancing detector

COVID-19 spreads among people who stay nearby (about 6 feet) for long periods. The virus spreads when an infected person coughs, sneezes, or even talks and droplets containing the virus are launched into the air. These droplets can travel a distance of about 6 feet and reach the mouth or nose of other people and get inhaled into their lungs. These infected people may not show symptoms; however, they play a role in the spreading of the virus. Social distancing helps to restrict the opportunities of coming in contact with infected people outside the home. It is important since this would reduce the spread of the virus. Social distancing has to be practiced strictly to stop the spread of this virus. It has been found through studies that the spread of this virus is not dependent on the age of the person and thus it is of paramount importance that social distancing is practiced by every single person (Bhatnagar et al., 2020). Several police officials have to risk their lives every day by being out in the field to make sure that social distancing is being maintained. Social distancing detector can help such officers by allowing them to monitor various public places remotely. Object detection would be an important module to detect people and analyze the distance between them. If the distance between the detected people is below a particular threshold then the respective

---

**FIGURE 4.6**  A. Waterlogged pothole. B. Normal pothole.
authorities would be alerted to take the necessary measures. Fig. 4.7 clearly describes how social distancing has helped in reducing the number of positive cases (Punn, Sonbhadra, & Agarwal, 2020).

Dataset used: Open image dataset. Images are resized such that the shorter edge contains “P” pixels, which range from 600 for a low resolution to 1024 for high resolution. Along with this, surveillance footage from Oxford Town Center is utilized, which is also further used for demonstrating the overall working of the module (Google, 2020; Wojke, Bewley, & Paulus, 2017).

Approach: Faster R-CNN model or YOLOv3 can be used for object detection with Deepsort for tracking objects surrounded by bounding boxes. The bounding boxes are then analyzed to identify people not following social distancing. Along with this, every bounding box is color-coded to keep track of people of the same group by having the same colored bounding boxes for them. The output also consists of a streamline plot showing the statistical data like the number of groups and ratio of the number of people to the number of groups, which can be helpful to determine the level of risk at a particular place. The workflow of the system is as follows:

1. Each detected person is associated with a tuple \((x, y, d)\) where \((x, y)\) is the centroid of the coordinates of the bounding boxes, \(d\) is the approximate depth of that person as viewed from the camera.
2. The pairwise L2 norm is then calculated. This is used to determine the closeness of an individual with each of the neighbors. The closeness threshold is updated constantly.
3. The color of the bounding box of a person is changed when they satisfy the closeness criteria of some other person, thereby indicating the
formation of groups. This indicates a violation of social distancing norms (Punn et al., 2020).

### 4.4.2.1 Results

The proposed system using faster R-CNN gives an mAP value of 0.969, while YOLOv3 has a value of 0.846 (Punn et al., 2020).

Fig. 4.8 shows the working of the model where the first image has a violation index of 3 (presence of larger groups of people) and the second Image has a violation index of 2 (fewer people in immediate proximity) (Punn et al., 2020).

### 4.4.3 COVID-19 detector based on X-rays

Most of the severe health conditions are observed as a result of the late detection of the infection. Because of the tremendous pressure on the healthcare department, there is a delay in the receipt of the actual laboratory diagnosis reports leading to the deterioration of the health condition of the patient. The test samples collected are being sent to the specialized laboratories in urban areas and this leads to a considerable delay, especially for rural areas. Had this delay been reduced, some precautionary medication could have been administered early, which might have helped in ameliorating the patient’s health condition. Moreover, early detection would have helped ensure that the patient is quarantined, thus restricting the further spread of the virus. X-ray machines are already present in most of the healthcare systems available in rural areas and by using these X-ray systems, the transportation time can also be saved as well as the pressure on the laboratories would be reduced to some extent. An AI-based COVID-19 detector can be developed, which could detect the abnormalities even before the lab reports confirm the clinical symptoms to detect the presence of the virus based on X-ray imagery. An object detection module is needed in such AI systems.
that will detect abnormalities from X-ray imagery that could be indicators of the virus. This will not only help the doctors administer immediate medical aid at an initial stage but also help to curtail the spread of the virus.

### 4.4.3.1 Architecture

Dataset used: Dataset prepared by Cohen, Morrison, and Dao (2020) by collecting X-ray images from various sources (open-access). This dataset contains chest X-ray images of 82 male patients and 43 female patients who were diagnosed as positive cases. Few samples from the ChestX-ray8 dataset that has been provided by Wang et al. (2017), which consists of chest X-ray images of normal patients and pneumonia patients, have been used.

DarkCovidNet: Darknet-19 model is used as a starting point. Each DarkNet layer consists of one convolutional layer, which is followed by BatchNorm and Leaky Rectified Linear Unit (ReLU) activation operation. Max pooling is used in all the pooling operations. The learning rate of $3 \times 10^{-3}$ is considered. A cross-entropy loss function is used along with Adam optimizer for updating the model parameters.

#### 4.4.3.1.1 Results

The proposed system has an accuracy of 98.08%. The model highlights imperfections in the X-ray images using heatmaps. Fig. 4.9 shows the heatmaps of corresponding chest X-ray images (Ozturk et al., 2020).

### 4.4.4 Face mask detector

Face masks act as the first line of defense against this deadly virus. Wearing a face mask prevents the spread of infection while also protecting the users. Face masks eliminate cross-contamination. Automatic face mask detection systems can help governments to monitor the citizens and ensure that they are taking necessary precautions (wearing face masks when going in public places) to ensure that the virus is not spreading. Object detection can be used

![FIGURE 4.9 X-ray images and corresponding heatmaps. Reprinted from Ozturk, T., Talo, T., Yildirim, E.A., Baloglu, U.B., Yildirim, O., & Acharya, U.R. (2020). Automated detection of COVID-19 cases using deep neural networks with X-ray images, Computers in Biology and Medicine, 121, 1–11.](image)
for detecting the presence of face masks on faces of the people, thereby proving if they are wearing a face mask or not (Jeremy et al., 2020).

4.4.4.1 Architecture

Dataset used: Face dataset of 1376 images divided between two classes:

- Without_mask: 686 images
- Mask: 690 images

The method used for creating the dataset:

1. Take normal face images (without masks).
2. Use a custom computer vision python script to append face masks on them.

The only precaution to be taken is that the images used to create the “mask” dataset cannot be used in the “Without_mask” dataset. The model becomes biased and would not generalize well if the original images are used as “Without_mask” samples (Rosebrock, 2020).

Network architecture: The proposed system uses YOLOv3 from the YOLO family for face mask detection. It uses DarkNet-53 for feature detection (Rodriguez & Lorenzo, 2020).

4.4.4.1.1 Results

Fig. 4.10 shows the results of face mask detection using YOLOv3 (Rodriguez & Lorenzo, 2020).

4.5 Conclusion

Object detection is a widely growing computer vision technique that is essential for understanding and analyzing visual scenes in a video or a photo.
It involves two processes: identifying and locating objects of different predefined classes. Deep learning-based approaches use CNNs to perform the task of object detection in an end-to-end, unsupervised manner. R-CNN and YOLO families, as discussed in the chapter, are popularly used for object detection. Each family has some trade-offs between speed and accuracy. R-CNN family algorithms have higher accuracy but slower speeds than the YOLO family, whereas YOLO family algorithms are computationally faster but have slightly lower accuracy than the R-CNN family. Object detection can be helpful in mitigating the transmission of the COVID-19 virus by automating monitoring systems, thereby reducing human dependence. Few other applications of object detection include self-driving cars, video surveillance, and anomaly detection.

References

Ananth, S. (2019) *Faster R-CNN for object detection*. Retrieved from https://towardsdatascience.com/faster-r-cnn-for-object-detection-a-technical-summary-474e5b857b46, Medium.

Bhatnagar, V., Poonia, R. C., Nagar, P., Kumar, S., Singh, V., Raja, L., & Dass, P. (2020). Descriptive analysis of COVID-19 patients in the context of India. *Journal of Interdisciplinary Mathematics*, 1–16. Available from https://doi.org/10.1080/09720502.2020.1761635.

Cohen, J.P., Morrison, P., & Dao, L. (2020). COVID-19 image data collection, e-print arXiv:2003.11597, arXiv.

Girshick, R. (2015). Fast R-CNN. *International Conference on Computer Vision (ICCV)*, 1440–1448. Available from https://doi.org/10.1109/ICCV.2015.169, IEEE.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 580–587. Available from https://doi.org/10.1109/CVPR.2014.81, IEEE.

Google. (2020). Open image dataset v6, Retrieved from https://storage.googleapis.com/openimages/web/index.html, Google.

Grel, T. (2017). *Region of interest pooling explained*. Retrieved from https://deepsense.ai/region-of-interest-pooling-explained, deepsense.ai.

Jeremy, H., et al. (2020). Face masks against COVID-19: An evidence review. *Preprints, 30* (20), 1–9. Available from https://doi.org/10.20944/preprints202004.0203.v2.

Krizhevsky, A., Sutskever, I., & Hinton, G. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6), 84–90. Available from https://doi.org/10.1145/3065386, ACM.

Lin, T. Y., Dollar, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2117–2125. Available from https://doi.org/10.1109/CVPR.2017.106.

Ozturk, T., Talo, M., Yildirim, E. A., Baloglu, U. B., Yildirim, O., & Acharya, U. R. (2020). Automated detection of COVID-19 cases using deep neural networks with X-ray images. *Computers in Biology and Medicine, 121*, 1–11. Available from https://doi.org/10.1016/j.compbiomed.2020.103792, ScienceDirect.

Punn, N.S., Sonbhudra, S.K., & Agarwal, S. (2020). Monitoring COVID-19 social distancing with person detection and tracking via fine-tuned YOLO v3 and Deepsort techniques, e-print arXiv:2005.01385v2, arXiv.
Redmon, J., & Farhadi, A. (2017). YOLO9000: Better, faster, stronger. Conference on Computer Vision and Pattern Recognition (CVPR), 6517–6525. Available from https://doi.org/10.1109/CVPR.2017.690.

Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement, preprint arXiv:1804.02767, arXiv.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, real-time object detection. Conference on Computer Vision and Pattern Recognition (CVPR), 779–788. Available from https://doi.org/10.1109/CVPR.2016.91, IEEE.

Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. Transactions on Pattern Analysis and Machine Intelligence, 39 (6), 1137–1149. Available from https://doi.org/10.1109/TPAMI.2016.2577031, IEEE.

Rodriguez, L., & Lorenzo, A. (2020) Face mask detector using deep learning (YOLOv3). Retrieved from https://medium.com/face-mask-detector-using-deep-learning-yolov3/face-mask-detector-using-deep-learning-yolov3-209b57177e92, Medium.

Rosebrock, A. (2020). COVID-19: Face mask detector with OpenCV, Keras/TensorFlow and deep learning. Retrieved from https://www.pyimagesearch.com/2020/05/04/covid-19-face-mask-detector-with-opencv-keras-tensorflow-and-deep-learning-Pyimagesearch.

Seong, S., Song, J., Yoon, D., Kim, J., & Choi, J. (2019). Determination of vehicle trajectory through optimization of vehicle bounding boxes using a convolutional neural network. Sensors (Basel), 19(01), 1–18. Available from https://doi.org/10.3390/s19194263, Sensors.

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. International Conference on Learning Representations (ICLR), preprint arXiv:1409.1556, arXiv.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., . . . Rabinovich, A. (2015). Going deeper with convolutions. Conference on Computer Vision and Pattern Recognition (CVPR), 1–9. Available from https://doi.org/10.1109/CVPR.2015.7298594.

Wang, X., Peng, Y., Lu, L., Lu, Z., Bagheri, M., & Summers, R. M. (2017). ChestX-Ray8: Hospital-scale chest X-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. Conference on Computer Vision and Pattern Recognition (CVPR), 3462–3471. Available from https://doi.org/10.1109/CVPR.2017.369.

Wojke, N., Bewley, A., & Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. International Conference on Image Processing (ICIP), 3645–3649. Available from https://doi.org/10.1109/ICIP.2017.8296962.

Xia, D., Chen, P., Wang, B., Zhang, J., & Xie, C. (2018). Insect detection and classification based on an improved convolutional neural network. Agricultural Sensing and Image Analysis, 18. Available from https://doi.org/10.3390/s1812416910.3390/s18124169.

Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional neural networks. European Conference on Computer Vision (ECCV), 818–833. Available from https://doi.org/10.1007/978-3-319-10590-1_53.