DarkCVNet: Optimized Pneumonia and Covid 19 detection using CXR Images

S Suganyadevi (suganya3223@gmail.com)  
KPR Institute of Engineering and Technology  
https://orcid.org/0000-0001-6096-7510

Seethalakshmi V  
KPRIET: KPR Institute of Engineering and Technology

Research Article

Keywords: Pneumonia, Covid 19, Classification, Accuracy, Hyper-Parameters

Posted Date: July 8th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1294582/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License. Read Full License
DarkCVNet: Optimized Pneumonia and Covid 19 detection using CXR Images

S Suganyadevi 1, V Seethalakshmi 2

1 Research Scholar, Department of ECE, KPR Institute of Engineering and Technology, Coimbatore, India

2 Associate Professor, Department of ECE, KPR Institute of Engineering and Technology, Coimbatore, India

E-mail address: suganya3223@gmail.com, seethav@kpriet.ac.in

https://orcid.org/0000-0001-6096-7510, http://orcid.org/0000-0001-6518-6350

Abstract

Since the advent of the Covid 19 infectious disease, a variety of studies have been conducted around the world in order to accurately forecast its outcome. Covid 19 is linked to the previous lung disease pneumonia, since many individuals died from severe chest congestion (pneumonic condition). The objective of this study is to use computerized x ray images to identify bacterial pneumonia, covid 19 viral pneumonia, other viral pneumonia, lung opacity and normal instinctively. It continues with a lengthy report on developments made in the accurate identification of pneumonia, followed by the authors' technique. For transfer learning, thirteen different deep Convolutional Neural Networks (CNNs) were used: AlexNet, ResNet18, ResNet50, ResNet101, VGG16, VGG19, MobileNetV2, GoogLeNet, DenseNet201, NasNetMobile, NasNetLarge, DarkNet and Inception-ResNet V2. Our data set consisting of 6530 images which are divided into five categories such as bacterial pneumonia, covid 19 viral pneumonia, other viral pneumonia, lung opacity and normal or healthy person’s x ray images, then the pre-processed images were trained for the transfer learning based classification task. Contrasting to other deep learning classification techniques with a large image data base, obtaining a large amount of pneumonia dataset for this classification task is difficult. As a result, we used several data augmentation techniques to enhance the CNN model's validation and classification accuracy as 99.50% and 98.20% respectively, and we produced significant validation accuracy. This DarkCVNet (DarkCovidNetwork) approach could be able to help with the trustworthiness and interpretability issues that come up frequently when dealing with medical images.

Keywords: Pneumonia, Covid 19, Classification, Accuracy, Hyper-Parameters

1 Introduction

Pneumonia is now a dangerous lung infection caused by bacteria, particularly coronavirus or any other viruses, and fungus [1]. Since December 2019, a novel Covid 19 virus (SARS-CoV-2) has spread across Wuhan, China, and a number of other countries. Covid 19 virus, similar to any other virus or bacteria, has been reported to cause pneumonia in some people. In all of these circumstances, however, the treatment is different. The initial screening/testing determines whether or not a person has pneumonia. It is critical to determine either the patient has bacterial pneumonia or Covid 19 viral pneumonia or any other kind of viral pneumonia other than Covid 19 to stop the virus from spreading [2]. By January 1st, 2022, about 305 million cases have been confirmed, with approximately 5.5 million deaths worldwide. Coronavirus (https://www.worldometers.info/coronavirus). Covid 19 is mainly
spread through personal contact from one individual to another. Healthy persons can become infected by coming into contact with those who have COVID-19 through their breath, hands, or mucus [3].

Examining every X-ray image and extracting key information takes a significant amount of time and requires the existence of medical professionals in the field. As a result, medical practitioners will require computer intervention to assist in the diagnosis of COVID-19 pneumonia patients using CXR images [4]. In the current situation, when millions of people must be tested every day to see if they are infected with the deadly COVID-19 viral, an automatic, efficient and precise computer-assisted approach is necessary to distinguish the signs of infection. There have been reports of significant subjective variations in radiologist’s decisions while diagnosing pneumonia. In low-resource countries there is also a limitation of certified radiologist, particularly in rural regions [5]. As a result, there is indeed a pressing demand for computer aided diagnosis systems that can assist radiologists in quickly recognizing distinct kinds of pneumonia from chest X-ray pictures. Deep learning methods in computer assisted approach add greatly to state-of-the-art medical image analysis and exhibit successful outcomes [6]. Most deep learning techniques and CNN models has used chest X-ray image data to detect the disease [7]. Although the availability of a variety of imaging methods, lung radiography is thought to having great substandard sensitivity for relevant examination findings [8]. Medical practitioners routinely employ X-ray imaging to analyze pneumonia, however X-ray imaging systems are an important aspect of medical care around the world [9]. GGO (Ground glass opacity) pertains to the hazy grey regions that may be seen in lungs CT scans or X-rays. Those grey patches represent increased density throughout the lungs. GGO can be caused by a variety of factors, including inflammation, infections and growths. Trusted Source discovered that GGO was the most prevalent abnormality among persons with COVID-19 related pneumonia in a 2020 review [10]. The proposed study assessed the performance of thirteen distinct pre-trained network designs (AlexNet, ResNet18, ResNet50, ResNet101, VGG16, VGG19, MobileNetV2, GoogLeNet, DenseNet201, NasNetMobile, NasNetLarge, DarkNet and Inception-ResNet V2) utilizing a transfer learning approach.

The state-of-the-art approaches for COVID-19 diagnosis are discussed in Section 2. In section 3, we covers the materials and methods of our proposed model. In section 4, we described results and discussions as well as some suggestions for future improvements and finally, in section 5, we provide our findings.

2 Related works

In contrast to deep learning systems, machine learning relies heavily on knowledge in the selection and extraction of appropriate features and has restricted performance. As a result, deep learning approaches have become increasingly popular in recent years. The followings are some of state of the art algorithms are based on deep learning and are built utilizing chest X-ray pictures or CT scans. Govardhan Jain et al proposed a two stage technique (ResNet 50 and ResNet 101) to categorize COVID-19 from normal X-ray images and produced 97.77% accuracy for 1832 images [2]. Mesut Togacar et al. proposed a deep learning models (MobileNetV2 and SqueezeNet) and used a SMO method for feature extraction with 99.27% accuracy [4]. Shervin Minaee et al. performed transfer learning for four CNN models (ResNet 18, ResNet 50, SqueezeNet, DenseNet 121) with 98% sensitivity [5]. Ioannis D. Apostolopoulos et al. utilized deep transfer learning for two different datasets with 96.78% accuracy [6]. Harsh Panwar et al. proposed gradient based colour visualization technique and achieved 95.61% accuracy [10]. Chaimae Ouchicha et al. introduced CVDNet model with 97.20% accuracy [11]. Prabira Kumar, Sethy et al. achieved 98.66% for ResNet 50 with support vector machines [12]. Amit Kumar Jaiswal et al. utilized a pixel wise segmentation with mask RCNN model and achieved higher accuracy
[13]. Tulin Ozturk et al. introduced DarkNet model with 87.02% accuracy for multi class classification [14]. Table 1 describes some of the deep learning techniques with its accuracies.

### Table 1 Different Deep Learning Models and its Performance

<table>
<thead>
<tr>
<th>S.No</th>
<th>Author</th>
<th>Deep Learning Techniques</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Apostolopoulos and Bessiana et al. [2]</td>
<td>VGG 19</td>
<td>97.8%</td>
</tr>
<tr>
<td>3.</td>
<td>Özturk et al. [14]</td>
<td>DarkCovidNet</td>
<td>87%</td>
</tr>
<tr>
<td>4.</td>
<td>Sethy et al. [12]</td>
<td>ResNet50</td>
<td>95.4%</td>
</tr>
<tr>
<td>5.</td>
<td>Yoo et al. [15]</td>
<td>ResNet 18</td>
<td>95.0%</td>
</tr>
<tr>
<td>6.</td>
<td>Panwar et al. [16]</td>
<td>nCOVnet based on VGG16</td>
<td>88.1%</td>
</tr>
<tr>
<td>7.</td>
<td>Albahli et al. [17]</td>
<td>ResNet152</td>
<td>87.0%</td>
</tr>
<tr>
<td>8.</td>
<td>Civit-Masot et al. [18]</td>
<td>VGG16</td>
<td>86.0%</td>
</tr>
<tr>
<td>9.</td>
<td>Wang et al. [19]</td>
<td>DeCovNet</td>
<td>90.1%</td>
</tr>
<tr>
<td>10.</td>
<td>Singh et al. [20]</td>
<td>MODE-based CNN</td>
<td>93.3%</td>
</tr>
<tr>
<td>11.</td>
<td>Ahuja et al. [21]</td>
<td>ResNet 18</td>
<td>99.4%</td>
</tr>
</tbody>
</table>

### 3 Materials and Methods

The suggested approach is divided into four steps, as illustrated in Fig. 4, namely i) Dataset Collection ii) Image pre-processing iii) Image augmentation (iv) Pre-trained CNNs to distinguish pneumonia cases caused by bacteria, viral, or specifically the Covid 19 virus, lung opacity, and Healthy.

MATLAB (R2020a) was used to train, assess, and test several algorithms in this work. To train the model, an Intel i9 9820X processor with an NVIDIA GeForce RTX 2080 Ti with 16 GB memory is employed.

#### 3.1 Dataset

Kaggle is the open-source repository that was used to collect CXR images of all five cases analysed, namely, Lung Opacity, normal / healthy, bacterial pneumonia, Covid 19 viral Pneumonia and any other viral Pneumonia. There are 3840 images from various matters pretentious by pneumonia (1290 images of bacterial pneumonia, 1345 images of Covid 19 pneumonia and 1205 images for other viral pneumonia), 1345 images for lung opacity and 1345 images from healthy persons among the 6530 chest X-ray images which is known as dataset1 and after performing different augmentation techniques the image count was increased to 16000 which is known as dataset2. We used 80 percent of the pictures for training and the rest 20% for testing. Furthermore, we used 20% of the data from the 80 percent of training data for validation purposes. Table 2 provides the dataset details associated with this study.

### Table 2 Total number of images in each category

<table>
<thead>
<tr>
<th>S. No</th>
<th>Image Category</th>
<th>Original No. of Images Dataset 1</th>
<th>Training Set (after Augmentation) Dataset 2</th>
<th>Test Dataset</th>
<th>Validation Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Healthy</td>
<td>1345</td>
<td>3200</td>
<td>240</td>
<td>250</td>
</tr>
<tr>
<td>2.</td>
<td>Lung Opacity</td>
<td>1345</td>
<td>3200</td>
<td>240</td>
<td>250</td>
</tr>
<tr>
<td>3.</td>
<td>Covid 19 Pneumonia</td>
<td>1345</td>
<td>3200</td>
<td>240</td>
<td>250</td>
</tr>
<tr>
<td>4.</td>
<td>Bacterial Pneumonia</td>
<td>1290</td>
<td>3200</td>
<td>240</td>
<td>250</td>
</tr>
</tbody>
</table>
3.2 Image preprocessing

Images are preprocessed in two phases.

Step 1: Image Resizing is the key step in preprocessing method. All of the images that were initially acquired has been scaled to the desired size of 299×299.

Step 2: The grayscale (input) resized images are filtered with an edge-preserving Gaussian bilateral filter. This filter smooth’s neighborhood’s with smallest variance (identical areas) and not neighborhood’s with significant variance (strong edges) while degree of smoothing is a small value.

\[ I_{\text{filtered}}(x) = \frac{1}{W_p} \sum_{x_i \in \Omega} f_r(\| I(x_i) - I(x) \|) g_s(\| x_i - x \|) \]  \hspace{1cm} (1)

\[ W_p(\text{Normalization term}) = \sum_{x_i \in \Omega} f_r(\| I(x_i) - I(x) \|) g_s(\| x_i - x \|) \]  \hspace{1cm} (2)

\( I_{\text{filtered}}(x) \) represents the filtered output image shows in equation (1) and equation (2). I represents the original image input that would be filtered. \( x \) denotes the present pixel to be filtered's coordinates. \( \Omega \) is the window adjusted in \( x \), then \( x_i \in \Omega \) is a different pixel. The range kernel \( f_r \) (Gaussian function) is

![Figure 1. The sample X ray images of 5 different classes taken from Edge-preserving Gaussian Bilateral Filter: a) Healthy Lung image b) Lung Opacity c) Covid Pneumonia d) Bacterial Pneumonia e) other Viral Pneumonia](image-url)
used to smooth out disparities in intensity. The domain (or spatial) kernel for smoothing coordinate variances is $g_s$ (Gaussian function).

### 3.3 Image Augmentation

As previously stated, CNNs perform better when dealing with huge datasets. The working database, on the other hand, is not that large. A prevalent trend in deep learning algorithm training is to use data augmentation techniques to turn a small dataset into a larger one. On the off chance that the genuine number of gathered pictures in each picture class varies, this difference in picture count makes a critical class awkwardness issue. This imbalanced information would prompt an assortment of issues, for example, over fitting, where the model can't sum up well to obscure datasets. Interestingly, exactness is definitely not a reasonable key measure in the present circumstance. The state of over fitting prompts the organization to get familiar with the particulars of the preparation dataset tests to where it can't sum up as expected. Subsequently, regularization procedures, for example, information expansion are utilized in this exploration to battle this issue of over fitting.

![Sample Augmented Image Datasets](image)

**Figure 2. Sample Augmented Image Datasets** (a) Original Image (b) Rotation [-50 50] (c) XTranslation [-50 50], YTranslation [-50 50] (d) Scaling [1.2, 1.5] (e) XReflection, YReflection (f) Gaussian Blur

Instead of gathering new data, augmenting existing data can increase the efficiency of deep learning techniques. Rotation, XTranslation, YTranslation, Scaling, XReflection, YReflection and Gaussian blur are the data augmentation techniques used in this study to increase the image count from 6530 to 16000. Images are then rotated at various angles between -50 and 50 degrees, Scaling could be done with the ratio of [1.2,1.5], XTranslation [-50 50], YTranslation [-50 50], XReflection, YReflection and Gaussian Blur to generate a larger number of images, which would then be included in the training dataset which are represented in figure 2.

### 3.4 Proposed model

Transfer learning is a methodology where the information obtained by a CNN from gave information is moved to tackle a related yet unique issue, typically requiring new information from a restricted example to prepare a CNN without any preparation. CNN utilizes a few structure blocks like as convolution layers, pooling layers, and completely associated layers to learn spatial orders of data independently and adaptively by backpropagation. For transfer learning, thirteen different deep Convolutional Neural Networks (CNNs) AlexNet, ResNet18, ResNet50, ResNet101, VGG16, VGG19, MobileNetV2, GoogLeNet, DenseNet201, NasNetMobile, NasNetLarge, DarkNet and Inception-ResNet V2 were used to perform multi-class classification in this study. Figure 3 shows the workflow of our model.
As a base model, we have utilized 13 different Pre-trained CNNs on ImageNet with pre-trained weights. The suggested model (Fine-tuned DarkCVNet) has been taught to adapt and learn the basic features of computer vision (e.g., edges and boundaries), ensuring that the model does not have to learn from scratch each time it is used to train on CXR datasets. We investigated the most appropriate hyperparameters as maximum epoch is 10 and maximum iteration is 500 and have given various initial learning rates (0.0001, 0.0003, 0.001) for different optimizers (SGD, ADAM, RMSProp) using that the maximum accuracy for the suggested model has been achieved after several attempts of experimentation. Figure 4 shows our proposed model architecture.
AlexNet

AlexNet is a CNN model that essentially affects the utilizations of profound figuring out how to machine vision. It astonishingly won the 2012 ImageNet LSVRC-2012 challenge by a huge degree (15.3 percent botch rates versus 26.2 percent blunder rates in runner up which is VGG-16). The organization's configuration was very like LeNet that of Yann LeCun et al's LeNet, yet it was more profound, with more channels per layer and layered convolutional layers. Convolutions, max pooling, dropout, information increase, ReLU initiations, and SGD with force were all important for it. After each convolutional and completely associated layer, it added ReLU initiations. Moreover, instead of regularisation, dropout is used to deal to address overfitting.

ResNet 18, 50, 101

The ResNet50 model was the winner of ILSRVC 2015, with a 3.57 percent error rate and the input image size is 224 by 224 pixels. ResNet is the very well deep learning method that was first published by Shaoqing Ren, Kaiming He, Jian Sun, and Xiangyu Zhang in a publication. ResNet18 model consists of 18 layers and the ResNet50 is a deep 50 layered ResNet blocks, each with two or three convolutional layers. ResNet-101 comprises of 101 layers of convolutional neural network [25]. This is established on the knowledge of skip connections and makes extensive use of batch-normalization to successfully train several of layers without losing speed.

Vgg 16, 19

VGG is a type of CNN that has been around for quite some time. It was derived from research into how to densify such networks. The network employs small 3 x 3 filters. Aside from that, the network is also known by its simplicity, with only pooling layers as well as a fully linked layers as additional features. Vgg16 model consists of 16 layers and Vgg19 has 19 layer deep.

MobileNetV2

MobileNetV1 introduces Depth wise Separable Convolution, which drastically reduces the network's complexity costs and model size, making it ideal for mobile platforms or devices with limited processing capacity. In MobileNetV2, an improved module with inverted residual pattern is introduced. Here non-linearities in restricted layers are eliminated. With MobileNetV2 as the spine for highlight extraction, state-of-the-art object identification and semantic segmentation results also are accomplished.

Google/Inception

The ILSVRC 2014 challenge was awarded by the Inception Network or GoogLeNet, that had a top-5 failure rate of 6.67 %, and this was practically human level ability. The model was designed by Google and features an upgraded version of the original LeNet design. This is modelled on the inception module idea. GoogLeNet is a 22 layer deep CNN that is a version of the Inception Network. 9 inception modules are placed on top each in the InceptionNet/GoogleLeNet design, having max pooling layers among them (to halve the spatial space). There are 22 layers in total (27 with the pooling layers). From the last inception module, it applies global average pooling.

DenseNet201

Huang et al. [26] presented the DenseNet201 architecture, which is one of the most recent variants based on the Dense network. Each layer in DenseNet gets extra data from all past layer and
sends its own characteristic to every single ensuing layer. The term link is utilized. Each layer gets aggregate information from the levels above it. A pooling layer and bottleneck development are likewise utilized in the DenseNet201 model. Here, the blunder worth can be passed on more straightforwardly to past layers. As past layers can get close oversight from the last order layer, this is a type of certain more profound management. This is the advantage of this model. DenseNet201 is used in this work which consists of 201 layers.

**NasNet**

Google has introduced NASNet that structured the task of finding the ideal CNN architecture as a Reinforcement Learning problem, owing to its vast computer capacity and engineering talent. The basic idea would have been to find the optimal combination of filter sizes, strides, output channels, number of layers, and other characteristics in the specified search space. The accuracy of the searched architecture on the given dataset was the reward for each search action in this Reinforcement Learning environment.

In the ImageNet competition, NASNet received a state-of-the-art result. The computer power required for NASNet, on the other hand, was so large that only a few companies were able to apply the same technology. NasNetMobile and NasNetLarge are the two techniques used here.

**DarkNet 19,53**

In the same way as Highway Networks and ResNet were presented to resolve the vanishing gradient difficult, DenseNet was offered to solve it (Huang et al. 2017, He et al. 2015a, Srivastava et al.). Several layers might yield very tiny or no knowledge because ResNet expressly maintains knowledge through additive identity modifications. To alleviate this difficulty, DenseNet employs cross-layer connections, albeit in an improved version. DenseNet connects every previous layer to the next layer in a feed forward technique. DarkNet19 is 19 layers deep and DarkNet53 consists of 53 layers. All subsequent layers used feature-maps of previous levels as inputs represented by the following equations of 3 and 4.

\[ F^k_x = g_c(l_c, k_1) \]  (3)

\[ F^k_l = g_k(F^k_t, ..., F^k_{t-1}) \]  (4)

**Inception-Resnet-V2**

One of the upgraded version of Inception V1 and Inception V2 model is Inception-ResNet-V2. Inception V4 with lingering associations (Inception-ResNet) has a similar summed up ability as standard InceptionV4, yet additionally with higher profundity and width, as indicated by Szegedy et al. They found that Inception-ResNet combines quicker than Inception V4, inferring that preparation Inception networks with leftover associations fundamentally speeds up the interaction.

A parameter would be a variable which is automatically optimized throughout the training phase, whereas a hyper parameter is one that must be set ahead of time. The following figure 5
represents the hyper parameters which are involved in enhancing the performance of our model.

**Figure 5. Hyper parameters of CNN Model**

The number of epochs refers to how many times the full training data is run through the neural network. We should increase the number of epochs until the difference between the test and training errors is negligible, here 10 is used. In the learning phase of convnet, mini-batch is frequently preferred. A range of 16 to 128 is a decent starting point for testing. It's worth noting that batch size affects convnet. Here 50 is used. The model's non-linearity is introduced via the activation function. In most cases, a rectifier and a convnet work well together. Sigmoid, tanh, and other activation functions, depending on the task, are also options. The learning rate determines how frequently the weight in the optimization method is updated. Depending on the optimizer we employ (SGD, Adam, RMSProp), we can utilise a 3 different constant learning rate of 0.0001, 0.0003 and 0.001, a progressively decreasing learning rate, momentum-based approaches, or adaptive learning rates.

**Optimizer**

One of the most crucial phases is selecting an algorithm for optimizing a neural network. The most commonly utilized optimization algorithms such as Stochastic Gradient Momentum (SGDM), Adaptive moment (ADAM) and Root Mean Squared Propagation (RMSProp) are investigated in the field of deep learning in this paper. When these algorithms run on minimal resources, DL approaches become more relevant in order to reduce recurring costs and give efficient results in less time. An optimizer is a strategy or algorithm for updating different parameters so that the loss can be reduced with far less effort. The following figure 6 shows the different kinds of optimizers available for neural network design.
SGDM

Due to its tendency to sway, SGD struggles arriving at a worldwide ideal, particularly on steep surface bends. Slopes that are boisterous or little may likewise be dangerous. In such examples, the force approach is planned to assist learning. It essentially looks to resolve two issues: helpless Hessian grid molding and stochastic inclination fluctuation. The target behind this methodology is to compute a running normal by remembering the earlier update for the current change like there were force from past updates. When utilizing SGD with energy, it can meet quicker and have less swaying. The speed variable \( v \) is utilized in SGD with energy. The heading and speed with which the boundaries travel through boundary space is called speed. It's set to the negative gradient's exponentially declining average. In addition, SGD with momentum necessitates the addition of a new hyper parameter, the momentum parameter, which controls how rapidly the contributions of prior gradients exponentially decay.

\[
v = \alpha v - \epsilon \frac{1}{m} \nabla_{\theta} \left( \sum_{i=1}^{m} L(f(x^{(i)}; \theta), y^{(i)}) \right) \\
\theta = \theta + v
\]

ADAM

Adam changes the learning rate for each weight of the neural organization by assessing the first and second snapshots of angle. Adam incorporates the advantages of AdaGrad, which is really great for inadequate inclinations, and RMSProp, which is great for on the web and non-fixed conditions. Adam has a significant advantage. First and foremost, the learning rate must be fine-tuned. It is also an easy-to-use approach that is insensitive to gradient rescaling on the diagonal. It has a low memory footprint and is computationally efficient. Furthermore, Adam is well suited to non-stationary targets as well as issues with very noisy and sparse gradients. The first moment variable \( m \) and second moment variable \( u \), are required by the Adam method. At time step \( t \), biased first and second moment estimates are updated after computing gradient is given in equation (7) and (8).

\[
m_t = \rho_t m_{(t-1)} + (1 - \rho_t) g_t \\
u_t = \rho_t u_{(t-1)} + (1 - \rho_t) g \odot g
\]

The first and second moments are then used to correct bias. Parameter changes are generated and implemented using revised moment estimates is given in (9), (10), (11), (12).
\[
m_t^\wedge = \frac{m_t}{(1-\rho_t^1)} \\
u_t^\wedge = \frac{u_t}{(1-\rho_t^2)} \\
\Delta \theta = -\varepsilon \frac{m_t^\wedge}{\sqrt{u_t^\wedge + \delta}} \\
\theta_t = \theta_{t-1} + \Delta \theta
\]  

(9)  
(10)  
(11)  
(12)  

3. RMSProp

RMSProp is another AdaGrad-modifying algorithm. By adopting an exponential average of the gradients rather than the cumulative sum of squared gradients, this seeks to lower the aggressiveness of the learning rate. The adaptive learning rate stays same, as the exponential average now punishes higher learning rates in conditions with fewer updates and lower rates in conditions with a greater number of updates. The squared gradient is generated after calculating gradient in terms of developing RMSProp is given in (13).

\[
r = \rho r + (1 - \rho) g \odot g
\]  

(13)  

Where \(\rho\) is the rate of decay. Then, as seen below, the parameter update is calculated and applied is given in (14) and (15).

\[
\Delta \theta = -\varepsilon \frac{m_t^\wedge}{\sqrt{u_t^\wedge + \delta}} \odot g
\]  

(14)  

\[
\theta = \theta + \nu
\]  

(15)

4 Results and Discussions

Covid 19 cases have been confirmed in 305 millions of people around the world, with 5 million of deaths as on 1st January 2022. Covid 19 has been labeled a global epidemic by the World Health Organization. We were able to identify Covid 19 from CXR data using the described method. Because pneumonia is one of Covid 19's symptoms, we compared Covid 19 viral pneumonia lung image, bacterial pneumonia lung image, other viral induced pneumonia lung image, lung opacity image and normal lung image by using thirteen different CNN models. The method starts with downsizing chest X-ray images to a fraction of their original size. The convolutional neural network architecture, which collects features of an image and classifies it, then identification and classification could be done in the next stage. Our proposed DarkCVNet architecture is based on the baseline model of DarkNet19. In this study, MATLAB (R2020a) was used to train, evaluate, and test numerous algorithms. An Intel i9 9820X processor with an NVIDIA GeForce RTX 2080 Ti with 16 GB memory is employed to train the model. The experimental study is based on the performance measurable matrices produced from the confusion matrix such as TP (True Positive), FP (False Positive), TN (True Negative), FN (False Negative), Accuracy, Sensitivity, Specificity, F-Score etc., which is shown in table 3.

<table>
<thead>
<tr>
<th>Sl. No</th>
<th>Metrics</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TP</td>
<td>If COVID-19 is identified in a COVID-19 affected person.</td>
</tr>
<tr>
<td>2</td>
<td>TN</td>
<td>If an individual is identified as NONCOVID-19 correctly</td>
</tr>
<tr>
<td>No.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>3</td>
<td>FP</td>
<td>Depicts an inaccurate identification in which a healthy person is found to have COVID-19.</td>
</tr>
<tr>
<td>4</td>
<td>FN</td>
<td>Indicates an inaccurate identification in which a person affected with COVID-19 is mistakenly identified as a healthy person.</td>
</tr>
</tbody>
</table>
| 5   | Accuracy | \[
\frac{\text{No. of Images Correctly predicted as both Covid} - 19 \text{ and Non Covid}}{\text{Total No. of Images}} = \frac{\text{TP} + \text{TN}}{(\text{TP} + \text{TN} + \text{FP} + \text{FN})}
\] |
| 6   | Sensitivity/Recall | \[
\frac{\text{No. of Images Correctly predicted as Covid} - 19}{\text{Total No. of Covid} - 19 \text{ Images}} = \frac{\text{TP}}{(\text{TP} + \text{FN})}
\] |
| 7   | Specificity | \[
\frac{\text{No. of Images Correctly predicted as Non Covid} - 19}{\text{Total No. of Non Covid} - 19 \text{ Images}} = \frac{\text{TN}}{(\text{TN} + \text{FP})}
\] |
| 8   | Precision | \[
\frac{\text{No. of Images Correctly predicted as Covid} - 19}{\text{Total No. of predicted Positive Images}} = \frac{\text{TP}}{(\text{TP} + \text{FP})}
\] |
| 9   | F-Measure | \[
2 \times \frac{\text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}
\] |
<p>| 10  | Confusion Matrix | Predicted Class |
|     | True Positive (TP) | False Negative (FN) |
|     | False Positive (FP) | True Negative (TN) |</p>
<table>
<thead>
<tr>
<th>CNN Technique</th>
<th>No. Of Layers</th>
<th>Image Size</th>
<th>Parameters (Millions)</th>
<th>Optimizer</th>
<th>Accuracy for Learning Rate for Dataset 1</th>
<th>Accuracy for Learning Rate for Dataset 2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.0001</td>
<td>0.0003</td>
</tr>
<tr>
<td>AlexNet</td>
<td>25</td>
<td>227×</td>
<td>61</td>
<td>SGDM</td>
<td>82.81%</td>
<td>75.78%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>227</td>
<td></td>
<td>ADAM</td>
<td><strong>83.59%</strong></td>
<td>74.22%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RMSPROP</td>
<td>64.84%</td>
<td>78.13%</td>
</tr>
<tr>
<td>ResNet18</td>
<td>71</td>
<td>224×</td>
<td>11.7</td>
<td>SGDM</td>
<td>84.38%</td>
<td>67.97%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>224</td>
<td></td>
<td>ADAM</td>
<td><strong>85.16%</strong></td>
<td>83.59%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RMSPROP</td>
<td>74.22%</td>
<td>68.75%</td>
</tr>
<tr>
<td>ResNet50</td>
<td>177</td>
<td>224×</td>
<td>25.6</td>
<td>SGDM</td>
<td>76.56%</td>
<td>76.56%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>224</td>
<td></td>
<td>ADAM</td>
<td>78.13%</td>
<td>82.81%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RMSPROP</td>
<td>78.91%</td>
<td>78.91%</td>
</tr>
<tr>
<td>ResNet101</td>
<td>347</td>
<td>224×</td>
<td>44.6</td>
<td>SGDM</td>
<td>83.59%</td>
<td>78.13%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>224</td>
<td></td>
<td>ADAM</td>
<td>76.56%</td>
<td>72.66%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RMSPROP</td>
<td>67.19%</td>
<td>71.88%</td>
</tr>
<tr>
<td>VGG16</td>
<td>41</td>
<td>224×</td>
<td>138</td>
<td>SGDM</td>
<td>70.31%</td>
<td>67.97%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>224</td>
<td></td>
<td>ADAM</td>
<td>80.47%</td>
<td>81.25%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RMSPROP</td>
<td>57.03%</td>
<td>67.19%</td>
</tr>
<tr>
<td>VGG19</td>
<td>47</td>
<td>224×</td>
<td>144</td>
<td>SGDM</td>
<td><strong>84.38%</strong></td>
<td>83.59%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>224</td>
<td></td>
<td>ADAM</td>
<td>79.69%</td>
<td>77.34%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RMSPROP</td>
<td>71.88%</td>
<td>72.66%</td>
</tr>
<tr>
<td>DenseNet201</td>
<td>708</td>
<td>224×</td>
<td>20</td>
<td>SGDM</td>
<td><strong>82.81%</strong></td>
<td>77.34%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>224</td>
<td></td>
<td>ADAM</td>
<td>69.53%</td>
<td>82.81%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RMSPROP</td>
<td>57.81%</td>
<td>76.56%</td>
</tr>
<tr>
<td>MobileNetV2</td>
<td>154</td>
<td>224×</td>
<td>3.5</td>
<td>SGDM</td>
<td>69.53%</td>
<td><strong>81.25%</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>224</td>
<td></td>
<td>ADAM</td>
<td>71.88%</td>
<td>80.47%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RMSPROP</td>
<td><strong>81.25%</strong></td>
<td>69.53%</td>
</tr>
<tr>
<td>GoogleNet/InceptionNet</td>
<td>144</td>
<td>224×</td>
<td>7</td>
<td>SGDM</td>
<td>82.03%</td>
<td>76.56%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>224</td>
<td></td>
<td>ADAM</td>
<td><strong>85.94%</strong></td>
<td>78.91%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RMSPROP</td>
<td>65.63%</td>
<td>75.78%</td>
</tr>
<tr>
<td>NasNet-mobile</td>
<td>913</td>
<td>224×</td>
<td>5.3</td>
<td>SGDM</td>
<td>68.75%</td>
<td>79.69%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>224</td>
<td></td>
<td>ADAM</td>
<td>75.78%</td>
<td>78.13%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RMSPROP</td>
<td>71.09%</td>
<td>65.63%</td>
</tr>
<tr>
<td>NasNet-Large</td>
<td>1243</td>
<td>331×</td>
<td>88.9</td>
<td>SGDM</td>
<td>75.78%</td>
<td>82.03%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>331</td>
<td></td>
<td>ADAM</td>
<td>78.91%</td>
<td>71.09%</td>
</tr>
</tbody>
</table>
The following Table 4 provides performance of Various CNN Techniques for 3 different optimizers such as SGD, ADAM and RMSProp with different learning rates of 0.0001, 0.0003 and 0.001.

<table>
<thead>
<tr>
<th>Model</th>
<th>Width</th>
<th>Height</th>
<th>Depth</th>
<th>SGD Learning Rate</th>
<th>ADAM Learning Rate</th>
<th>RMSProp Learning Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>DarkNet 19</td>
<td>64</td>
<td>256×21</td>
<td>256</td>
<td>70.31% 75.78%</td>
<td>80.47% 88.25%</td>
<td>57.03% 85.16%</td>
</tr>
<tr>
<td>DarkNet 53</td>
<td>184</td>
<td>256×41</td>
<td>256</td>
<td>84.38% 95.27%</td>
<td>79.69% 91.56%</td>
<td>63.28% 74.17%</td>
</tr>
<tr>
<td>Inception-ResNet-V2</td>
<td>824</td>
<td>299×55.9</td>
<td>299</td>
<td>78.91% 89.80%</td>
<td>83.59% 94.48%</td>
<td>72.66% 83.54%</td>
</tr>
</tbody>
</table>

We have made a comparison for thirteen different fine-tuned pre-trained CNN models with 3 different optimizers such as SGD, ADAM, and RMSProp with 3 different learning rates as 0.0001, 0.0003, and 0.001. Our proposed DarkCVNET model outperforms the other state of the art methods with training accuracy of 98.20% and validation accuracy of 99.10% with ADAM optimizer and with the 0.001 learning rate. The following figure 6 provides the confusion matrix for 5 class classification of our proposed DarkCVNet model.
Table 5 also includes precision, recall, specificity, F-score, and accuracy statistics for the five-class classification test. Table 5 shows that the suggested model detected Pneumonia and Covid19 with an average accuracy of 98.20 percent, with average precision, recall, specificity, and F-score values of 98 percent, 98.1 percent, and 99.49 percent, and 97.99 percent, respectively.

Table 5 TP, FP, TN, FN, Precision, Recall, Specificity, F-Score and Accuracy of our DarkCVNet model
Performance analysis could be done with some parameters such as Precision, Recall, Specificity, F-Score and Accuracy for 5 classes of Bacterial Pneumonia, Covid Pneumonia, Healthy, Lung Opacity and other Viral Pneumonia lung images which is described in figure 7.

The plots could reveal useful information about the model's training, such as its speed of convergence over epochs (slope). Is it possible that the model has already converged or not (plateau of the line). Is it possible that the mode is over-learning the training data or not (inflection for validation line). Our proposed model was trained for 100 epochs with 5000 iterations and it was produced 99.50% validation accuracy which is shown in figure 8. Each iteration involves a gradient estimation and a network parameter update. Apart from tuning hyper parameters, Class imbalance is a major issue in deep learning models because model will not behave well for the unseen dataset.
Our fine-tuned DarkCVNet model provides highest accuracy of 98.20% with 0.001 learning rate of ADAM optimizer. Some of the sample images obtained from DarkCVNet model is shown in figure 9. Since the X ray images of infection are not always clear, they are frequently misidentified as other diseases and benign anomalies. Among the most significant drawbacks of lung radiography analysis is their inability to detect Covid 19 in its early phases due to a lack of sensitivity in GGO monitoring. Furthermore, specialists may misclassify bacterial or viral induced pneumonia photos, resulting in patients receiving incorrect medicine and, as a result, deteriorating their condition. Another limitation is over fitting which means model has not learned the decision boundaries and hasn't been extended to unseen data and has only learned the data that should be avoided.

In the future, we want to add more images to our model to validate it. This model can be stored in the cloud to give fast diagnosis and aid in the rehabilitation of impacted patients. This should result in a significant reduction in clinician workload. We will use CT scans to detect COVID 19 and compare the results to the suggested model trained on X-ray pictures.

5 Conclusion

We described a deep learning approach for pneumonia detection from Chest X rays by fine-tuning thirteen pre-trained convolutional models (VGG16, VGG19, AlexNet, DenseNet201, ResNet50, ResNet101, MobileNet, and GoogleNet) on our training set that have shown promise in a number of tasks in recent years, and evaluate their effectiveness in terms of pneumonia diagnosis. Before using deep learning techniques to train each image, we preprocessed it. The application of pre-processing techniques to the images is one of the new novel aspect of the suggested approach. More efficient characteristics are derived from the image data when pre-processing techniques are used. Here Gaussian Bilateral filter is used to smooth’s both uniform areas and neighborhoods with greater variance as degree of smoothing increases. In our study, thirteen fine-tuned pre-trained models were compared with augmented dataset and non-augmented dataset. Our fine-tuned DarkCVNet model was achieved with an accuracy of 99.16% with 0.001 learning rate of ADAM optimizer. We hope that this CAD (computer-assisted diagnostic) tool can considerably assist the radiologist in taking clinically better useful images and quickly identifying pneumonia and its type.
Declarations:

Funding

No funding was received to assist with the preparation of this manuscript.

Conflicts of interest

The authors have no competing interests to declare that are relevant to the content of this article.

Availability of data and material

The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Code availability

The code generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Authors’ contributions

All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by S.Suganyadevi, V.Seethalakshmi. The first draft of the manuscript was written by S.Suganyadevi and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

References:


