Arabian Journal for Science and Engineering (2022) 47:10811–10822
https://doi.org/10.1007/s13369-022-06841-2

RESEARCH ARTICLE-COMPUTER ENGINEERING AND COMPUTER SCIENCE

C-COVIDNet: A CNN Model for COVID-19 Detection Using Image Processing

Neha Rajawat[1] · Bharat Singh Hada[2] · Mayank Meghawat[2] · Soniya Lalwani[3] · Rajesh Kumar[4]

Received: 22 February 2021 / Accepted: 31 January 2022 / Published online: 30 April 2022
© King Fahd University of Petroleum & Minerals 2022

Abstract
COVID-19 has become a global disaster that has disturbed the socioeconomic fabric of the world. Efficient and cost-effective diagnosis methods are very much required for better treatment and eliminating false cases for COVID-19. COVID-19 disease is a type of respiratory syndrome, thus lung X-ray analysis has got the attention for an effective diagnosis. Hence, the proposed study introduces an Image processing based COVID-19 detection model C-COVIDNet, which is trained on a dataset of chest X-ray images belonging to three categories: COVID-19, Pneumonia, and Normal person. Image preprocessing pipeline is used for extracting the region of interest (ROI), so that the required features may be present in the input. This lightweight convolution neural network (CNN) based approach has achieved an accuracy of 97.5% and an F1-score of 97.91%. Model input images are generated in batches using a custom data generator. The performance of C-COVIDNet has outperformed the state-of-the-art. The promising results will surely help in accelerating the development of deep learning-based COVID-19 diagnosis tools using radiography.

Keywords COVID-19 detection · Deep learning · Convolution neural network · Image processing

1 Introduction

Coronavirus pandemic commonly known as COVID-19 has challenged and changed the socioeconomic world order. After witnessing great advancements in technology, nobody had expected that human beings would experience such a loss of health and wealth. Due to no specific treatment available, authorities are only relying on behavioral treatments to face such pandemics. Most of the countries have experienced a complete shutdown to break the chain of virus spread. World Health Organization (WHO) [1] has released guidelines on personal and social hygiene, and to exercise social distancing. Rapid testing and surveys are being conducted to reduce the spread of the virus and to deliver timely medical services. Due to the high cost of medical services, especially low and middle income countries are struggling to stop this pandemic. In such a scenario, early detection and efficient deployment of technology can save cost and time by eliminating false cases. COVID-19 is a type of respiratory syndrome which is quite similar to Pneumonia. Basic symptoms are dry cough, fatigue, shortness of breath, and mild fever. These properties make the identification of these diseases even more difficult. Radiographic images such as X-rays and CT scans of lungs are capable of providing enough information to detect the infection of Coronavirus. X-ray imaging is a cost-effective way to diagnose lung infections. Several institutes and researchers are collecting the dataset of X-ray images of COVID patients and are releasing them in the public domain. Advancement in deep learning technologies is capable of performing the image classification task

✉ Soniya Lalwani
slalwani.pdf@rtu.ac.in

Neha Rajawat
nrhada19@gmail.com

Bharat Singh Hada
bharat.aryawart@gmail.com

Mayank Meghawat
mayank.meghawat@gmail.com

Rajesh Kumar
rkumar.ee@mnit.ac.in

[1] Department of Mathematics, Career Point University, Kota, India

[2] Samsung R & D Institute, Noida, India

[3] Department of Mathematics, Bal Krisha Institute of Technology, Kota, India

[4] Department of Electrical Engineering, Malaviya National Institute of Technology, Jaipur, India

very efficiently and the research community has used deep learning algorithms to develop methods and models to detect COVID-19 infection using X-rays imaging.

ReCoNet [2] is a COVID detection CNN model based on residual images. The authors have trained the model on two datasets: CheXpert and COVIDx to avoid the overfitting of the model and tested on COVIDx. They have applied a multilayer CNN approach for preprocessing, feature extraction, and classification tasks. Jaiswal et al. [3] introduces a Coronavirus detection model trained on chest X-rays and CT scan images. The model called COVIDPEN is an example of a transfer learning method that used Pruned Efficient Net, and the model is tested on both CT scans and radiographic images. In addition to the classification task, authors have proposed a method for interpretation of prediction using the local interpretable model-agnostic explanation which is an image segmentation technique to represent the local and regional features. Wang L and Wong A [4] have come up with a deep CNN-based model COVID-Net which is also based on the COVIDx dataset. COVID-Net is quite a heavy model with 11.75 million trainable parameters which is trained on a combination of five different datasets for three-class classification of COVID, non-COVID, and pneumonia patients. ImageNet-based transfer learning approach [5] used in the DeTraC CNN model for differentiating COVID-19 from CXR images of normal and SARS (Severe Acute Respiratory Syndrome) patients. Pre-trained ImageNet weights are used for extracting features and classification purposes. High-dimensional feature space is reduced using principal component analysis (PCA) method. Zhang et al. [6] have used chest CT scan images to detect the presence of COVID-19 infection using five-layered deep-CNN architecture which uses stochastic pooling layers. Convolution layers are combined with normalization layers, and dropout layers are combined with fully connected layers. This algorithm has achieved a recall of 93.28%±1.50%, and an accuracy of 93.64%±1.42%, in classifying COVID-19 from normal persons. The limitation of this work is that it has done only binary classification of COVID-19 disease and does not cover the pneumonia infection. Another limitation is the size of the dataset which consists of a total of 640 images of both COVID-19 and healthy individuals, respectively. In the continuation of this work, the authors have proposed a seven-layered deep-CNN architecture [7] with the same stochastic pooling concepts for binary COVID-19 diagnosis. The authors have used the same dataset of 640 images of COVID-19 and healthy patients, respectively, again but to increase the sample space, they have applied data augmentation. The tenfold cross-validation technique is used to train the model and achieve a recall of 94.44±0.73 and an accuracy of 94.03±0.80. It is clear that the approach is different this time but the results do not show any significant improvement and the limitations stated for previous work are still valid. Albert N [8] has developed a CNN model to solve the problem of COVID-19 detection using the chest radiography data. This research has used CoronaHack-Chest X-Ray [9] dataset available on the Kaggle platform. Data augmentation techniques are applied to increase the amount of COVID images. The author has performed a set of experiments with DenseNet-121, ResNet-50, and EfficientNet with a different set of hyperparameters. All the approaches that have been discussed till now include image datasets that highlight the importance of image processing. The next paragraph discusses some prior arts which will be covering some important aspects of image processing and its role in improving the model performance.

Multimedia content can store the information more precisely and information can be stored more effectively with many hidden and rich features. Considering the challenge to decode and retrieve the information, Wang et al. [10] have designed a novel image retrieval system using the visual saliency features present in the dataset. Image features such as direction, intensity, and distribution of colors are used to generate an overall saliency map. To clearly explain the pattern of the images, multi-feature fusion techniques are used and image complexities are defined using Cognitive load complexity and Cognitive level complexity. The proposed image retrieval system is designed using logistic regression, tested on different datasets and also compared with state-of-the-art models. Image segmentation is a widely used method for information extraction from the non-useful background and 2D segmentation models have been widely used methods. As an extension, AlZu'bi et al. [11] has proposed a modification of 2D Fuzzy C-Means (FCM) to segment 3D medical image volumes. It is observed that efficiency is the main challenge while dealing with the 3D data; thus to overcome this challenge, the authors have implemented the proposed model on Graphical Processing Unit (GPU) in parallel to CPU. The authors have claimed that parallel implementation is 5 times faster than the sequential operation of the same combination. As another example of information extraction using the segmentation process, watermark detection and extraction using CNN proposed by Li et al. [12] can be discussed. In the preprocessing step, the algorithm takes a grayscale image as input and using discrete cosine transform (DCT) block embeds the respective watermarking in the input. Certain watermark data is used while training the model which helps the system to effectively detect and extract the watermark signal. Lv et al. [13] present a novel application of CNN for Relation extraction which is one of the most important tasks in Natural Language Processing (NLP). Authors have utilized both CNN and directional self-attention network (DiSAN) model where pooling layers of CNN are modified to reduce the dimensions of the feature vector. The proposed DiSAN-2CNN model encodes input samples on both feature level and information which is time-dependent. Cervical detection and image segmentation using

TCT images was done by Long et al. [14], keeping Mask R-CNN as basic building block. To find the best feature set, neural network's feature space was searched using NAS-FPN framework. After generating different feature maps, overlapping was performed to avoid the missed identification and also to improve the performance of higher dimension using lower feature space. A software named Labelme was used to annotate and segment into the elliptical cells, leukocytes, and fungi. The number of data samples augmented using different combination of image rotation, scaling, offset and transformations. It is observed that the algorithmic combination of Mask R-CNN+NAS-FPN+Soft NMS has achieved the highest recall of 72.9% that does not seems to be great for a medical application. It provides a further scope of improvement in model performance by means of more data samples or improved model structure. Remote sensing is an application that highly relies on a quality image data and an efficient object detection model. Li et al. [15] proposed a multiscale fully convolutional neural network (MFCN) to extract the information about ground objects in remote sensing applications. Lack of data samples could be a problem to identify an object correctly. To avoid this issue weighted binary cross-entropy (WBCE) approach is used to minimize the loss of training phase and dice coefficient is utilized to handle the problem of data unbalancing. This work had achieved an optimized accuracy of 93.5% and F1-score of 91.1%. Definitely the working performance of this approach can be improved by supplying more and better image samples. After discussing the prior arts, the limitations and possible shortcomings can be summarized as follows and the countermeasure are also provided inline:

- Most of the models described in the literature review are built using transfer learning approaches which increase the model complexity. The proposed model is modeled in-house with just 655 thousand trainable parameters and yet able to outperform the existing works.
- Data sample used in the prior arts contain fewer images which presents the problem of overfitting. To avoid this problem in the proposed work, model performance is analyzed for both validation and test data set.
- In the results section, Table 7 represents the model complexities in terms of trainable parameters. Complex and heavier models increase the training time and need more resources to train and deploy on real devices.

This paper proposes an efficient and lightweight CNN-based deep learning model C-COVIDNet, capable of detecting COVID-19 infection with high accuracy. The model contains a total of eight layers, consisting of five convolutions and three fully connected layers. C-COVIDNet is trained on a combined dataset of images retrieved from the online repository Kaggle [16–18]. The dataset contains X-ray images belonging to three categories: COVID-19, Viral Pneumonia, and Normal or Healthy people. X-ray images are preprocessed through an image processing pipeline which is capable of segmenting the ROI (Lungs in this case) from the non-useful background part of the X-ray. The proposed model accepts normalized two-dimensional input composed of the original image and ROI. In addition to COVID-19 detection, C-COVIDNet is also able to distinguish between COVID and Pneumonia patients. The model is tested on the benchmarking datasets used in other studies to prove a generalized accuracy and experiment results are presented in the coming sections. The contributions of this work can be highlighted as follows:

- CNN-based detection model to categorize input X-Ray images into COVID-19, pneumonia, and healthy individuals.
- An efficiently designed image processing pipeline that highlights useful information by extracting the ROI mask.
- A custom image data generator algorithm that can provide the preprocessed data to training module in batches to speed up the process.

The rest of this paper is organized as follows: Section 2 explains the technical details of the methodology used in this work; experiments and results are discussed in Sect. 3. The key finding, observation, and comparison of the model with existing prior arts are also done in Sect. 3. Model performance along with comparative analysis are presented in Sect. 4, and finally, the paper is concluded in Sect. 5 with future remarks.

2 Material and Methods

2.1 Dataset

In the proposed work, a CNN-based multi-classification model is trained using a combined data set of image samples belonging to healthy individuals, COVID-19 patients, and Pneumonia patients. Detailed information about datasets and their properties are explained in Table 1. Though the number of samples of COVID-19 is lower compared to the other two categories yet the proposed model can distinguish well-enough among all three categories.

2.2 Data Preprocessing

The recent advancement in CNN models proved to be quite robust in identifying hidden features of image data but the information provided by raw data can lead up to a baseline accuracy. To improve further, efficient data preprocessing methods are needed to feed the interesting part of an image and direct the model to ignore the noise. This interesting part

Table 1 Overview of datasets used in this paper

Dataset reference(↓)/samples(→)	COVID	Pneumonia	Normal	Total
COVID-19 Radiography database [16]	219	1345	1341	2905
COVID-19 Detection X-Ray dataset [17]	69	617	1330	2006
COVID-19 (COVID-19 & pneumonia) [18]	576	4273	1583	6432

Algorithm 1 Image Preprocessing Pseudo Code

```
Input: Grayscale Image of Chest X-Ray
Output: Combined image which has ROI as 2^nd channel.
Output image shape = 128, 128, 2

START
$removeBackgroundNoise(image)$:
  • $temp\_img \leftarrow image.copy()$
  • $n\_components, output, stats, centroids \leftarrow cv.connectedComponentsWithStats(temp\_img, connectivity = 4)$
  • $component\_size\_list \leftarrow stats[: - -1]$
  • FOR $(i = 0\ to\ n\ components)$ :
      – IF $component\_size\_list[i] < 500$ :
          • $image[output == i] \leftarrow 0$
  • RETURN $image$
$processImage(imagepath)$:
  • $img \leftarrow cv.imread(imagepath)$
  • $gray\_img \leftarrow cv.cvtColor(img, cv.COLOR\_BGR2GRAY)$
  • $threshold \leftarrow cv.threshold(gray\_img, 127, 255, cv.THRESH\_BINARY\_INV + cv.THRESH\_OTSU)$
  • $ROI \leftarrow gray\_img\&threshold$
  • $ROI \leftarrow cv.GaussianBlur(gray\_img, (9, 9), 0)$
  • $ROI \leftarrow cv.equalizeHist(gray\_img)$
  • $ROI \leftarrow removeBackgroundNoise(gray\_img)$
  • $model\_input \leftarrow merge(img, ROI)$
  • $model\_input \leftarrow 255 - model\_input$
  • $model\_input \leftarrow cv.resize(model\_input, 128, 128)$
  • RETURN $model\_input$
STOP
```

of an image is also known as ROI [19] which contains the most amount of information needed for the point classification. In this work, image processing has been used for finding ROI. The details are given by pseudo-codes in algorithm 2.2.

2.2.1 Region of Interest (ROI)

It is a sub-region of an image that is capable of providing the exact and the most amount of information to train a CNN model efficiently. Lungs are examples of ROI in chest X-ray images. In this work, lungs are segmented from the background noise image processing to feed only the required information to the CNN model.

2.2.2 Model Input Preparation Using ROI

The preprocessing pipeline shown in Fig. 1 explains the step involved in preparing the model input. The original X-ray image consists of RGB channels. However, to extract ROI, the grayscale image has provided excellent results. The final model input contains ROI as a second channel attached to the original grayscale image. In this way, features associated with ROI will go as additional information along with the original features.

- Inverted Binary Threshold [20]: In this approach, each pixel value is compared against a predefined threshold value. If the pixel value is greater than the threshold, then set to zero; otherwise, assign maximum intensity value, i.e. 255. Thus, the input grayscale image is converted to binary format.
- Otsu's threshold method: Thresholding is a method of pixel-level manipulation in images. Each pixel value is compared with a threshold value and accordingly set to a new value. Otsu threshold does not require a predefined threshold value but is chosen dynamically [28]. In the current problem, we are specifically dealing with binary images so, Otsu's method analyzes the histogram generated from image pixel intensities and chooses a threshold that efficiently bifurcates the two peaks. The resultant threshold value (t) should minimize the weighted within-class variance $\sigma_w^2(t)$ is defined by equation 1.

$$\sigma_w^2(t) = w_1(t)\sigma_1^2(t) + w_2(t)\sigma_2^2(t) \tag{1}$$

where $w_1(t)$ and $w_2(t)$ are class weights for both intensity classes in a binary image, respectively, and $\sigma_1^2(t)$ and $\sigma_2^2(t)$ are weighted variance values for respective classes calculated as follows: class weights:

$$w_1(t) = \sum_{i=1}^{t} P(i) \ \&\ w_2(t) = \sum_{i=t+1}^{T} P(i) \tag{2}$$

The quantity of the pixels with a specified gray-level is denoted by i. The general number of pixels in the image is n. Thus, the probability of gray-level i occurrence is:

$$P(i) = \frac{n_i}{n} \tag{3}$$

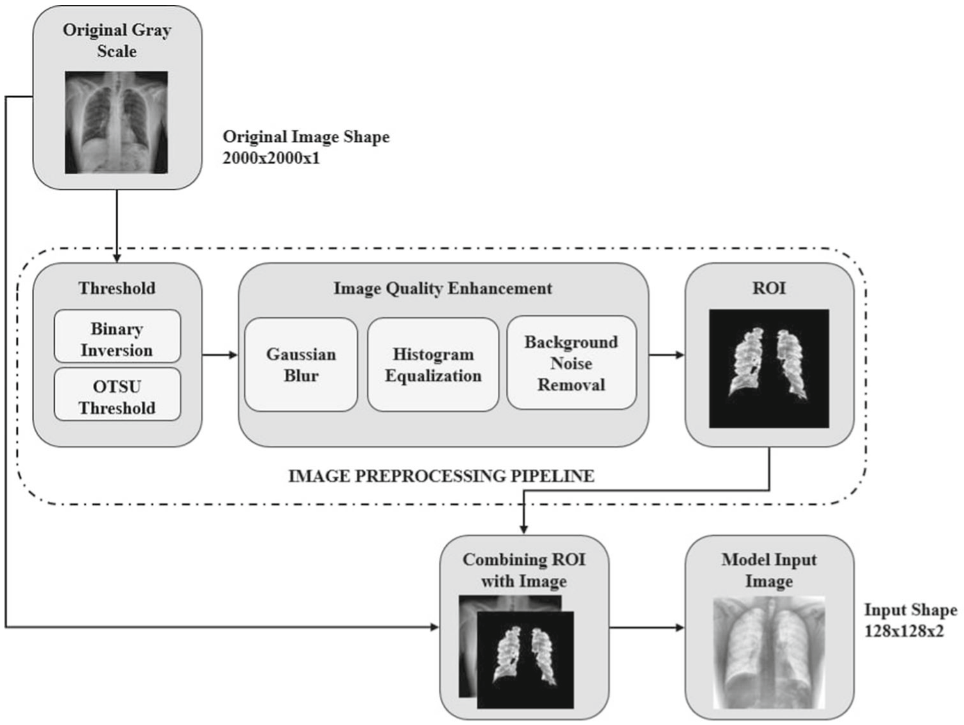

Fig. 1 Image preprocessing pipeline for feature engineering

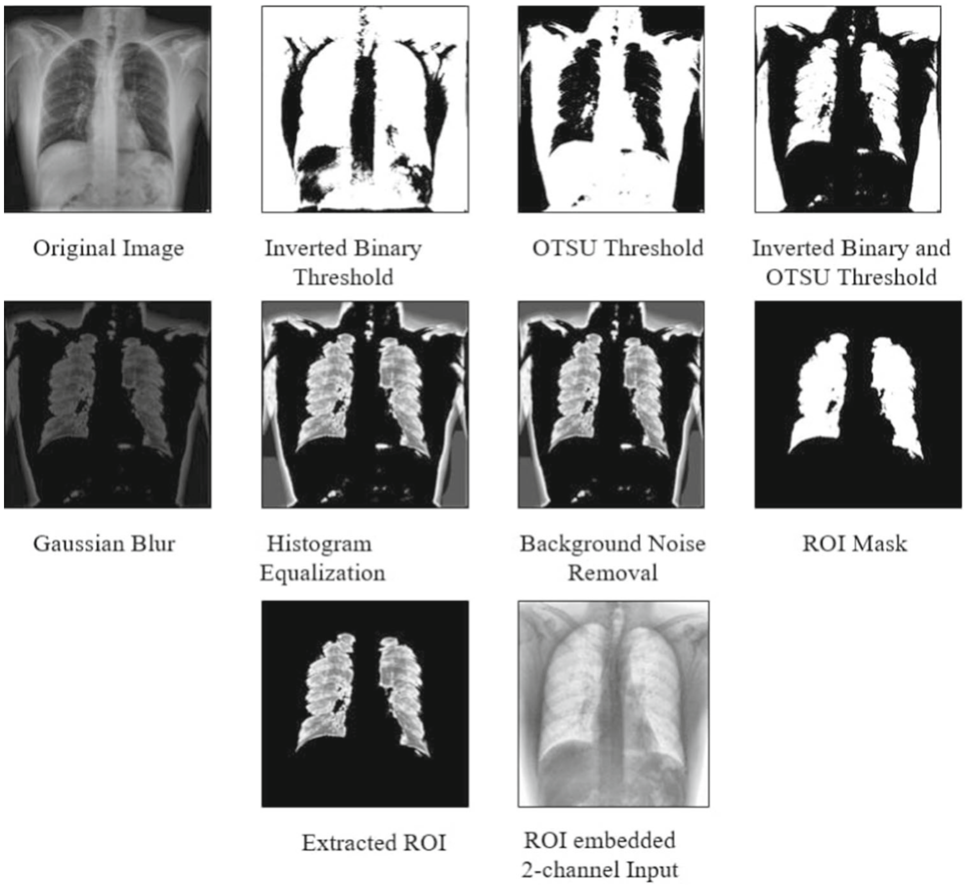

Fig. 2 Illustrative images generated at intermediate steps of image preprocessing

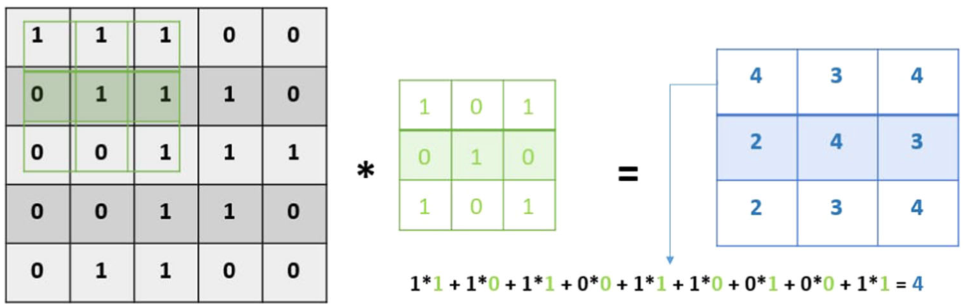

Fig. 3 Example explaining the kernel trick used in CNN models

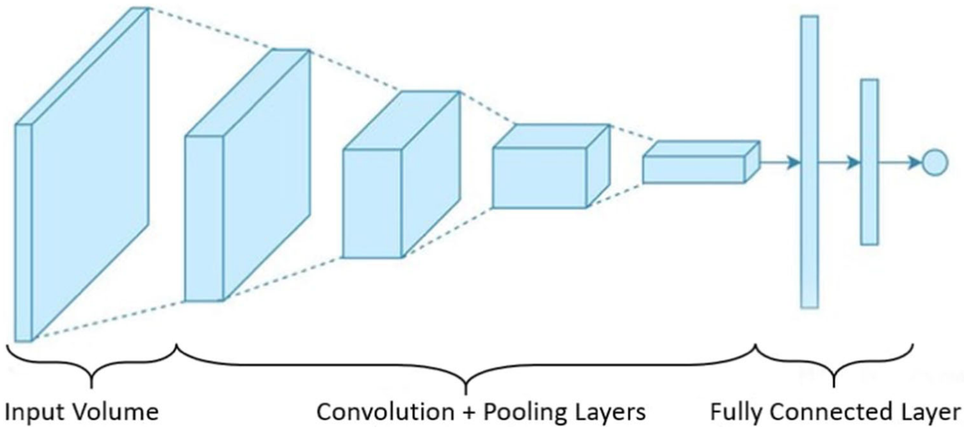

Fig. 4 An illustration for basic CNN architecture

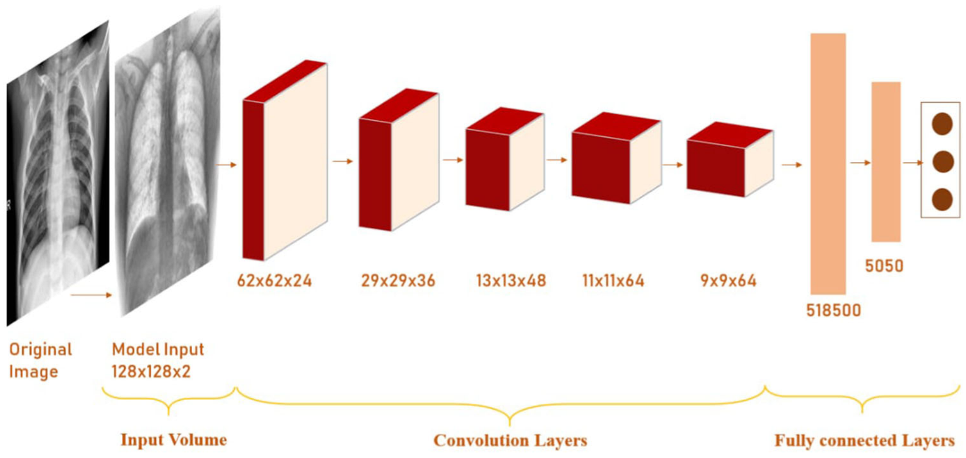

Fig. 5 C-COVIDNet

weighted class means:

$$\mu_1(t) = \sum_{i=1}^{t} \frac{iP(i)}{w_1(t)} \ \& \ \mu_2(t) = \sum_{i=t+1}^{T} \frac{iP(i)}{w_2(t)} \tag{4}$$

weighted class variance:

$$\sigma_1(t) = \sum_{i=1}^{t} [i - \mu_1(t)]^2 \frac{P(i)}{w_1(t)} \ \& \ \sigma_2(t) = \sum_{i=t+1}^{T} [i - \mu_2(t)]^2 \frac{P(i)}{w_2(t)} \tag{5}$$

- Gaussian blur: The blurring of an image is used for reducing the noise. Applying Gaussian blur [21] is the same as the convolution of an image with Gaussian function. It reduces the high-frequency components of an image and typically works as a low-pass filter. Gaussian function in one dimension is

$$G(x) = \frac{1}{\sqrt{2n\sigma^2}} e^{-\frac{x^2}{2\sigma^2}} \tag{6}$$

Gaussian function in two dimensions is:

$$G(x, y) = \frac{1}{\sqrt{2n\sigma^2}} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{7}$$

where x and y are the horizontal and vertical distances, respectively, and σ is the standard deviation.

- Histogram equalization: A graph, representing the pixel intensity distribution of an image, is known as a histogram. It is a technique used for enhancing the contrast of an image. To do so, most frequent intensity values will be stretched out of their original boundaries. When useful information is represented by these most frequent local intensities, then histogram equalization [22] tries to highlight this information more. Let the input image consists of k different intensity levels; the histogram equalization transformation function $T(r)$ is defined as equation 7:

$$T(r_k) = \sum_{j=0}^{k} p_r(r_k) = \sum_{j=0}^{k} \frac{n_j}{n} \quad 0 \leq r_k \leq 1 \\ and\ k = 0, 1, ..., L-1 \tag{8}$$

where $p_r(r_k) = \frac{n_j}{n}$ = probability density function (pdf) of input image for level j n = total number of pixels in the image n_j = number of pixel for level j

- Background noise removal: The purpose of this step is to remove insignificant intensity values. These insignificant intensities are nothing but noise. The images are segmented into connected components. If the size of any connected component contains less than 500 pixels, then it is removed by setting the value to zero.

2.3 Illustrative Images Generated from Image Preprocessing

Images in Fig. 2 represent the output generated at different stages of the image preprocessing pipeline. It is clear after observing the images that the ROI mask is certainly helping in enhancing the model input. ROI extraction also worked as a filter to remove insignificant information present in the background.

2.4 CNN and the C-COVIDNet

Deep learning algorithms have been used in large amounts to solve the problems of the computer vision domain. Deep layered architecture is capable of identifying high-level features and hidden information from multidimensional image input. CNN models are specially prepared to solve the computer vision tasks with any need for handcrafted features and minimal data preprocessing. CNN models are designed to learn from low-level to high-level features using the nonlinear activations of previous layers. CNN takes advantage of features specific kernels or filters which sense the presence of features in a smaller segment of the input image. As the layers of CNN grow deeper, it starts to learn the global structures and shapes over features extracted in previous layers.

2.4.1 Why Do CNN Models are Preferred for Image Analysis?

CNN is capable of capturing the spatial and temporal variation in an image by applying relevant filters or kernels. These filters are matrices that learn a multidimensional space instead of a single-pixel value and the filter does not change in the entire course of learning. This approach reduces the learnable parameters with a massive scale. If an image is fed to an artificial neural network (ANN) model, then it will try to learn the information provided by individual pixels due to the absence of kernel trick. In such a case, learnable parameters will be unmanageable for larger image sizes. Figure 3 can be referred to check how kernel or filter work in CNN.

2.5 Basic CNN Architecture

CNN model consists of three building blocks as shown in Fig. 4:

- Convolution Layer: These layers are responsible for learning the high-level features of an input image. There can be multiple convolution layers that can learn the image features in an incremental way.
- Pooling Layer: It reduces the spatial size of convolved volume thus decreasing the computational power needed to process the data. It is also useful to extract the dominant features. This layer also performs noise suppression.
- Fully Connected or Dense Layer: Once the size of the input volume is efficiently reduced by pooling layers, fully connected layers are used for learning complex and nonlinear combinations of high-level features. Fully connected layers typically work as an ANN model which is fed a one-dimensional vector formed from the output of the pooling layer.

Conventionally, the pair of convolution and pooling layers are considered as a single layer of CNN and there can be any number of pairs of such layers.

2.6 C-COVIDNet

The motivation behind developing the C-COVIDNet is to develop a lightweight CNN-based model which can compete with the heavy transfer learning-based approach. Therefore, data preprocessing pipeline is designed to supply refined and exact information as lung segments using ROI. To keep the trainable parameters as minimum as possible, the model does not use any pooling layer, instead, dropout regularization is done at every layer for better generalization. This design is inspired by a CNN model used for developing self-driving cars [23]. Figure 5 explains the layered arrangement of the proposed model.

2.6.1 Model Architecture

The proposed model is an eight-layered CNN architecture that consists of five convolution layers and three dense layers where the last layer represents the output layer. Model input has a size of 128x128x2 in which the ROI image is stacked upon the original grayscale input. The first three convolution layers use kernels of shape 5x5 with a common stride of shape 2x2, and the number of kernels in these three layers is 24, 36, and 48, respectively. Rest two convolution layers use 3x3 kernels without any stride and the number of kernels in both layers is 64. Each of these convolution layers is activated using rectified linear unit (RELU) activation function, and the dropout function is associated with eliminating the risk of overfitting. Dropout is a regularization method in which a fixed number of layer nodes are randomly ignored or dropped out during the training process. This way, training hyperparameters are computed and adjusted using many different architectures in parallel with a different number of nodes and permutations. This random selection is done based on the probabilistic selection criteria passed in the dropout function. In the proposed model, the first two convolutions use the dropout rate as 0.2, and the rest three convolutions use 0.1 which means during any training instance, 20% and 10% nodes will be dropped out, respectively. After the convolution operations, model starts using dense layers connection of three layers consisting of 100, 50, and 3 output nodes, respectively. Except for the output layer which uses the SOFTMAX activation function, the other two dense layers use RELU activation only. The first two dense layers are also using a dropout mechanism with dropout rates of 0.5 and 0.3 respectively. Trainable parameters are only 655 thousand which indicates that the model is quite lightweight. Table 2 provides complete details of the layer schema used in C-COVIDNet.

2.6.2 Feature Normalization

Normalizing the value of the feature is important for assigning relative importance for image pixels. It helps machine learning algorithms to minimize the bias toward any specific type of data sample. Normalization also ensures a smooth convergence of cost function [24]. In the proposed model, input volume is normalized using the min-max scaling approach. Normalized pixel value belongs to a range of [– 0.5, 0.5]. If x is the pixel value, then normalization is to be as:

$$x_{norm} = \frac{x}{255} - 0.5 \tag{9}$$

2.6.3 Image Data Generation in Batches

Image data handling is a time-consuming task that increases the preprocessing and training time considerably. To solve this problem, a custom image data generator function is used with which preform the data preprocessing task in batches and feed that batch to the model.

Before starting every epoch, a random batch of images would be selected and only that amount of data will be handled. In this way, the time required to handle a single large chunk of the dataset is eliminated. Using this method entire pipeline is completed in just 6 hours on a resource-constrained system. Model training pipeline and image data generator are explained in pseudo-codes in Algorithm 2.6.3 and 3:

Algorithm 2 Pseudo-Code for Image Data Generator

```
START
$imageDataGenerator(features, labels, batch\_size)$:
  • $batch\_features \leftarrow init\_zeroes(128, 128, 2)$
  • $batch\_labels \leftarrow init\_zeroes(batch\_size)$
  • FOR ($i = 0$ to $batch\_size$) :
      – $index \leftarrow generate\_random\_number(0, length(features))$
      – $processed\_image \leftarrow preprocessImage(features[index])$
      – $processed\_image \leftarrow processed\_image.reshape(128, 128, 2)$
      – $batch\_features[i] \leftarrow processed\_image$
      – $batch\_labels[i] \leftarrow labels[index]$
  • RETURN $batch\_features,\ batch\_labels$
STOP
```

Algorithm 3 Pseudo-Code explaining usage of Image data generator and feature normalization

```
Precondition:
D: X-ray image dataset.
$X_{train},\ y_{train},\ X_{test},\ y_{test}$: Train and testing split of dataset D
START
$imageDataGenerator(features, labels, batch\_size)$:
  • $batch\_features, batch\_labels \leftarrow imageDataGenerator(X_{train}, y_{train}, batch\_size)$
  • $normalized\_features = normalizeInputData(batch\_features)$
  • $fit\_model(normalized\_features)$
  • $y_{pred} = predict(y_{train})$
  • RETURN $y_{pred}$
STOP
```

Python source code and related code files for the proposed model are shared on GitHub repository: https://github.com/Macky1290/sCovidNet

Table 2 C-COVIDNet architecture overview

No.	Layer type	Number of Kernels/probability	Kernel size	Activation function	Input size	Output size	Parameters (incl. bias term)
1	Convolution2D	24	5 × 5	RELU	128 × 128 × 2	62 × 62 × 24	2424
	Dropout	0.2	–	–	62 × 62 × 24	62 × 62 × 24	0
2	Convolution2D	36	5 × 5	RELU	62 × 62 × 24	29 × 29 × 36	21,636
	Dropout	0.2	–	–	29 × 29 × 36	29 × 29 × 36	0
3	Convolution2D	48	5 × 5	RELU	29 × 29 × 36	13 × 13 × 48	43,248
	Dropout	0.1	–	–	13 × 13 × 48	13 × 13 × 48	0
4	Convolution2D	64	3	RELU	13 × 13 × 48	11 × 11 × 64	27,712
	Dropout	0.1	–	–	1 × 11 × 64	1 × 11 × 64	0
5	Convolution2D	64	3 × 3	RELU	1 × 11 × 64	9 × 9 × 64	36,928
	Dropout	0.1	–	–	9 × 9 × 64	9 × 9 × 64	0
	Flatten	Number of nodes after flattening = 5184					0
6	Dense	100	–	RELU	5184	100	518,500
	Dropout	0.5	–	–	100	100	0
7	Dense	50	–	RELU	100	50	5050
	Dropout	0.3	–	–	50	50	0
8	Dense	3	–	SOFTMAX	50	3	153

Table 3 Confusion matrix derived over validation and testing data

Class labels: Predicted (↓) / Actual (→)	COVID	Pneumonia	Normal
COVID	428	2	7
Pneumonia	5	1242	37
Normal	15	117	1888

Table 5 Confusion matrix for ablation experiments

Class labels Predicted (↓) / Actual (→)	COVID	Pneumonia	Normal
COVID	229	11	42
Pneumonia	21	614	208
Normal	22	73	820

Table 4 Evaluation metrics of C-COVIDNet

Dataset type	No. of images	Accuracy	F1-Score
Training	5364	97.5	97.91
Validation	1789	95.3	95.5
Testing	1952	94.8	94.9

3 Results

In the experimental setup, the model has used the adaptive moment estimation (Adam) optimization approach. Adam optimization uses adaptive learning rate and a set of hyper-parameters for better estimation of gradients with minimal tuning [25]. This optimizer is well-suited where the dataset is large enough (e.g., natural language processing, computer vision, etc.). The performance criteria for the model are to maximize the F1-score value for robustness and accuracy for

Table 6 Model validation against other datasets

Dataset	Number of images	Accuracy (%)	F1-score (%)	Precision (%)	Recall (%)
CoronaHack	5933	97.7	97.0	97.7	97.7
Images 10,00	98	98.9	98.7	98.9	98.9

Table 7 Comparison of C-COVIDNet with prior studies

Author	Model	Trainable parameters **(million)**	Accuracy (%)	F1-score (%)	Precision (%)	Recall (%)
Ahmed et al. [2]	ReCoNet	2.5	97.4	–	–	96.3
Amit et al. [3]	COVIDPEN	5.3	96	94	92	96
Wang et al. [4]	COVID-Net	11.75	93.3	–	96.4	87.1
Nikita [8]	DenseNet-121	8.6	85	90	92	88
Zhang et al. [6]	5L-CNN-CD	1.8	93.6	–	–	93.2
Zhang et al. [7]	7L-CNN-CD	2.5	94.03	–	–	94.44
Abbas et al. [5]	DeTraC	60	93.1	–	–	–
Proposed model	C-COVIDNet	0.655	97.5	95.1	95.1	96.2

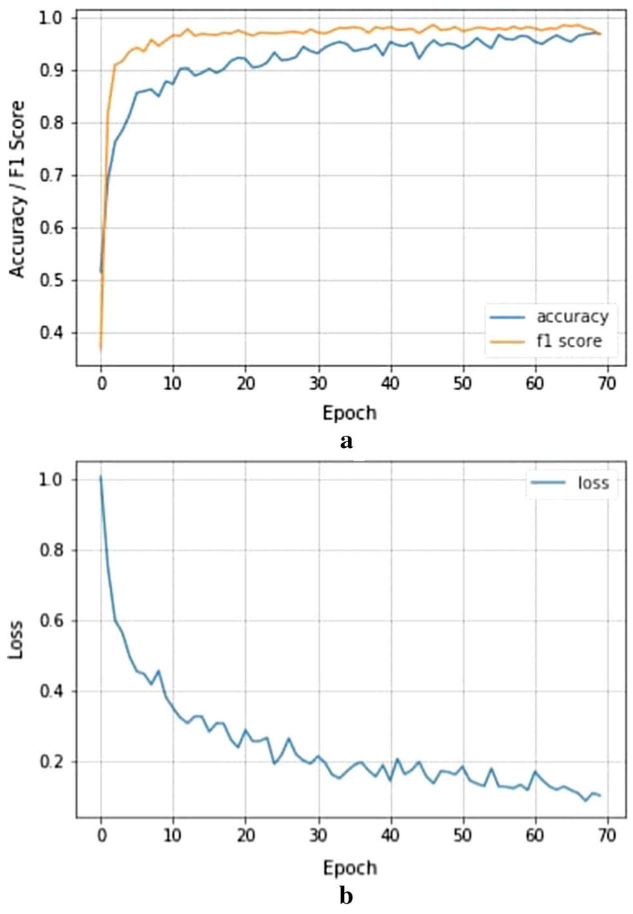

Fig. 6 Accuracy and F1-score **a** convergence, **b** loss computation

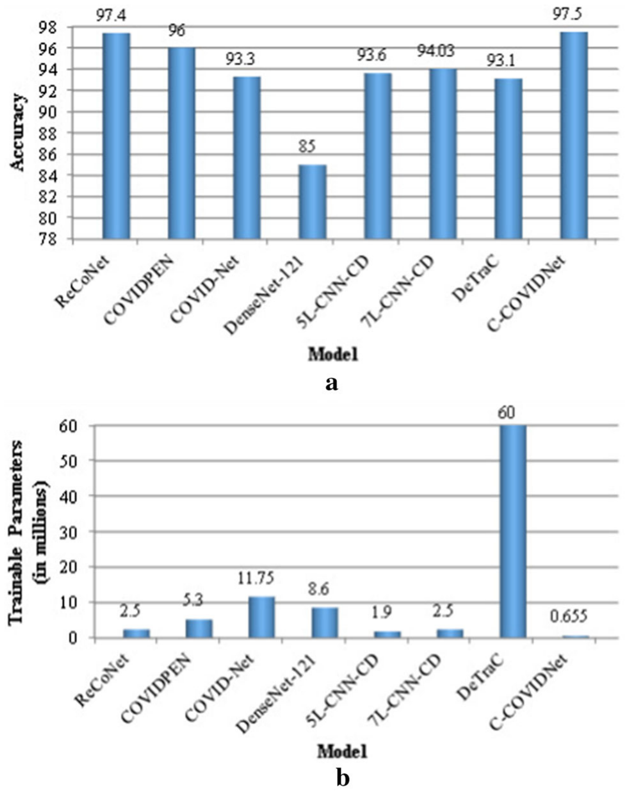

Fig. 7 Comparison charts for the proposed model and existing prior arts **a** performance comparison in terms of accuracy **b** complexity compared in terms of trainable parameters

effectiveness without overfitting. Therefore, validation accuracy is compared with testing accuracy.

- Confusion Matrix: It is a tabular representation of model performance that describes where exactly the model fails to correctly predict the validation instances. The numbers of true and false predictions are summarized with respective values and broken down by each outcome label. Table 3 explains the confusion matrix for the proposed model.
- Accuracy: Accuracy is the most straightforward performance metric and is represented as a ratio of correctly predicted samples to the total samples. Accuracy is a simple yet basic measurement of model performance so one should look into other metrics also to check the effectiveness of the model.

$$\text{Accuracy} = \frac{(\text{TP} + \text{TN})}{(\text{TP} + \text{TN} + \text{FP} + \text{FN})} \tag{10}$$

where TP, TN, FP, and FN refer True Positive, True Negative, False Positive, and False Negative.

- Precision: Precision is the ratio of correct positive prediction out of all positive predictions made by the model.

$$\text{Precision} = \frac{\text{TP}}{(\text{TP} + \text{FP})} \tag{11}$$

- Recall (Sensitivity): Recall is a metric that quantifies the number of correct positive predictions made out of all positive predictions that could have been made.

$$\text{Recall} = \frac{\text{TP}}{(\text{TP} + \text{FN})} \tag{12}$$

- F1-score: This parameter tries to find out a balanced sweet spot between precision and recall. It tries to maximize the robustness (it does not miss a significant number of instances) and exactness (how many instances it classifies correctly) of the model. It is the harmonic mean of precision and recall.

$$\text{F1} = \frac{2(\text{Precision} \times \text{Recall})}{(\text{Precision} + \text{Recall})} \tag{13}$$

Table 4 represents the size of training datasets, testing datasets, and model performance, respectively. Values of accuracy and F1-score are well-balanced for both the training and testing phase. Such models perform quite good when deployed in a real-life example.

The proposed model is trained in such a way that it tried to maximize both accuracy and F1-score value at the same time. This approach ensured the robustness of the model and generalize well with all three differentiable classes. Figure 6 provides a clear indication that the model performed well. The model is trained for 70 epochs, where it achieved a saturation point. Training loss after 70 epochs is less than 0.1. Even though accuracy attained stabilization after 15 epochs, F1-score kept on increasing and loss is approaching zero.

3.1 Ablation Results Without Image Preprocessing

The results described in previous paragraphs represent the performance of the end-to-end model pipeline. To judge the effectiveness of the model in isolation, experiments are performed with applying any image preprocessing on raw data samples. For this purpose, COVID-19 (COVID-19 & Pneumonia) dataset is utilized which contains 6432 image samples in total, and train and test data split is done in 80:20 ratio, respectively. Model training is done on raw RGB images, and only the input images are resized as the expected input size of the proposed model. The optimal testing accuracy of the model is 87% on the test set and 89% on the validation set. The ablation results for the experiment done without image preprocessing are shown in Table 5. One can observe that without preprocessing also the proposed model has worked quite well, and the image processing pipeline has also helped to improve the performance. If we compare the confusion matrices of with and without data preprocessing, then it is quite evident that ablation modeling seems to more confused, and mis-classifications need to be removed for effectiveness.

3.2 Model Validation Against Other Datasets

To show that the proposed C-COVIDNet model has a wide range of acceptability and general accuracy across other chest X-ray datasets, the model was verified against two publicly available datasets CoronaHack-Chest X-Ray Dataset [26] and COVID-19 Patients Lungs X-Ray Images 10000 [27]. Sample images for these datasets were not included in the training phase so that model can depict its effectiveness against unknown data. Testing accuracy for these CoronaHack and Images 10000 datasets are 97.7% and 98.9%, respectively, which is quite similar as obtained during training of the model. These observations prove that model is working equally well across various X-ray datasets and also provide benchmarking for the proposed methodology. Table 6 explains the overall results for both of these datasets along with the size of the testing sample space. It can be observed that the proposed model has remarkable accuracy.

4 Discussion

C-COVIDNet has outperformed state-of-the-art works. The approach used is lightweight to implement for any real-life example. The model has achieved the highest value for accuracy and F1-score compared to the prior studies, whereas precision and recall values are almost comparable. Most of the studies have not provided information related to all the four performance parameters, but the proposed work has judged the model against each of them. Table 7 compares the performance of the proposed model with recently published studies in this domain. The detailed comparisons are represented using bar charts in Fig. 7 where performance and complexity comparison is done. It is quite clear that the proposed C-COVIDNet model is less complex in terms of trainable parameters but the accuracy is superior.

5 Conclusion and Future Remarks

COVID-19 pandemic has affected the social, physical, and mental well-being of human life across the world. Slight negligence can cause a loss of life; therefore, timely and cost-effective cure is the need of the hour. This paper has proposed an efficient and lightweight COVID-19 detection

CNN model named as C-COVIDNet, which is inspired from a CNN model, used for developing self-driving cars. The model is trained on a combined dataset derived from three publicly available datasets COVID-19 Radiography Database, COVID-19 Detection X-Ray Dataset, and COVID-19 (Covid-19 & Pneumonia). An efficient image preprocessing pipeline is designed to supply refined and exact information as lung segments by extracting ROI. The image preprocessing module contains inverted binary thresholding, Otsu's thresholding, Gaussian blurring, histogram equalization and background noise removal. The proposed model consists of eight layers including five convolutions layers and three dense layers where total trainable parameters are only 655 thousand. In the experimental setup, the model has used the adaptive moment estimation (Adam) optimization approach and trained for around 70 epochs. The optimal training accuracy and F1-score achieved are 97.5% and 97.91%. Model is tested against two publicly available datasets CoronaHack-Chest X-Ray Dataset and COVID-19 Patients Lungs X-Ray Images 10000 where testing accuracy was 97.7% and 98.9%, respectively. Dataset used in this research felt the requirement of quality data, specifically for COVID-19 patients. It highlights the demand for clean, balanced, and variety of data. The promising results in the proposed study will surely help in accelerating the development of deep learning-based COVID-19 diagnosis tools using radiography. Predicting a single wrong result may cause great loss for human life so future possibilities require a robust detection methodology that keeps sensitivity and specificity as high as possible.

References

1. Martelli-Júnior, H.; Machado, R.A.; Martelli, D.R.; Coletta, R.D.: Dental journals and coronavirus disease (COVID-19): a current view. Oral Oncol **106**, 104664 (2020)
2. Ahmed, S.; Yap, M.H.; Tan, M.; Hasan, M.K.: Reconet: multi-level preprocessing of chest X-rays for COVID-19 detection using convolutional neural networks. medRxiv (2020). https://doi.org/10.1101/2020.07.11.20149112
3. Jaiswal, A.K.; Tiwari, P.; Rathi, V.K., Qian, J.; Pandey, H.M.; Albuquerque, V.H.C.: Covidpen: a novel COVID-19 detection model using chest X-rays and CT scans. medrxiv (2020). https://doi.org/10.1101/2020.07.08.20149161
4. Wang, L.; Lin, Z.Q.; Wong, A.: Covid-net: A tailored deep convolutional neural network design for detection of COVID-19 cases from chest X-ray images. Sci. Rep. **10**(1), 1–12 (2020)
5. Abbas, A.; Abdelsamea, M.M.; Gaber, M.M.: Classification of COVID-19 in chest X-ray images using DeTraC deep convolutional neural network. Appl. Intell. **51**(2), 854–864 (2021)
6. Zhang, Y.D.; Satapathy, S.C.; Liu, S.; Li, G.R.: A five-layer deep convolutional neural network with stochastic pooling for chest CT-based COVID-19 diagnosis. Mach. Vis. Appl. **32**(1), 1–13 (2021)
7. Zhang, Y.D.; Satapathy, S.C.; Zhu, L.Y.; Górriz, J.M.; Wang, S.H.: A seven-layer convolutional neural network for chest CT based COVID-19 diagnosis using stochastic pooling. IEEE Sens. J. (2020)
8. Albert, N.: Evaluation of contemporary convolutional neural network architectures for detecting COVID-19 from chest radiographs (2020). arXiv preprint arXiv:2007.01108
9. Dataset.: CoronaHack-Chest X-Ray-Dataset (2020). https://www.kaggle.com/praveengovi/coronahack-chestxraydataset
10. Wang, H.; Li, Z.; Li, Y.; Gupta, B.B.; Choi, C.: Visual saliency guided complex image retrieval. Pattern Recogn. Lett. **130**, 64–72 (2020)
11. AlZu'bi, S.; Shehab, M.; Al-Ayyoub, M.; Jararweh, Y.; Gupta, B.: Parallel implementation for 3D medical volume fuzzy segmentation. Pattern Recogn. Lett. **130**, 312–318 (2020)
12. Li, D.; Deng, L.; Gupta, B.B.; Wang, H.; Choi, C.: A novel CNN based security guaranteed image watermarking generation scenario for smart city applications. Inf. Sci. **479**, 432–447 (2019)
13. Lv, X.; Hou, H.; You, X.; Zhang, X.; Han, J.: Distant Supervised Relation Extraction via DiSAN-2CNN on a Feature Level. Int. J. Seman. Web Inf. Syst. **16**(2), 1–17 (2020)
14. Long, M.; Liang, G.; Zheng, Y.; Li, Z.; Zhong, J.: Cervical cell TCT image detection and segmentation based on multi-scale feature fusion. In: 2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), vol. 5, pp. 192–196 (2021)
15. Li, X.; He, M.; Li, H.; Shen, H.: A combined loss-based multiscale fully convolutional network for high-resolution remote sensing image change detection. IEEE Geosci. Remote Sens. Lett. (2021)
16. Dataset.: COVID-19 Radiography Database (2020). https://www.kaggle.com/tawsifurrahman/covid19-radiography-database
17. Dataset.: COVID-19 Detection X-ray Dataset (2020). https://www.kaggle.com/darshan1504/covid19-detection-xray-dataset
18. Dataset.: Chest X-ray (Covid-19 & Pneumonia) (2020). https://www.kaggle.com/prashant268/chest-xray-covid19-pneumonia
19. Lin, H.; Si, J.; & Abousleman, G.P.: Region-of-interest detection and its application to image segmentation and compression. In: 2007 IEEE International Conference on Integration of Knowledge Intensive Multi-Agent Systems. pp. 306–311 (2007)
20. Simple Thresholding.: http://docs.opencv.org/4.x/d7/dd0/tutorial_js_thresholding.html
21. Flusser, J.; Farokhi, S.; Höschl, C.; Suk, T.; Zitova, B.; Pedone, M.: Recognition of images degraded by Gaussian blur. IEEE Trans. Image Process. **25**(2), 790–806 (2015)
22. Kim, J.Y.; Kim, L.S.; Hwang, S.H.: An advanced contrast enhancement using partially overlapped sub-block histogram equalization. IEEE Trans. Circuits Syst. Video Technol. **11**, 475–84 (2001)
23. End-to-End Deep Learning for Self-Driving Cars.: (2020). https://developer.nvidia.com/blog/deep-learning-self-driving-cars/
24. Feature Scaling for Machine Learning.: Understanding the Difference Between Normalization vs. Standardization (2020). https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/
25. Kingma, D.P.; Ba, J.: Adam: a method for stochastic optimization (2014). arXiv preprint arXiv:1412.6980
26. CoronaHack-Chest X-Ray-Dataset. https://www.kaggle.com/praveengovi/coronahack-chest-xraydataset
27. COVID-19 Patients Lungs X-Ray Images 10000. https://www.kaggle.com/nabeelsajid917/covid-19-x-ray-10000-images
28. Khambampati, A.K.; Liu, D.; Konki, S.K.; Kim, K.Y.: An automatic detection of the ROI using Otsu thresholding in nonlinear difference EIT imaging. IEEE Sens. J. **18**(12), 5133–5142 (2018)