Abstract

COVID-19 is a severe respiratory tract infection. COVID-19 caused by SARS-CoV-2 can readily spread through contact with an infected person. Monotonically increasing SARS-CoV-2 infections have not only wasted lives but also severely damaged the financial systems of both developing and developed countries. This high spread rate pressure on the health care systems which raises the need for fast methods for diagnosing this disease. Convolutional Neural Networks (CNN) show great success for various computer vision tasks. However, CNN is a scale-variant model and is computationally expensive. In this Thesis, novel architectures are proposed for multiscale feature extraction and classification and lightweight architecture for COVID-19 diagnosing. The proposed I which is a lightweight CNN model exploits spatial kernel separability to reduce the number of the training parameters to a large extent and regularize the model to only learn linear kernel. Furthermore, This model uses residual connection and batch normalization extensively to maintain the network stability during the training process and provide the model with the regularization effect to reduce overfitting. Proposed CNN II learns multiscale features using a pyramid of shared convolution kernels with different atrous rates. This scale-invariant CNN uses an attention-based mechanism that is used to guide and select the correct scale for each input. Proposed CNN II is an end-to-end trainable network and exploits a novel augmentation technique, Texture Augmentation, to reduce overfitting. The lightweight architecture is trained using the QaTa-Cov19 benchmark dataset achieving 100% for accuracy, sensitivity, precision, and F1-score with a very low parameter count (150K) compared with the other methods in the literature. Proposed method II achieved a 0.9929 for $F1 - score$ tested on the QaTa-Cov19 benchmark dataset with a total of $5,040,571$ trainable parameters.

Acknowledgements

Praise be to ALLAH first and last. Prayers and peace be upon MOHAMMED the Messenger of ALLAH.

I would like to thank my supervisors and my teachers *Dr. Ahmed M. Hamad* and *Prof. Khalid M. Amin* for their help and guide during my Study. They have been helpful with background information and have continually encouraged me and helped me with comments on my work. They always had time to discuss new ideas and give feedback on early ideas and research problems.

Contents

List of Figures

List of Tables

List of Abbreviations

COVID-19	COronaVIrus Disease 2019
SARS-CoV-2	severe acute respiratory syndrome coronavirus 2
WHO	World Health Organization
rRT-PCR	real-time reverse transcription polymerase chain reaction
TMA	transcription-mediated amplification
RT-LAMP	reverse transcription loop-mediated isothermal amplification
CXR	Chest X-rays
CNN	convolutional Neural Network
tanh	Hyperbolic tangent
ReLU	Rectified Linear Unit
BoW	Bag-of-Words
SVM	Support Vector Machine
ICS	Internal Covariate Shift
BN	Batch Normalization
ASPP	Atrous Spatial pyramid
RC	Representation-based classification
BGWO	binary grey wolf optimization
ANN	Artificial neural network
PCA	principal component analysis
DT	Decision Tree
NB	Naive Bayes
FOSF	first order statistical features
GLCM	Gray Level Co-occurrence Matrix
HOG	Histogram of Oriented Gradients
KNN	k-nearest neighbors
RBF	radial basis function
RSB	Residual Separated Block
SWASPP	Spatially weighted Atrous Spatial Pyramid
DSWASPP	Densely stacked SWASPP
SPP	Spatial Pyramid Pooling

SASPP-net	Switchable Atrous Spatial Pyramid Pooling
FX	feature extraction modules
Adam	Adaptive Moment Estimation

To my Family

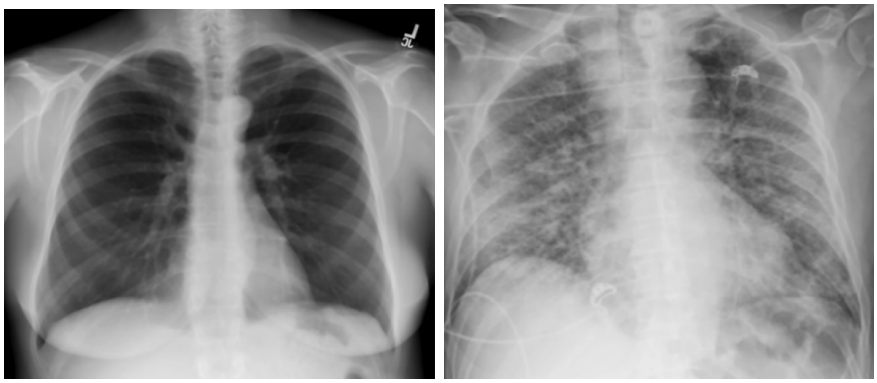
Chapter 1

Introduction

1.1 Novel Coronavirus Disease

COronaVIrus Disease 2019 (COVID-19) is a severe respiratory tract infection that can range from mild to lethal [1]. COVID-19 is a contagious disease that can readily spread through direct or indirect contact with an infected person [2]. COVID-19 is caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2). COVID-19 initially appeared in Wuhan, China late 2019 and spread worldwide [3]. World Health Organization (WHO) declared the outbreak of COVID-19 as a Public Health Emergency of International Concern in January 2020, and a pandemic in March 2020 [4]. Ongoing SARS-CoV-2 infections have not only devastated human lives but also significantly damaged the financial health of both developing and developed countries. Therefore, there is an urgent need to control the pandemic by accelerating the development and mass production of efficacious vaccines against SARS-CoV-2. Healthcare practitioners, researchers, and policymakers around the globe were thrown a challenge to deliver adequate prevention and treatment modalities to combat the pandemic. From the initial stage of this pandemic, scientists were focused on either repurposing the existing drugs or developing vaccines against COVID-19 [5]. Fig. 1.1 illustrates the difference between the COVID-19 and normal lungs.

Typical diagnostic tools for COVID-19 are virus' nucleic acid by real-time reverse transcription polymerase chain reaction (rRT-PCR), transcription-mediated amplification (TMA), or reverse transcription loop-mediated isothermal amplification (RT-LAMP) from a nasopharyngeal swab [6]. These manual traditional methods are time-consuming and complex. Chest X-rays (CXR) and chest CT offer fast screening methods for COVID-19 [7][8][9]. CXR and chest CT are preferred when RT-PCR testing is not available in time [10]. CXR has many



(A) Chest X-ray image of normal lung. (B) Chest X-Ray image of COVID-19 patient

FIGURE 1.1: Samples of CXR images of Normal and COVID-19 pneumonia

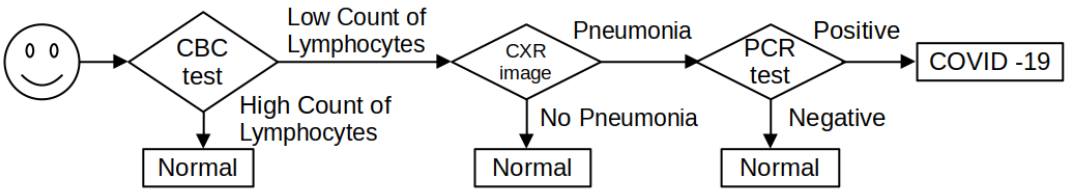


FIGURE 1.2: Typical hospitals routine of reducing COVID-19 spread

advantages over chest CT including the widespread of acquisition devices, low cost, and the speed of the acquisition[11][12][13][14]. These advantages lead CXR to be a fixed routine for hindering COVID-19 spread.

1.2 Motivation

Fig. 1.2 illustrates a typical Egyptian hospital routine for reducing COVID-19 spread among healthcare workers. This process is performed for every visitor to the hospital. Phases of CBC tests and PCR tests are automated and do not consume time. While CXR image diagnosing involves human factors that act as a bottleneck of the process. This process can be fully automated using image classification models. As a consequence *1)* Total time required for every visitor will be reduced. *2)* The load of radiologists is reduced.

Deep learning [15] models, namely convolutional Neural Network (CNN) [16], have shown a great performance in computer vision problems such as object detection [17][18][19][20] and object recognition [21][22]. CNN is initially introduced in [16]. CNN is based on hierarchical learning of the convolutional kernels which are organized in layers [23]. The function of lower-order layers is

to learn low-level features such as edges and corner points [24]. Higher order layers learn high-level features such as objects [24]. Typical CNN architecture is several convolutional layers that are connected sequentially [21]. This sequential connection of the layers does not scale well for deep CNN [22]. CNN shows great performance when it has a large number of layers [22]. To allow CNN to take advantage of the deep architecture, residual connections are used [22]. Residual connections solve the vanishing gradient problem when gradients approach zeros. Also, residual connections allow the reuse of earlier features [25]. Layers with residual connections are easier to optimize than plain layers as they can easily approximate the identity mapping [22]. In this thesis, CNN is used to automate and accelerate the diagnosing pipeline of COVID-19.

1.3 Detection Challenges

Like many computer vision problems, COVID-19 has the following challenges

- **Availability of Data.** CNN is a deep learning technique that requires a large number of training Images. However, the current COVID-19 dataset are small-size dataset which is challenging to train large networks with.
- **Scale.** Like many computer vision models CNN is a scale variant. CNN can not recognize images with scales different from the scales it trained on.
- **Vanishing Gradient Problem.** Deep CNN suffers a vanishing gradient which prevents CNN parameters from being updated.
- **Exploding Gradient Problem.** Deep CNN suffers Exploding gradient which makes CNN diverge.

Proposed Systems try to overcome these problems as will be detailed in chapters 4 and 5.

1.4 Thesis Objective and Contributions

The objective of this thesis is to improve classification accuracy and reduce the computational complexity of detecting COVID-19 cases from CXR Images. Contributions of this thesis are as follows.

- *Lightweight classification model.* A Lightweight classification model with a very low parameter number is proposed for the classification of the CXR

images. It reduces the computational complexity which allows deployment on mobile devices and saves bandwidth of the network during the model distribution process.

- *Scale invariant model.* Pneumonia scales in CXR are not uniformly distributed which prevents CNN from recognizing infrequent scales. Scale-invariant model is proposed for the detection of COVID-19 pneumonia at different scales.
- *High Detection accuracy.* Proposed systems provide superior performance according to various classification metrics.

1.5 Thesis Organization

This thesis is organized as follows:

- **Chapter 2**: includes required Background to understand the Thesis.
- **Chapter 3**: includes and illustrates the recent and related work in the COVID-19 detection literature.
- **Chapter 4**: presents the proposed work I which presents a lightweight classification model.
- **Chapter 5**: presents the proposed work II which includes the scale-invariant model for COVID-19 classification.
- **Chapter 6**: illustrates the experimental results for both proposed work I and II and quantitative analysis of the proposed work I and II is provided.
- **Chapter 7**: concludes the thesis and provides planning for the future work as an extension of the proposed approach

Chapter 2

Background

This chapter includes the required background to understand the thesis proposal presented in the following chapters.

2.1 Convolutional Neural Networks (CNN)

CNN was initially introduced by LeCun [16] which is based on learning adaptive convolutional kernels. CNN consists of two parts convbase and densebase parts. For the Convbase instead of connecting all of the units in a layer to all the units in a preceding layer, convolutional networks organize each layer into feature maps [16], which can be thought of as parallel planes or channels. In a convolutional layer, the weighted sums are only performed within a small local window *i.e)* receptive field and weights are identical for all pixels, just as in regular shift-invariant image convolution and correlation. This parameter sharing reduces the required total number of parameters and allows learning to shift-invariant convolutional kernels. These convolutional kernels produce equivariant feature maps. Fig. 2.1 represents the typical CNN architecture of LeNet.

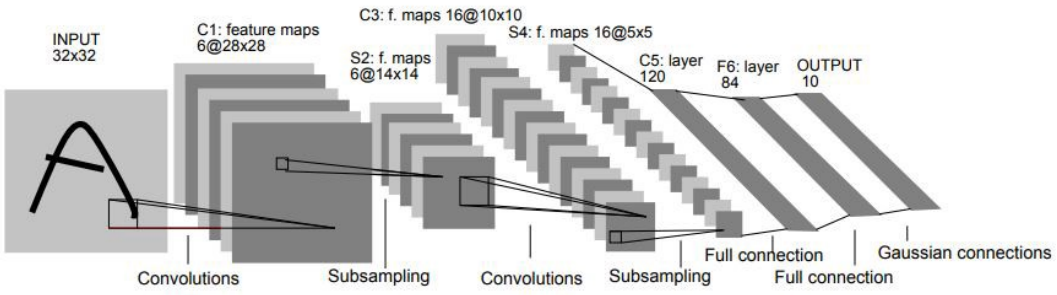


FIGURE 2.1: Typical CNN architecture

2.1.1 Convolutional Layer

The building block of the convolutional layer is the 2D convolutional kernel. Fig. 2.2 illustrates the 2D convolutional layer. Each 2D convolution kernel takes as input all of the C_{i-1} channels in the preceding layer, windowed to a small area, and produces the values in one of the C_i channels in the next layer. For each of the output channels, we have $K^2 \times C_{i-1}$ kernel weights, so the total number of learnable parameters in each convolutional layer is $K^2 \times C_{i-1} \times C_i$. In Fig. 2.2, we have $C_{i-1} = 6$ input channels and $C_i = 4$ output channels, with an $K = 3$ convolution window, for a total of $9 \times 6 \times 4$ learnable weights, shown in the middle column of the figure. Since the convolution is applied at each of the $W \times H$ pixels in a given layer, the amount of computation (multiply-adds) in each forward and backward pass over one sample in a given layer is $W \times H \times K^2 \times C_{i-1} \times C_i$. To fully determine the behavior of a convolutional layer, we still need to specify the following hyperparameter:

- **Padding.** Padding is used to preserve the spatial dimension of the input feature map after the convolution operation is performed. Typically, it is performed by inserting $\lfloor K/2 \rfloor$ columns for both sides and $\lfloor K/2 \rfloor$ rows to the top and bottom.

- **Stride.** Stride is the step taken between two centers when performing the convolution operation. Typically, Stride is equal to 1. Stride can act as a downsampling operation that can be performed instead of the pooling operation.

- **Dilation.** Dilation is a technique that enlarges the kernel by inserting zeros between its consecutive elements. As a result, it covers a larger area of the input without increasing the total number of parameters. Which can be described as

$$y[i] = \sum_{k}^{K} x[i + r \times k] w[k]$$

  where x is the input signal, w is the convolutional filter with a size of K, y is the resultant signal, and r is the dilation rate.

Generally, each convolutional layer computes some activation based on its input and a nonlinear function. The activation function used in each layer has a great effect on the modeling of the problem and also the semantic assigned to the output of some units is affected by this choice. The most important activation functions will be introduced in the following section.

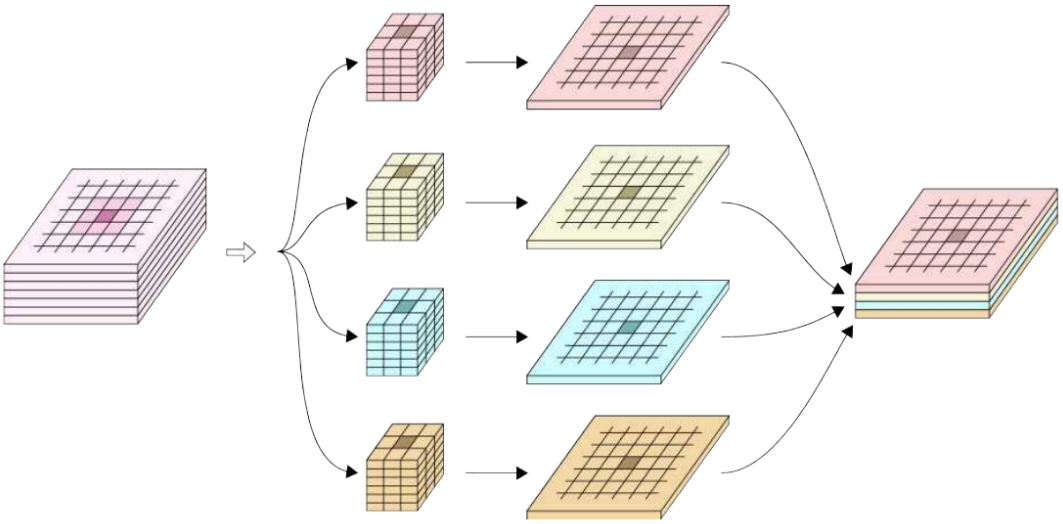


FIGURE 2.2: Convolutional layer with single input feature map and four convolutional kernels

2.2 Activation functions

The activation function is one of the most important components of a CNN. To tackle non-linearly separable problems it is imperative to map the input into a linearly separable space. The activation function does this by performing an *element-wise nonlinear transformation* of the pre-activation that comes from the linear combination of the convolutional layer.

The linear combination of the convolutional layer and the nonlinearity work together closely: the latter is usually fixed and does not evolve during training, but maps its input to a highly non-linear space; the former, is determined by the learned weights which are learned during the training process and uses the activation function to map the calculated activation into a new space where they are simpler and easier to separate. It is interesting to point out that N consecutive linear combination is a single linear combination. The activation function breaks this property and introduces deeper networks.

In the following subsection common activation function is presented.

2.2.1 Sigmoid

The sigmoid, often called *logistic*, is a differentiable monotonically increasing function that takes any real-valued number and maps it to $[0, 1]$. As illustrated in its representation in Fig. 2.3, for large negative numbers it approaches 0. However, for large positive numbers, it approaches 1. It is defined as

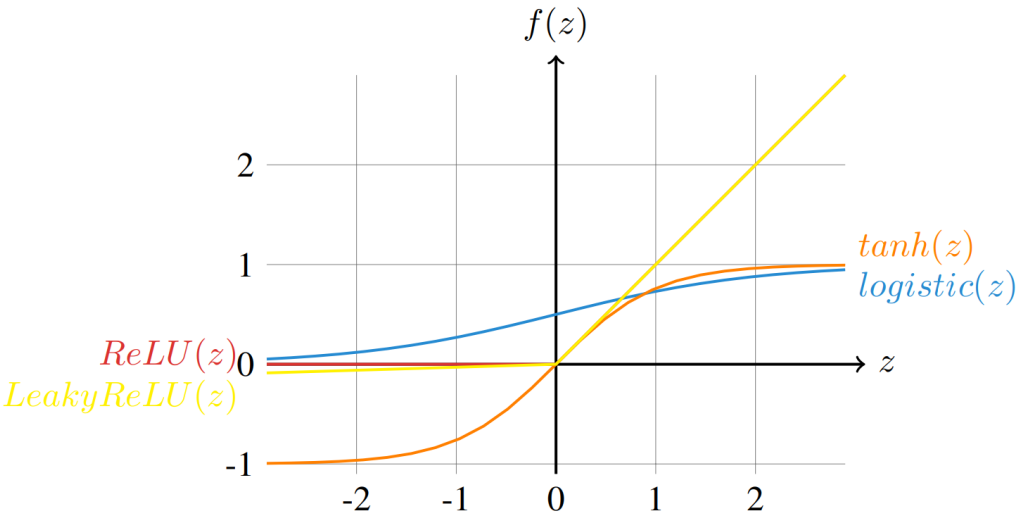


FIGURE 2.3: Some of the most common activation functions: sigmoid, tanh, ReLU, and Leaky ReLU. ReLU and Leaky ReLU are overlapping for $z \geq 0$.

$$logistic(\mathbf{z}) = \frac{1}{1+\exp(-\mathbf{z})}. \tag{2.1}$$

The logistic function is the most used nonlinearity historically due to its possible interpretation as the firing probability of a neuron given its activation: when the activation is low the neuron fires less often whereas when the activation is high the frequency of the spikes increases.

Another very important property of the logistic function is that it is a very simple and fast derivative computation such as follows:

$$\begin{aligned}
\frac{\partial}{\partial \mathbf{z}} logistic(\mathbf{z}) &= \frac{\exp(-\mathbf{z})}{(1+\exp(-\mathbf{z}))^2}, \\
&= \frac{1}{1+\exp(-\mathbf{z})} \cdot \frac{\exp(-\mathbf{z})}{1+\exp(-\mathbf{z})}, \\
&= logistic(\mathbf{z}) \cdot \frac{\exp(-\mathbf{z})}{1+\exp(-\mathbf{z})}, \\
&= logistic(\mathbf{z}) \cdot \frac{1+\exp(-\mathbf{z})-1}{1+\exp(-\mathbf{z})}, \\
&= logistic(\mathbf{z}) \cdot \left(1 - \frac{1}{1+\exp(-\mathbf{z})}\right), \\
&= logistic(\mathbf{z}) \cdot (1 - logistic(\mathbf{z})).
\end{aligned} \tag{2.2}$$

It becomes out of favor due to the following major drawbacks:

- *Saturation region which weakens the gradient propagation:* The backpropagation algorithm exploits the gradient of the loss function to update the parameters of the network. However, sigmoid have a derivative of zero at both ends which saturates the training. This problem – often referred to as *vanishing gradient problem* – makes training very slow or prevents it in some cases. As a result, sigmoid requires a very careful initialization of the weights of the network.

- *The output is not zero-centered:* according to [26] normalized activation (i.e., activation with zeros mean and unit variance) can accelerate the training. However, the sigmoid always has non-negative activation as a result non a non-zero-center mean which slows down the training.

2.2.2 Hyperbolic tangent (tanh)

The hyperbolic tangent, *tanh*, is a differentiable monotonically increasing function that maps any real-valued number to $[-1, 1]$. This nonlinearity has the same problems as sigmoid except its activation is zeros-center.

$$tanh(\mathbf{z}) = \frac{1 - exp(-2\mathbf{z})}{1 + exp(-2\mathbf{z})}. \tag{2.3}$$

2.2.3 Rectified Linear Unit (ReLU)

Rectified Linear Unit (ReLU) is introduced in [23]. It has become the nonlinearity of choice in many applications [23][22]. It is defined as

$$relu(\mathbf{z}) = max(0, \mathbf{z}). \tag{2.4}$$

Although very simple, it has some very interesting properties[27] as follows:

- *No positive saturation:* the ReLU does not respond, or saturate, for non-positive inputs, but does not otherwise. This ensures a flow of gradient, and update signal, whenever the input is non-negative, which was found to significantly speed up the convergence of training.

- *Cheap to compute:* Unlike many other activation functions that require expensive computation, such as exponential function, ReLU's implementation simply a threshold at zero. Another important characteristic is

that the gradient is trivial to compute:

$$\nabla(relu(\mathbf{z}^{(l)})) = \begin{cases} \mathbf{a}^{(l-1)}, & \text{if } \mathbf{z}^{(l)} > 0, \\ 0, & \text{if } \mathbf{z}^{(l)} < 0, \\ undefined, & \text{if } \mathbf{z}^{(l)} = 0. \end{cases} \tag{2.5}$$

- *Induce sparsity:* ReLU units induce sparsity, whenever the input is negative their activation is zero. Sparsity is a desired property: as opposed to dense encoding, sparsity will produce representations where only a few entries change upon small variations of the input, i.e., it will produce a representation that is more consistent and robust to perturbations. Furthermore, sparsity allows compact encoding, which is desirable in many contexts such as, e.g., data compression and efficient data transfer. Finally, it is also usually easier to linearly separate sparse representations [28].
- *ReLU units can die:* ReLU does not restrict the gradient flow from the positive part. The large gradient can update the weight in a such way it can not activate again, i.e., it always produces a negative value. This problem can be partially solved with some ReLU variant such as leaky ReLU or a parametric ReLU.

2.2.4 Leaky Rectified Linear Unit (Leaky ReLU)

Leaky ReLUs have been proposed as a way to mitigate the saturated units of ReLUs caused by extreme updates, by preventing the unit from having zero gradient thus allowing a gradient to flow through the unit, potentially recovering extreme values of the weights. Leaky ReLUs are widely adapted and defined as follows:

$$leaky\_relu(\mathbf{z}) = max(\beta * \mathbf{z}, \mathbf{z}), \tag{2.6}$$

where β is a small constant.

2.2.5 Softmax

Unlike previously mentioned functions softmax differs in that it does depend on all the values of the dimensions altogether to produce the categorical distribution of the over N classes. It is defined as follows:

$$softmax(z_i) = \frac{\exp(z_i)}{\sum_{k=0}^{K} \exp(z_k)}, \tag{2.7}$$

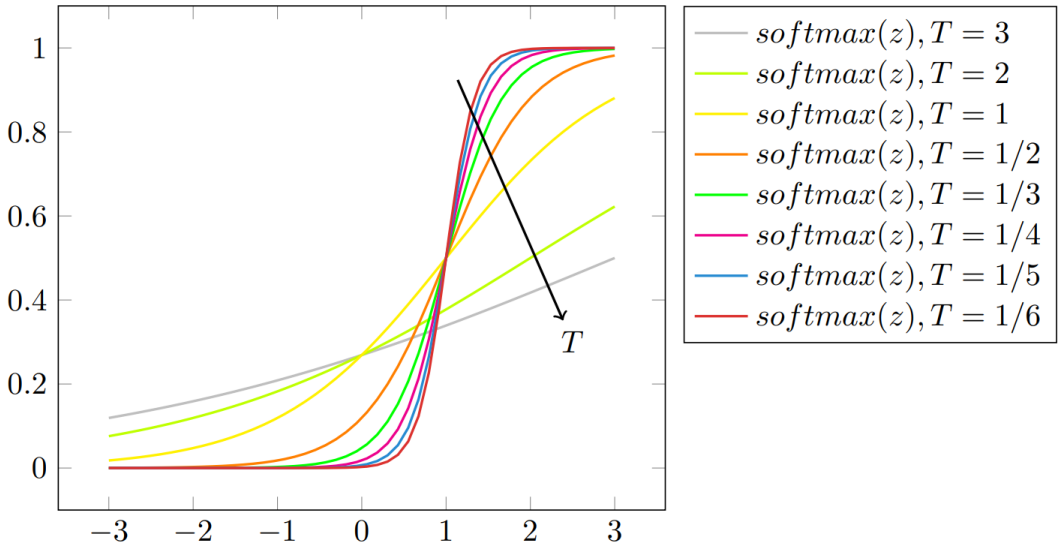


FIGURE 2.4: The steepness of softmax function as temperature T grows.

where K is the number of classes, i.e., of dimensions (or neurons).

Temperature parameter T can be used with the softmax which controls its steepness (see Fig. 2.4), i.e., to manage the randomness of predictions. High temperature, case of $T = \inf$, produces a uniform categorical distribution. While a small temperature produces a peaked probability distribution for the larger value.

$$softmax(z_i) = \frac{\exp(z_i/T)}{\sum_{k=0}^{K} \exp(z_k/T)}. \tag{2.8}$$

2.3 Pooling

In addition to convolutional, *pooling* operations are an important building block in CNNs. Pooling operations reduce the spatial size of feature maps by using some aggregation, i.e.) max or average, function to summarize a particular region. The following properties affect the output size o_j of a pooling layer along axis j:

- i_j: input size along axis j,
- k_j: pooling window size along axis j,
- s_j: stride (distance between two consecutive positions of the pooling window) along axis j.

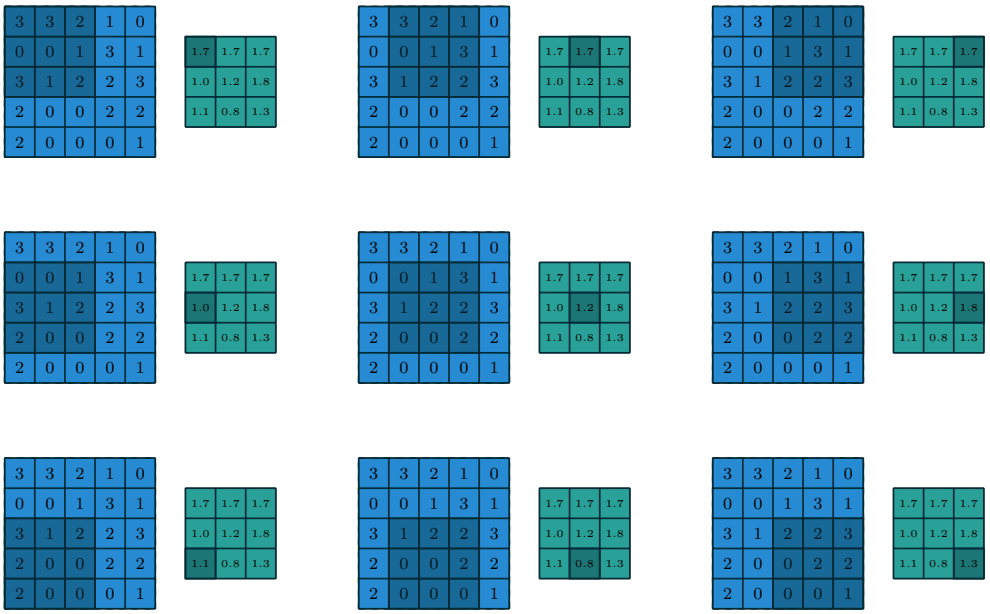


FIGURE 2.5: Computing the output values of a 3×3 average pooling operation on a 5×5 input using 1×1 strides.

2.3.1 Average Pooling

Average pooling is performed by sliding a window over the input feature map and performing and averaging the content of the window. Fig. 2.5 provides an example of average pooling.

2.3.2 Max pooling

Max pooling is performed the same way as average pooling, but instead of performing the averaging as an aggregation function, it performs a max function. Fig. 2.6 provides an example of average pooling.

CNN is used for feature extraction from images followed by fully connected layers for classification. As convolution operation is applied in a sliding window fashion it can accept input of varied size, resulting in a varied size output. CNN is followed by fully connected layers which can accept input of fixed size. This makes CNN incapable of accepting varied-size inputs. Thus images are first reshaped into some specific dimension before feeding into CNN. This creates another issue of image warping and reduced resolution. Spatial Pyramid pooling comes as a counter to this problem.

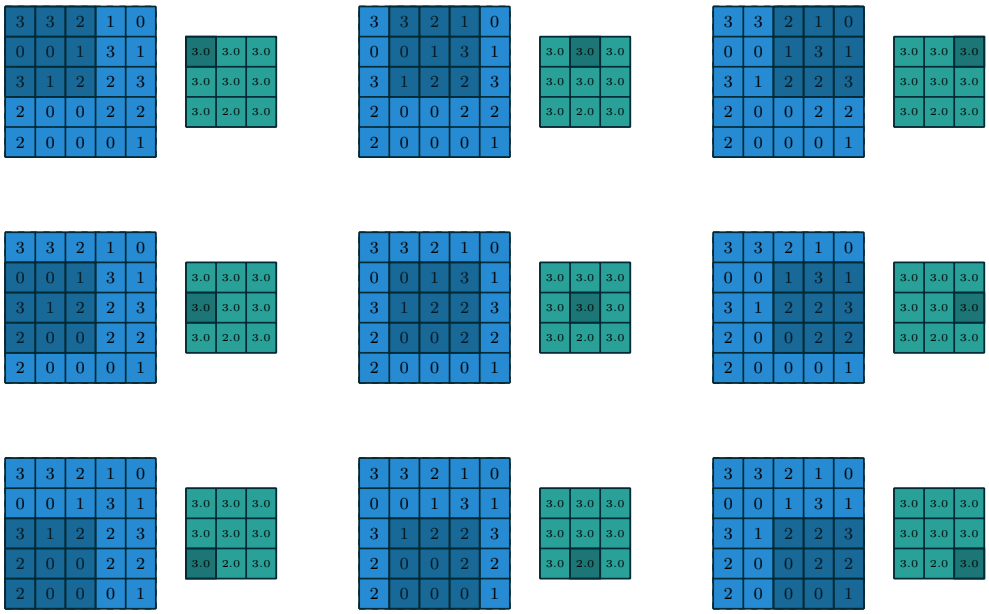


FIGURE 2.6: Computing the output values of a 3×3 max pooling operation on a 5×5 input using 1×1 strides.

2.4 Spatial Pyramid Pooling

The convolutional layers process arbitrary variable lengths, and also they produce outputs of variable sizes. Classifiers like SVM, decision tree, or fully-connected layers require fixed-length input feature vectors. Such vectors can be generated by the Bag-of-Words (BoW) approach [29] that pools the features together. Spatial pyramid pooling [30], [31] improves BoW in that it can preserve the spatial information of the input by pooling local spatial bins. These spatial bins have sizes that are proportional to the input image, so the number of bins is fixed regardless of the input dimensions. While sliding window pooling of the previous deep networks [23] the number of sliding windows depends on the input size. To adopt the deep network for images of arbitrary sizes, we replace the last pooling layer with a spatial pyramid pooling layer. Fig. 2.7 illustrates our method. In each spatial bin, we pool the responses of each filter. The outputs of the spatial pyramid pooling are kM-dimensional vectors with the number of bins denoted as M (k is the number of filters in the last convolutional layer). The fixed-dimensional vectors are the input to the fully-connected layer. With spatial pyramid pooling, the input image can be of any size.

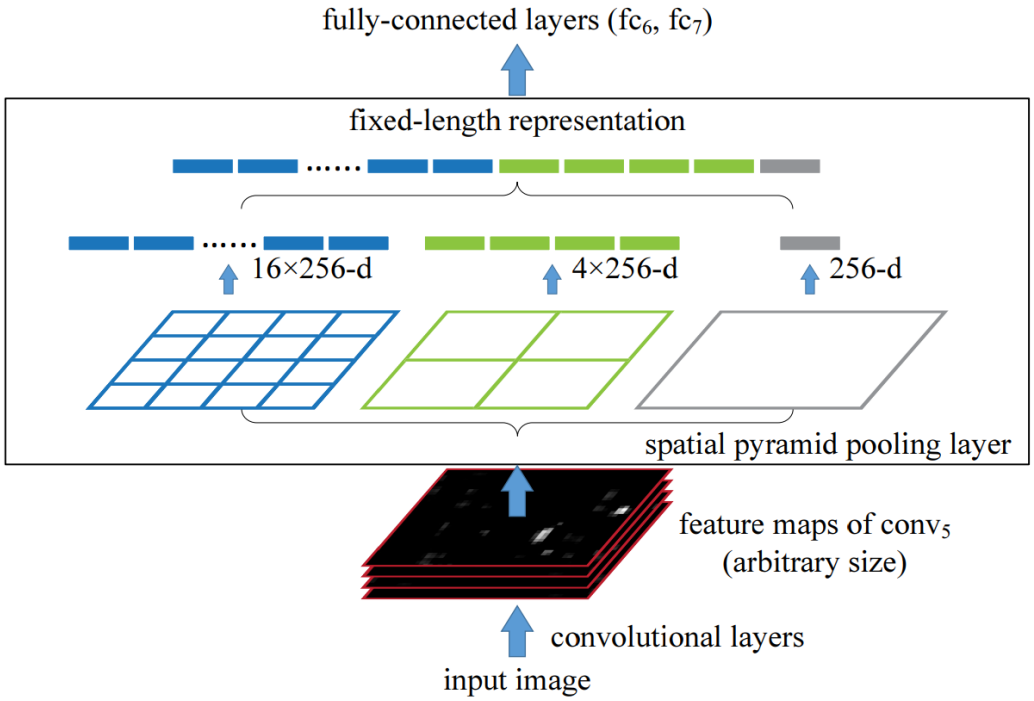


FIGURE 2.7: spatial pyramid pooling layer. Input feature map is divided into pins for each pin an aggregation function is performed

2.5 Neural Network Attention

In a vision system, an attention mechanism can be defined as a dynamic selection process that is realized by weighting features according to the importance of the input in an adaptive manner. Attention attempts to selectively concentrate on relevant information or features while ignoring other irrelevant ones. Currently, attention-based models have shown great success in natural language processing [32] and computer vision tasks [33]. Attention modules guide the network to focus on the most relevant features only [33]. Attention has many categories such as channel attention and spatial attention. Stated as follows

2.5.1 Channel Attention

Internal CNN feature maps have different channels in different usually each channel represents different objects [34]. Channel attention recalibrates the weight of each channel, and can be viewed as an object selection process, thus determining what to pay attention to. [35] is a pioneering channel attention

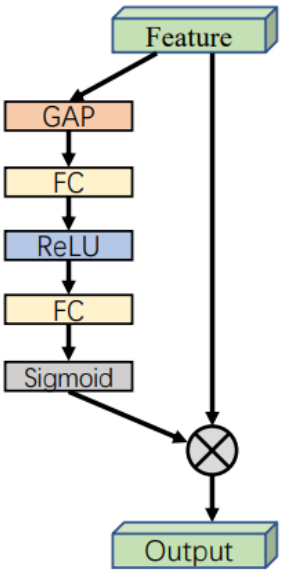


FIGURE 2.8: Channel attention block of SE network

mechanism that is formulated as follows

$$\begin{aligned} Att(X) &= \sigma(W_2 ReLU(W_1 GAP(X))) \\ Y &= Att(X)X \end{aligned} \tag{2.9}$$

where σ is sigmoid activation function, GAP is global average pooling. Fig. 2.8 illustrates the channel attention mechanism of [35]

2.5.2 Spatial Attention

Spatial attention can be considered as an adaptive spatial region selection mechanism: where to pay attention.

In [36] generates a spatial attention map by utilizing the spatial relationship of features. Unlike channel attention, spatial attention focuses on where there is an informative part, which is complementary to channel attention. They compute the spatial attention by first applying average-pooling and max-pooling operations over each channel and concatenating them to generate the feature descriptor. After the pooled-feature concatenation convolution layer is applied to generate a spatial attention map $\mathbf{M}_s(F) \in \mathcal{R}^{HW}$ which encodes the spatial importance. We aggregate channel information of a feature map by using two pooling operations, generating two 2D maps: $\mathbf{F}^s_{avg} \in \mathbb{R}^{1\times H\times W}$ and $\mathbf{F}^s_{max} \in \mathbb{R}^{1\times H\times W}$. Each denotes average-pooled features and max-pooled features across the channel. Those are then concatenated and convolved by a standard convolution layer, producing the 2D spatial attention map. In short,

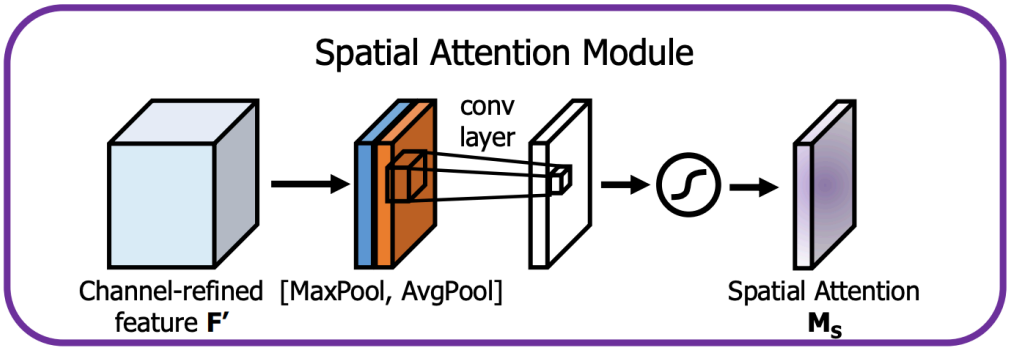


FIGURE 2.9: Spatial attention mechanism

the spatial attention is computed as:

$$\begin{aligned} &\mathbf{M}_s(F) = \sigma\left(CONV_{7\times7}\left(\left[\text{AvgPool}(F); \text{MaxPool}(F)\right]\right)\right) \\ &OR \\ &\mathbf{M}_s(F) = \sigma\left(CONV_{7\times7}\left(\left[\mathbf{F}^s_{avg}; \mathbf{F}^s_{max}\right]\right)\right) \end{aligned} \tag{2.10}$$

where σ denotes the sigmoid function, $CONV_{7\times7}$ represents a convolution operation with the filter size of 7×7, and $[\cdot]$ is a concatenation operator.

2.6 Normalization

Normalization is crucial for training a deep network. Normalization allows the use of a larger learning rate and results in a smoother loss landscape of the objective function. Different types are summarized as follows:

2.6.1 Batch Normalization

As proposed in [26] batch normalization is reducing *Internal Covariate Shift* (ICS). ICS is defined as the change in the distribution of network activations due to the change in network parameters during training. Batch Normalization (BN) mitigates the ICS by normalizing the internal feature maps, minibatch, to have zero mean and unit variance. BN also allows the CNN to undo the normalization by scaling and shifting the normalized features using γ, β.

BN is applied to each layer for a minibatch $\mathcal{B}$ as follows:

$$\begin{aligned}
\mu_{\mathcal{B}} &= \frac{1}{m}\sum_{i=1}^{m} x_i \\
\sigma_{\mathcal{B}}^2 &= \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2 \\
\hat{x}_i &= \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \\
y_i &= \gamma \hat{x}_i + \beta = \mathrm{BN}_{\gamma,\beta}(x_i)
\end{aligned} \tag{2.11}$$

BN has many benefits which are summarized as follows:

- **Accelerate network training.** BN allows a larger learning rate.
- **Regularization effect.** BN has a regularization effect due to batch construction being done stochastically.
- **Mitigate the effect of the saturating activation function.** Normalization performed by BN relocate the layer activation to the linear regime of saturating activation function such as sigmoid.

2.6.2 Layer Normalization

Unlike batch normalization, Layer Normalization [37] directly estimates the normalization statistics from the summed inputs to the neurons within a hidden layer so the normalization does not introduce any new dependencies between training samples.

We compute the layer normalization statistics over all the hidden units in the same layer as follows:

$$\begin{aligned}
\mu^l &= \frac{1}{H}\sum_{i=1}^{H} a_i^l \\
\sigma^l &= \sqrt{\frac{1}{H}\sum_{i=1}^{H} \left(a_i^l - \mu^l\right)^2}
\end{aligned} \tag{2.12}$$

where H denotes the number of hidden units in a layer. Under layer normalization, all the hidden units in a layer share the same normalization terms μ and σ, but different training samples have different normalization terms. Unlike batch normalization, layer normalization does not impose any constraint on the size

FIGURE 2.10: Data augmentation using crop augmentation with different preserving degrees and the corresponding accuracy of recognizing certain classes using ResNet

of the mini-batch and it can be used in the pure online regime with a batch size of 1 sample.

2.7 Augmentation

Data augmentation [38] is a technique that is used for enlarging the training set, based on different modifications, using label-preserving transformation. Data augmentation not only helps to grow the dataset but also increases the diversity of the dataset and introduces robustness to these transformations. When training machine learning models, data augmentation acts as a regularizer and helps to avoid overfitting.

Data augmentation techniques have been found useful in domains like NLP and computer vision. In computer vision, transformations like cropping, flipping, and rotation are used. Fig. 2.10 represents pre-class performance to the degree of augmentation.

2.8 Dropout

Dropout [39] is a regularization technique for neural networks that drops a unit (along with connections) at training time with a specified probability p (a common value is $p = 0.5$). At test time, all units are present, but with weights scaled by p (i.e. w becomes pw). Dropout has the following benefits.

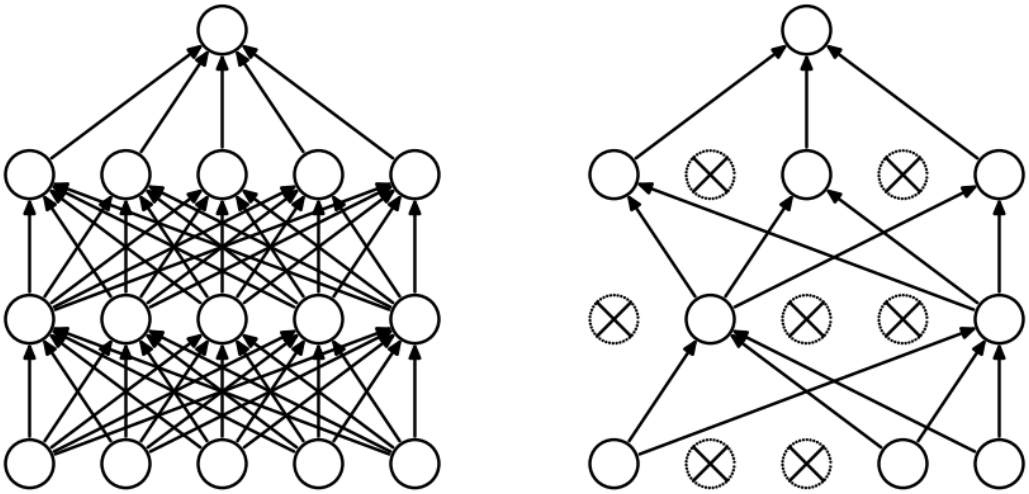


FIGURE 2.11: Dropout Neural network Model. **Left**: A standard network with two hidden layers. **Right**: An example of a thinned net produced by applying dropout to the network on the left. Crossed units have been dropped.

- **Reduces co-adaptation between the neurons.** Complex co-adaptation between neurons occurs when the neurons of one layer depend on the neurons on the next layers to correct their errors. Dropout fixes this issue by making neuron existence stochastic.
- **Implicit ensemble.** Dropout approximately combining exponential number of different thinned neural network architectures efficiently

Fig. 2.11 illustrates the operation of the Dropout.

2.9 Multiscale Recognition

CNN, like many computer vision models, is a scale-variant [40] model such that it cannot recognize objects at various scales unless it is explicitly trained to recognize such objects. Many approaches have been developed to overcome this problem. The first approach is referred to as *shared-net*. The shared-net approach creates a scale pyramid of the input image to train a single shared network using multiple scales. This shared network produces a feature vector for each scale which in turn is fused with an aggregation function to produce the final prediction [41][42][43][44]. Shared-network needs to evaluate each scale independently which is time-consuming.

The second approach is to reuse low-level features produced by the intermediate CNN layers using a skip connection [45][46]. These features are considered

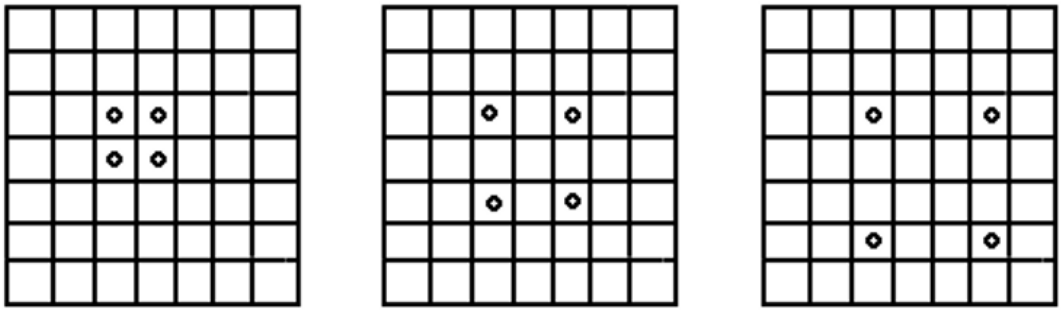


FIGURE 2.12: A grid represents a feature map and the circle inside the cell represents the corresponding value of 2×2 convolutional kernel. **left**: A 2×2 convolutional kernel with an atrous rate $r = 1$. **middle**: A 2×2 convolutional kernel with an atrous rate $r = 2$. **right**: A 2×2 convolutional kernel with an atrous rate $r = 3$.

multi-scale due to the different receptive fields of the corresponding layers. The training of these networks is performed through two phases. A backbone network is trained in the first phase. Then, it is fine-tuned during multi-scale feature extraction [47][48]. The drawback of this approach is the separation between the classifier training and the feature extraction.

The third and the recent approach is to deploy atrous convolution [49] in CNN context [50][51]. Atrous convolution is defined as follows:

$$y[i] = \sum_{k}^{K} x[i + r \times k]w[k]$$

where x is the input signal, w is the convolutional filter with a size of K, y is the resultant signal, and r is the atrous rate. Atrous convolution can be seen as a convolution operation between the input signal and the upsampled version of the filter. This upsampling is performed by introducing $r - 1$ zeros between the kernel values. Also, Atrous convolution can be seen as a convolution between the downsampled version of the input signal and the kernel. Atrous convolution allows the change in the receptive field and controls the input signal resolution without increasing the parameter number [52] (*i.e,* a 3×3 convolutional kernel and a dilation rate of $r = 2$ has the same receptive field of 5×5 convolutional kernel and dilation rate of $r = 1$). Atrous convolution can be used to build the resultant scale-space through performing Atrous convolution with multiple Atrous rates, r, which is known as Atrous Spatial pyramid Pooling (ASPP) [50].

ASPP has a regularization effect as it ensures that the convolutional kernels will learn useful features across the scale space of the input. Fig. 2.12 shows an example of Atrous convolution with different rates.

2.10 Summary

This chapter provides an introduction to several key concepts and techniques in the field of deep learning, with a focus on convolutional neural networks (CNNs) and their applications in computer vision.

The chapter begins by introducing the basic architecture of a CNN, including convolutional layers and pooling layers. It then discusses the role of activation functions in deep learning, such as sigmoid, tanh, and rectified linear units (ReLU), and their impact on model performance.

The chapter goes on to explore more advanced techniques in CNNs, including spatial pyramid pooling, which enables the network to learn features at multiple scales. It also covers channel attention and spatial attention mechanisms, which allow the network to selectively focus on certain features during training and inference.

The chapter then discusses the importance of normalization techniques in deep learning, including batch normalization and layer normalization, which help to stabilize the network during training and improve its generalization performance.

The chapter concludes with a discussion of various data augmentation techniques, such as image rotation and flipping, as well as dropout, which can help to prevent overfitting in the network. Finally, the chapter introduces the concept of multiscale recognition, which involves training the network to recognize objects at different scales, and its importance in achieving high accuracy in computer vision tasks.

Overall, this chapter provides a comprehensive introduction to the key concepts and techniques in deep learning, with a focus on CNNs and their applications in computer vision to understand this thesis.

Chapter 3

Literature Review

CNN outperformed traditional computer vision approaches for medical image classification due to its current advances in architecture design. Typically, CNN architectures are composed of two parts [23]. The first part is called a convolutional part which further consists of convolutional layers interconnected with some connection patterns that are responsible for feature extraction. The second part is the fully connected layers which consist of Dense layers that are responsible for classification. This section discusses previous CNN-based CXR classification systems reporting the modeling mechanisms and the reported results.

3.1 CNN Based Models

In this section, CNN-based models for COVID-19 detection are reviewed.

3.1.1 Reliable COVID-19 Detection Using Chest X-ray Images

In [53], authors have adapted transfer learning techniques to train CNN models. Training is performed through two phases. Fig. 3.1 represents the proposed system by [53]. In phase I, u-shaped architecture is excluded by removing the skip connections, which perform the concatenation operation. The reason for constructing an encoder-decoder network without skip connections is that the contributions from the initial layers are avoided; therefore, the network can make decisions from the high-level features that are closer to the segmentation mapping of the input image. In phase II, the decoder network is discarded, and the encoder network is fine-tuned for the binary classification task. Adam

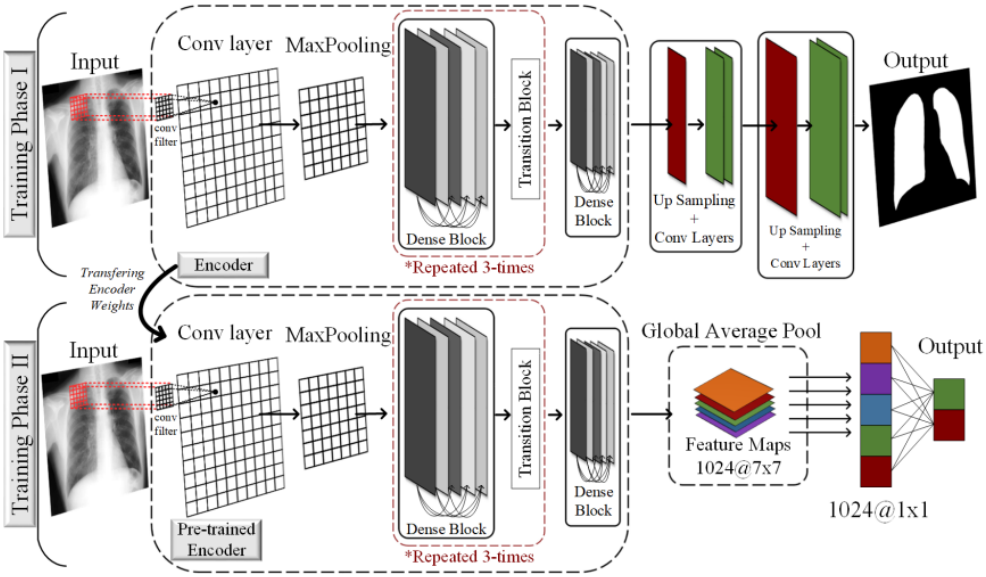


FIGURE 3.1: ReCovNet model proposed by [53]. It uses a pre-trained encoder for the classification of COVID-19 CXR image

optimizer is used to train their network. The model is trained with the QaTa-COV19 dataset. The method achieved a performance of 98.57% for sensitivity and 99.77% for specificity.

3.1.2 Advance Warning Methodologies for COVID-19 Using Chest X-Ray Images

In [54], authors evaluated the performance of different classifiers for classifying the feature produced by DenseNet121. Their methodology is as follows:

Feature Extraction using DensNet121

They pretrained two DenseNet121 [25] models on Early-QaTa-COV19 and ChestX-ray14 datasets are used to extract 1024-D feature vectors by taking the output after global pooling just before the classification layer. Then, a dimensionality reduction is applied over the calculated features with principal component analysis (PCA) by choosing the first 512 principal components.

Representation-based classification

Representation-based classification (RC) techniques are used in many different classification tasks such as face recognition in [55], hyperspectral image classification [56], and human action recognition [57]. The authors classified the

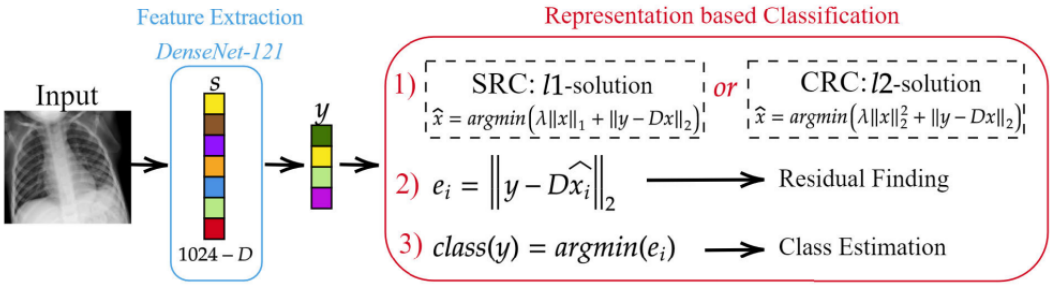


FIGURE 3.2: sparse and Collaborative Representation learning pipeline of [54]

feature vector produced by DenseNet121 using either Sparse RC or Collaborative RC. It is performed as follows:

$$\hat{x} = \arg\min_{x}(\lambda||x||_1 + ||y - Dx||_2) \tag{3.1}$$

where D is the dictionary of features produced by DenseNet121 and projected using PCA. Sparse RC minimizes ℓ^1 of $\hat{x}$. While Collaborative RC minimizes ℓ^2 of $\hat{x}$ as follows:

$$\hat{x} = \arg\min_{x}(\lambda||x||_2 + ||y - Dx||_2) \tag{3.2}$$

For both types of representations $\hat{x}$ reconstruction error is calculated as follows:

$$e_i = ||y - D\hat{x}||_2 \tag{3.3}$$

Then assign a class of the lower construction error as follows:

$$class(y) \arg\min(e_i) \tag{3.4}$$

Fig. 3.2 represents the classification pipeline presented in [54]. They achieved a .98 and .97 accuracy for Sparse Representation-based classification and Collaborative Representation-based classification, respectively. Authors of [54] also evaluated SVM for classifying features produced by DenseNet121 and recorded .98 of accuracy.

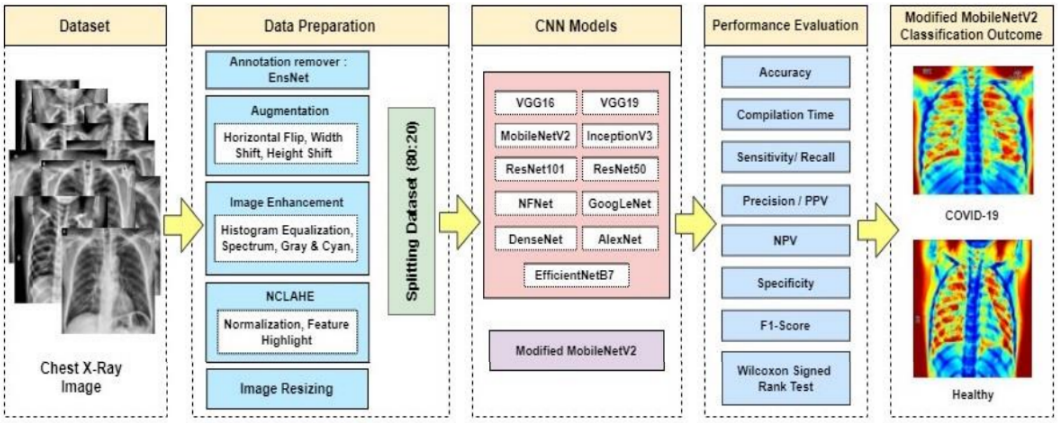


FIGURE 3.3: Pipeline proposed in [58] for COVID-19 classification

3.1.3 COVID-19 Detection Using DL Algorithm on CXR Images

In [58], authors proposed a modified MobileNetV2 CNN model where the stander convolution is replaced by a depth-wise convolution. Their model is trained with 3616 COVID-19 CXR images and 10,192 normal CXR images. The dataset is initially preprocessed by reshaping input images to 299×299 and an image enhancement technique is applied. The model achieved an accuracy of 98% for the binary classification task. Fig. 3.3 illustrates general pipeline proposed by [58] which is detailed as follows:

The preprocessing phase of [58] initially performs image augmentations which include horizontal flip, rotation, width shift, and height shift on all the extracted data from the original dataset. Also, image enhancement applied Histogram equalization, Spectrum, Grays, and Cyan. The N-CLAHE algorithm was then used to normalize pictures and highlight smaller features for machine learning classifiers to notice. Thereafter, the images were resized to the classifier's standard resolution. After resizing the picture, the machine learning classifier used the enhanced (52,000) images in a ratio of 80% data for training, whereas 20% was used for testing. They proposed modified MobileNetV2 network and achieved 98% of accuracy

In [59] authors have proposed a hybrid model for multilabel classification where VGG16 is used as a feature extractor. Their system is trained by a combined dataset of QaTaCov19 and Chest X-Ray achieving an accuracy of 91.09%.

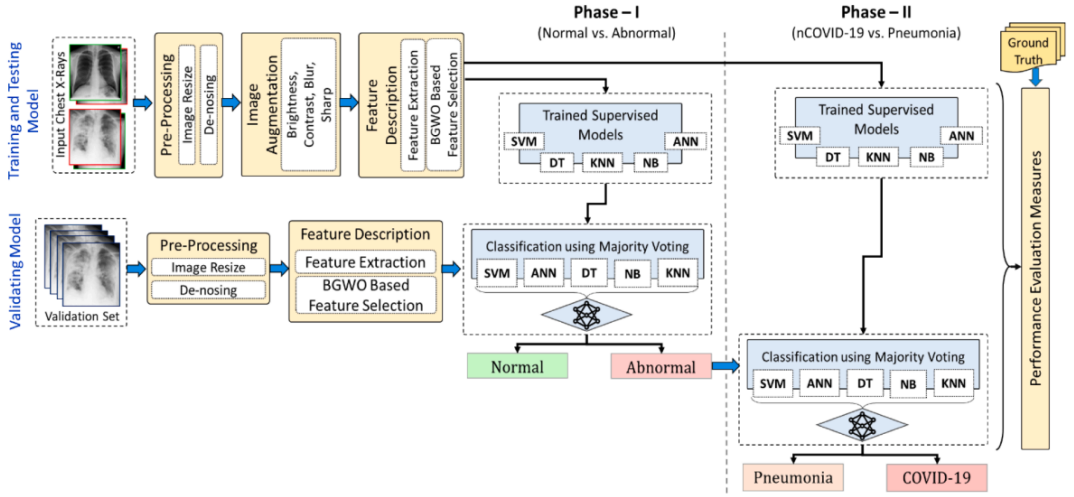


FIGURE 3.4: Proposed machine learning pipeline of [60]

3.2 Traditional Computer Vision Models

3.2.1 COVID-19 detection in CXR images using majority voting based classifier ensemble

In [60], authors proposed a two-phase COVID-19 multi-label classification system as illustrated in Fig. 3.4. The system phase I classifies the input CXR images as normal or COVID-19. In case the image is classified as COVID-19, phase II further classifies this image as pneumonia or COVID-19. Phase I and Phase II have the same structure with a total of 8196 local features are extracted and then classified using the ensemble module. The base classifier of the ensemble module consists of NB, ANN, DT, and SVM. A majority voting is used to combine the predictions of these classifiers. Their system achieved accuracy of 98.062% and 91.329% for Phase I and Phase II, respectively. Both Phases have the same anatomy of preprocessing and feature Description. Feature extraction is performed by extracting a total of 8196 features as follows:

- 8 features using FOSF [61].
- 88 feature using GLCM [62].
- 8100 feature using HOG [63].

binary grey wolf optimization (BGWO) [64] is to select the most relevant features from the previously extracted features. The final prediction of the set is the majority vote of seven benchmark classifiers (ANN, KNN, NB, DT, SVM (linear kernel, radial basis function (RBF) kernel, and polynomial kernel).

TABLE 3.1: Overview of recent studies for classifying COVID19

Work	#Samples	Classes	Model	Best Performing Model	Performance
[65]	1428	3	VGG19, MobileNetV2, Inception, Xception	MobileNetV2	Acc = 96.78%
[66]	204	2	VGG16 + Resnet50	VGG16 + Resnet50, custom CNN	Acc = 89.2%
[67]	100	2	ResNet50, InceptionV3, and InceptionRes-NetV2	ResNet50	Acc = 98%
[68]	21152	2	CNN	CNN	Acc = 94.64%
[69]	5184	2	ResNet18, ResNet50, SqueezeNet, DenseNet-121	SqueezeNet	Sensitivity = 98%
[70]	13975	2	COVID-CAPS	COVID-CAPS	Acc = 95.7%,
[71]	400	2	VGG16, InceptionResNetV2, ResNet50, DenseNet201	NasNetMobile	Acc = 93.94%
[72]	75	2	VGG19, Xception, ResNetV2, DenseNet201	VGG19, DenseNet	F1 scores = 0.91
[73]	1127	2	Modified Darknet	Modified Darknet	Acc = 98%
[74]	1257	3	Xception	Xception	Acc = 94%
[75]	2356	3	ACoS system	ACoS	Acc = 91.33%
[76]	6100	3	SVM, LR, DT, kNN + VGG16, ResNet50	Mean result	Acc = 98.5%
[77]	1428	2	VGG16	VGG16	Acc = 96%
[78]	79500	3	Grad-CAM	Grad-CAM	Acc = 91.5%
[79]	6200	4	CSEN-based classifier	CSEN-based Classifier	Sensitivity = 98%
[80]	13975	19	COVID-Net	COVID-Net	Acc = 93.3%
[81]	196	3	DeTrac	Detrac	Acc = 93.1%
[82]	3150	3	CapsNet	CapsNet	Acc = 97%
[83]	1127	3	Xception	Xception	Acc = 97%
[84]	7470	2	MD-Conv	MD-Conv	Acc = 93.4%
[85]	380	2	Novel CNN Model	Novel CNN Model	Acc = 91.6%
[86]	247	2	BMO-CRNN	BMO-CRNN	Acc = 97.31%

3.3 Summary

CNN is a powerful computer vision model. It is used intensively for COVID-19 detection. Table 3.1 summarizes recent related work for COVID-19 detection. Despite the high accuracy of current CNN-based CXR classification methods, these methods don't address the problem that CNN is a scale variant model. This problem hinders the recognition of large-scale COVID-19 pneumonia. In this thesis, a novel scale-invariant CNN architecture is proposed for the classification of COVID-19 pneumonia. The proposed architecture deploys Atrous convolution method for learning the scale-invariant features. Then, an attention model is utilized to automatically and internally select at which scale CNN should consider and ignore other scales. The proposed architecture exploits texture augmentation to reduce the overfitting and artificially enlarge the training dataset. The experimental results show that the proposed system outperforms the previous CNN-based CXR classification methods with lower trainable parameter numbers.

Chapter 4

Proposed Methodology I

In this chapter, a novel architecture is introduced for detecting COVID-19 that is designed to be lightweight. The architecture is based on two key components: spatial kernel separability and residual connection. By exploiting spatial kernel separability, the number of training parameters is significantly reduced, making the model more computationally efficient. Additionally, residual connections are used extensively to maintain network stability during the training process and to provide the model with regularization effects that reduce overfitting. This combination of spatial kernel separability and residual connections creates a lightweight architecture that is highly effective at detecting COVID-19. Overall, this chapter provides a valuable contribution to the field of COVID-19 detection by introducing a novel and efficient architecture that can help to identify the disease quickly and accurately.

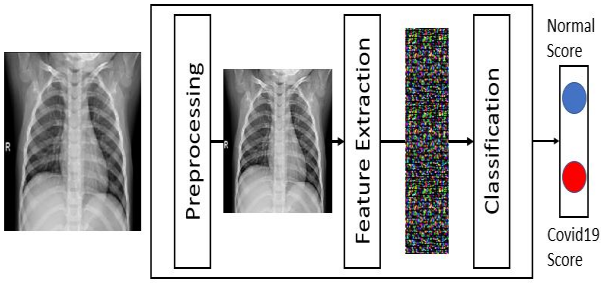


FIGURE 4.1: The phases of the proposed method I.

4.1 Methodology I

In this section, a proposed method I to detect COVID-19 disease from chest X-ray images is presented. The proposed method exploits the CNN model to classify the input chest X-ray image into one of two categories; normal case or

Covid-19 case. The proposed method I consists of three phases: preprocessing, feature extraction, and classification. The proposed method phases are shown in Fig. 4.1.

4.1.1 Preprocessing Phase

The preprocessing phase is responsible for resizing and normalizing the input chest X-ray images. The pre-processing phase is employed to maintain the numerical stability of the model and reduce the co-variance shift [16]. In addition, this phase leads the learning model of the CNN model to reduce the required overhead to adapt to the different scales of different features of the input data. Reshaping size is determined empirically. The input chest X-ray image is re-sized and then adapted and normalized to a normal distribution as follows:

$$Y := \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \tag{4.1}$$

where μ and σ are the mean and standard deviation of chest X-ray image (X), respectively.

After re-sizing the input chest X-ray image, the input image is normalized to have a zero mean and unit standard deviation. Then, the image can be scaled and shifted with a normalization parameter which is determined and adapted by the training dataset during the training process according to the following equation:

$$Z := w_1 Y + w_2 \tag{4.2}$$

where w_1 and w_2 are a trainable parameter.

Unlike the normalization method presented in [26], the batch normalization process presented in this paper has a z-score normalization parameter that is used in both the training and validation phases.

4.1.2 Feature Extraction and Classification

CNN models achieved outstanding success in image recognition [15]. This phase is responsible for extracting spatial features from the normalized chest X-ray image using a tailored CNN model. This phase is based on learning the CNN model by the input preprocessed chest X-ray images. The design of the tailored CNN model is described as follows:

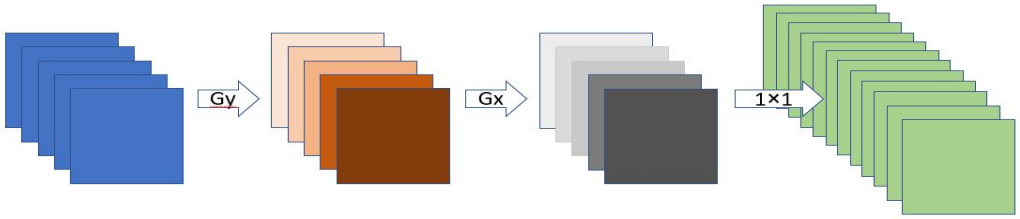


FIGURE 4.2: Separable convolution Gy and Gx have kernel size of $M \times 1$ and $1 \times M$. The combination of these kernels is approximately a $M \times M$ kernel and depth-wise convolution is applied by a 1×1 convolution. The output depth is padded with zeros to have the same spatial size of Gy, Gx. Gy, Gx are performed channel-wise.

Separable CNN kernels

Kernel separability [87] [88] is based on decomposing a 2D convolution kernel to linear combinations of two 1D vectors which leads to a large reduction in the total number of resulting parameters. For example, a 2D kernel of size 9×9 has a total number of $9^2 = 81$ trained parameters. Whereas in the case of separating this 2D kernel to linear combinations of two 1D vectors of sizes 9×1 and 1×9, this results in a total number of $9 + 9 = 18$ trained parameters. As a consequence, kernel separability reduces the number of CNN model operations (such as multiplication and addition). A 2D kernel of $k \times k$ applied for a 2D signal with spatial dimensions of $M \times N$ has a total number of $(N-4)(M-4) \times k^2$ operations but in case of applying kernel separability yields $2(N-4)(M-4)k$ operations. The flow of separated convolution operation is summarized in Fig. 4.2. Fig. 4.3 represents the structure, denoted by the Separated Convolutional Layer, used in the proposed method with kernel size of $(M \times N)$ and satisfying the convolutional kernel separability. Separated Convolutional Layer is composed of three consecutive layers. The first convolutional layer has a kernel size of $(M \times 1)$ and the number of convolutional neurons and filters are equal to the number of channels as the input feature map and the convolution operations are performed channel-wise. The second layer operates in the same way as the first layer but it has a kernel of size $(1 \times M)$. The third layer is the convolutional layer with a kernel of size (1×1) and the number of convolutional neurons is N. The collaboration of the three layers are connected to perform similarly to the convolutional layer with a kernel size of $(M \times M)$ and the number of neuron and filter is the same as N but with a large difference in the performance.

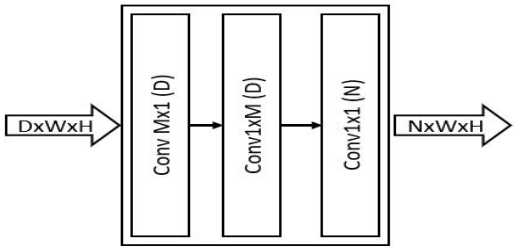


FIGURE 4.3: Separated Convolutional Layer
composed of three consecutive layers. The first Convolutional layer has a kernel size of $(M \times 1)$ and D convolutional neuron. The second layer operates in the same way as the first layer but it has a kernel of size $(1 \times M)$ and D convolutional neuron. The third layer is the convolutional layer with a kernel of size (1×1) and the number of convolutional neurons is N.

Batch Normalization and Activation function

In the proposed method linear separable convolutional kernels are followed by a batch normalization and an activation function. Rectified Linear Unit (ReLU) [27] is a nonlinear activation that allows the network to fit and approximate highly non-linear dataset distribution. The proposed method employs batch normalization which is described in [26].

Batch Normalization [26] reduces internal covariate shift produced as a result of moving between layers during the feedforward procedure [26]. Batch Normalization makes the loss landscape smoother and reduces the number of saddle points [89] which allows to use of higher learning rates. Using a higher learning rate makes the network training faster [26]. Batch normalization reduces the vanishing gradient problem and exploding gradient problem as it makes the resulted activation scale independent from the trainable parameter scale [26]. Batch normalization has the effect of regularization because of the inherited randomness when selecting the batch sample [26] which help the generalization to unseen chest X-ray image.

Deep and larger receptive field Network design

Deeper convolutional neural network design is a very important task for any image recognition task [22]. Training a deeper network is very expensive and has many challenges such as vanishing gradient problem, exploding gradient problem, and degradation problem [22]. The exploding gradient problem occurs when the gradient update becomes very large (approaching infinity) resulting in the network diversion. A vanishing gradient problem occurs when the gradient update becomes very small (approaching zero) resulting in preventing the

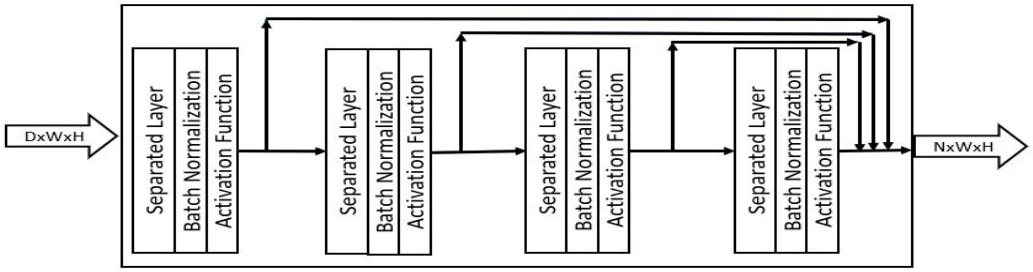


FIGURE 4.4: The stack of residual separated block (RSB) consists of four layers of separated convolutional layer each of which is followed by batch normalization and activation function.

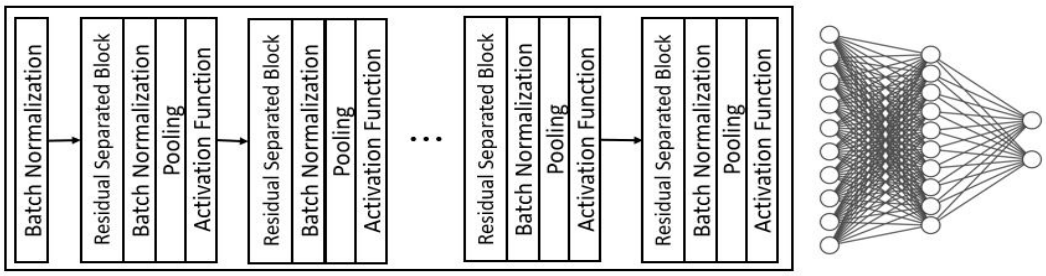


FIGURE 4.5: The complete proposed tailored CNN architecture.

parameter update for early layers [26] and preventing the network from learning new patterns. Batch normalization [26] and the use of ReLU activation function [23] alleviate these two problems.

The deep layers of CNN networks sometimes need to approximate the identity function which is not a simple task, especially with the existence of a non-linear function. Residual connection [22] overcomes this problem by using skip connection as shown in Fig. 4.4. Fig. 4.4 represents the building block layer of the feature extraction phase, denoted by a stack of Residual Separated Block (RSB). RSB consists of four layers of separated convolutional layers, each layer is followed by a batch normalization and an activation function. It has an output of depth N where each sublayer produces an output of depth $N/4$ which is concatenated at the end of the layer to produce a depth N. RSB produces a feature map that includes both low-level features and high-level features.

Unlike the traditional neural network, which is fully connected to the previous layer, the convolutional neural network is connected locally to a local region of the previous feature map. This introduces the concept of the network receptive field [90]. The receptive field should be large enough to capture large patterns in the input chest X-ray image. Therefore, any consecutive convolutional layers in the proposed method without a pooling layer in between a larger

kernel size are used in one of them. Residual Separated block, RSB, in Fig. 4.4 may have kernel sizes of 3, 5, 7, and 9, respectively.
Fig. 6.2 Represent a complete CNN architecture.

4.2 Summary

In this chapter, the proposed lightweight CNN architecture for COVID-19 detection is designed with the concept of spatial separability in mind. The spatial separability of the convolutional kernel is used to enforce the learning of linear kernels, which reduces the number of training parameters and improves computational efficiency. Essentially, the model is designed to recognize patterns in the data that are linearly separable, which enables the use of simpler and more efficient models.

The proposed architecture comprises separated kernel convolutional layers that are connected by a residual connection. The use of separated kernel convolutional layers helps to reduce the number of training parameters, while the residual connection improves network stability during the training process. The residual connection enables the model to retain important features while also discarding unimportant ones, which helps prevent overfitting.

Batch normalization is used in the proposed architecture to maintain network stability during the training process. The technique standardizes the inputs of each layer, which improves the convergence rate of the model. This is done by normalizing the layer inputs to have zero mean and unit variance, which helps prevent internal covariate shifts. Internal covariate shift refers to the change in the distribution of network activations that occurs during training, which can slow down the convergence rate.

In summary, the proposed lightweight CNN architecture for COVID-19 detection is based on the spatial separability of the convolutional kernel, with separated kernel convolutional layers connected by a residual connection. Batch normalization is used to maintain network stability during training, which improves the convergence rate of the model. By combining these techniques, the proposed architecture offers an efficient, accurate, and stable method for detecting COVID-19.

Chapter 5

Proposed Methodology II

CNN, like many computer vision models, is a scale-variant [40] model such that it cannot recognize objects at various scales unless it explicitly trained to recognize such objects. Data augmentation can accomplish some degree of invariance as it allows the network to be trained with distorted samples, but it not the case for pneumonia scales. This chapter presents a CNN architecture that learns multiscale features using scale pyramid of the CNN's internal feature maps. Scale pyramid is constructed using atrous convolution of various dilation rates. The correct scale from scale pyramid that allows minimization of the objective function loss is selected using the spatial attention mechanism.

5.1 Methodology II

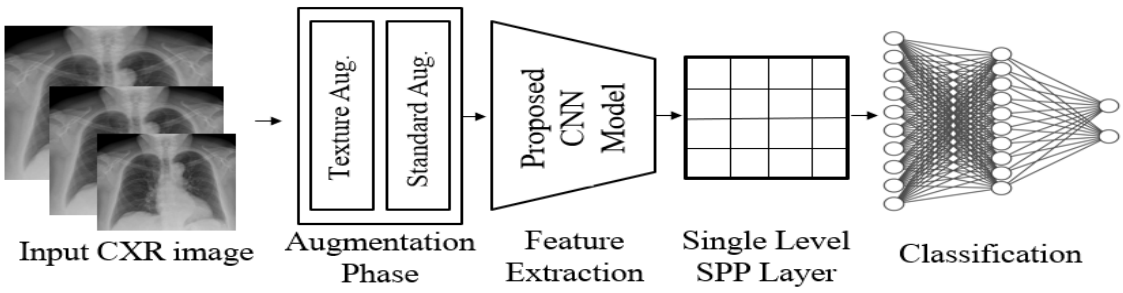


FIGURE 5.1: Proposed method for COVID-19 classification from CXR images.

The proposed system presented in this chapter proposes a novel CNN micro-architecture model for learning scale-invariant features from row input CXR images and then classifies these features into normal or COVID-19 cases. Fig. 5.1

illustrates trainable end-to-end pipeline of the proposed system. The proposed system depends on a novel Spatially weighted Atrous Spatial Pyramid Pooling (SWASPP) to extract multi-scale features of input CXR images. A novel attention module is then used to fuse the extracted these multi-scale features and select relevant features' scale that the next layer should consider.

5.1.1 Data augmentation

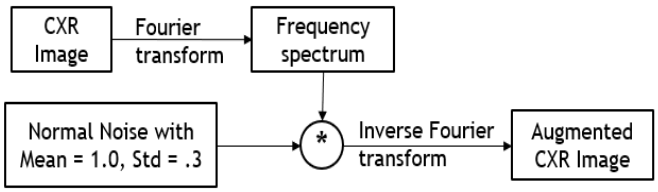


FIGURE 5.2: Texture Augmentation module

The first phase of the proposed CXR classification system is data augmentation phase. Data augmentation is used to reduce the overfitting by artificially enlarge the training dataset [23] using label preserving transformation. Data augmentation phase introduces a degree invariance to a distortion transformation such as the flipping and rotation. The input CXR images are augmented using texture augmentation. Texture augmentation is performed by introducing a multiplicative normally distributed noises to the frequency spectrum of the input image. CXR image is transformed to the frequency spectrum using the fourier transform. Noise is modeled using $\mathcal{N}(\mu = 1, \sigma = 0.3)$. Fig. 5.2 illustrates texture augmentation process for frequency distortion of the CXR image. Fig. 5.3 shows the original CXR image and the corresponding frequency distorted CXR image. A standard augmentation techniques such as random rotation, horizontal flipping, and vertical flipping are included in the augmentation process.

5.1.2 Spatially Weighted Atrous Spatial Pyramid Pooling

Atrous convolution is a powerful technique for adjusting the resolution of convolutional kernels. This allows to effectively enlarge the field-of-view of the kernel

FIGURE 5.3: Texture Augmentation
The resulting CXR image from Texture augmentation **left**: is the original image. **Right** is the augmented CXR Image

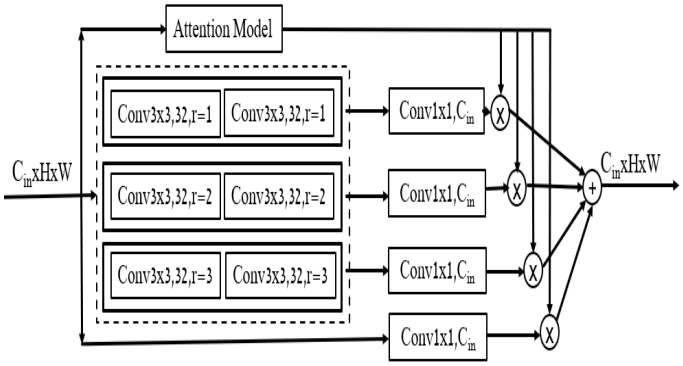


FIGURE 5.4: Spatially weighted atrous spatial Pyramid Pooling (SWASPP) interal layers within dashed square are parameter shared.

without increasing neither the number of kernel parameters nor the computational complexity of the convolution operation. Atrous convolution is equivalent to performing downsampling and then performing convolution with original kernel without dilation. As a result different dilation rates of the kernel corresponding to different downsampling degrees. A novel spatially weighted atrous spatial pyramid pooling (SWASPP) micro-architecture is presented that exploit the scale space of the CNN's feature maps. Fig. 5.4 shows the architecture of the SWASPP. In Fig. 5.4, internal pipelines, bounded by dashed-line square, are parameter-shared and each pipeline of these has a different dilation rates. These pipelines are responsible for extracting multi-scale, scale invariant, features. Sharing of the parameters enforce these pipelines to learn features that exists at multiple levels of scale-pyramid and hence scale-invariance. For a given input CXR image, three scales feature maps are produced.

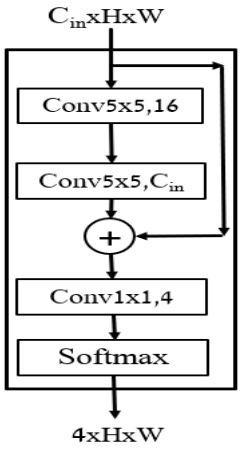


FIGURE 5.5: Attantion module structure used by SWASPP micro-architecture

To fuse these feature maps produced by different pipeline of the SWASPP from the input feature map, an attention module is emerged. Attention module can be thought as a pixel level classification of which scale does this pixel it belongs to. Fig. 5.5 illustrates the proposed attention module structure. Proposed attention module generates four heatmaps. The first three heatmaps correspond to the three scale feature maps while the remaining heatmap corresponds to the input feature map itself. These heatmaps are summed up to one (*i.e.,* for a spatial position (x, y), $\sum_{i=1}^{4} H(i, x, y) = 1$ where $H(i, x, y)$ is the i heatmap produced by the attention module). To make sure this property holds, softmax function is used.

The proposed mirco-architecture uses a pixel level weights produced by corresponding attention module rather than a single weight value for each scale. A single input CXR image may have multiple COVID-19 pneumonia scales which effectively lead to simply averaging the scale space when using single weight for each scale on scale space. In SWASPP, every convolution operation is followed by a BN and leakyReLU [23] non-linearity except the re-projection layers that used to project back to the input space is not followed by nonlinearity. BN allows the use of larger learning rate [26] and makes network stable during training [26]. BN makes the loss landscape of the optimization problem significantly smoother [89]. leakyReLU is used to reduce the vanishing gradient problem [23]. A bottleneck is introduced within both the attention module and multi-scale feature extractor pipeline. A bottleneck in SWASPP is used to project the input feature map of dimension $C_{in} \times H \times W$ to $32 \times H \times W$ then re-project back to $C_{in} \times H \times W$. Multi-scale feature extraction is preformed on the projected dimension. Same logic is applied to the attention module where the input feature map is projected to a dimension of $16 \times H \times W$. This bottleneck allows the efficient use of model capacity and reduce the network computational complexity [25]. It only allows the flow of important information and discarding irrelevant information.

5.1.3 Proposed CNN Architecture

SWASPP is densely stacked [25] together as Fig. 5.6 illustrates. This kind of connectivity allows implicit deep supervisions as each layer is effectively connected to the last layer using shorter path also facilitate feature reuse [25] and gradient flow. Residual layers are easier to optimize if the required mapping is the identity mapping or simply near to it [22]. Densely stacked SWASPP is denoted by (DSWASPP). Convolutional part of proposed model consists of stacking six DSWASPP layers such that the first four layers are interconnected using maxpooling to reduce the spatial size and enlarge the Network receptive field. A single level Spatial Pyramid Pooling (SPP) [91] is added after to produce a fixed size feature vector for a variable size input. SPP layer divides the input feature map into $10 \times 10 = 100$ bins then performs a max for each bin as an aggregation function.

The fixed length feature vector produced by SPP is used as an input to dropout [39] layer. Dropout layer randomly sets the activation of to 0 with a probability of 0.5. Dropout prevents the overfitting and reduce complex

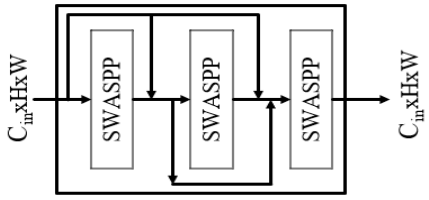


FIGURE 5.6: Densely connected SWASPP (DSWASPP): is a stack of densely connected SWASPP, such that the output of any SWASPP is Concatenated to the input of all next layers. All the three layers produce an output of dimension of $C_{in} \times H \times W$.

co-adaptation between the neurons allowing them to learn better representation [39]. It allow implicit ensempling of exponential number of sampled thin network from the original network which enhance the network performance [39]. The result of dropout layer is used as input to the classification network. Classification network consists of a fully connected layers with a 3 Dense layers such that the output layer is 2-neuron for binary classification *i.e)* COVID19 or not. Table 5.1 shows the details of the proposed architecture.

5.2 Summary

Convolutional Neural Networks (CNNs) have been widely used in computer vision tasks, including image classification, object detection, and segmentation. However, CNNs are known to be scale variant models, meaning that they can miss important features at different scales. To overcome this issue, various approaches have been proposed, such as shared networks, feature pyramid networks, and atrous convolution.

Atrous convolution, also known as dilated convolution, increases the receptive field of the convolutional kernel without increasing the number of parameters or computational complexity. In the proposed work II, atrous convolution is used to construct the scale space of the input feature, allowing the CNN to extract multiscale features.

Moreover, to select the correct scale and fuse multiple scales of the input feature map, a spatial attention module is used in the proposed work II. This attention mechanism guides the network to focus on the most relevant parts of the feature maps, which helps to improve the accuracy of the network.

To further enhance the performance of the network, a novel CNN architecture is proposed in which multiscale feature maps are internally produced and then fused using an attention-based mechanism. This approach leads to a compact representation of the input data via a bottleneck dimension introduced in

TABLE 5.1: Proposed CNN architecture of methodology II

Layer Name	**Proposed CNN Architecture of Methodology II**		
	Input Shape	***Output Shape***	***Param. Count***
Input layer	-	$1 \times 320 \times 320$	0
BatchNorm-1	$1 \times 320 \times 320$	$1 \times 320 \times 320$	2
DSWASPP-1	$1 \times 320 \times 320$	$32 \times 320 \times 320$	121,035
Maxpooling-1	$32 \times 320 \times 320$	$32 \times 160 \times 160$	0
DSWASPP-2	$32 \times 160 \times 160$	$64 \times 160 \times 160$	298,236
Maxpooling-2	$64 \times 160 \times 160$	$64 \times 80 \times 80$	0
DSWASPP-3	$64 \times 80 \times 80$	$128 \times 80 \times 80$	604,956
Maxpooling-3	$128 \times 80 \times 80$	$128 \times 40 \times 40$	0
DSWASPP-4	$128 \times 80 \times 80$	$128 \times 80 \times 80$	784,092
DSWASPP-5	$128 \times 80 \times 80$	$128 \times 80 \times 80$	784,092
DSWASPP-6	$128 \times 80 \times 80$	$128 \times 80 \times 80$	784,092
SPP-1	$128 \times 80 \times 80$	12800	0
Dropout-1	12800	12800	0
FC-1	12800	128	1,638,528
FC-2	128	128	16,512
FC-3	128	64	8,256
FC-4	64	2	130
Softmax	2	2	0
Total Number of Parameter			5,040,571

Any linear combination is followed by BN and leakyReLU nonlinearity excluding re-projection layer of the SWASPP modules

both the multiscale feature extractor module and the attention module. Overall, these techniques and architectures help to improve the performance and accuracy of CNNs for computer vision tasks.

Chapter 6

Experimental Results

In this chapter proposed methodologies are evaluated and compared with the related work.

All models are Trained using QaTa-Cov-19 [92] dataset using NVIDIA Tesla P-100 GPU and programmed using PyTorch.

6.1 QaTa-COV19 Dataset

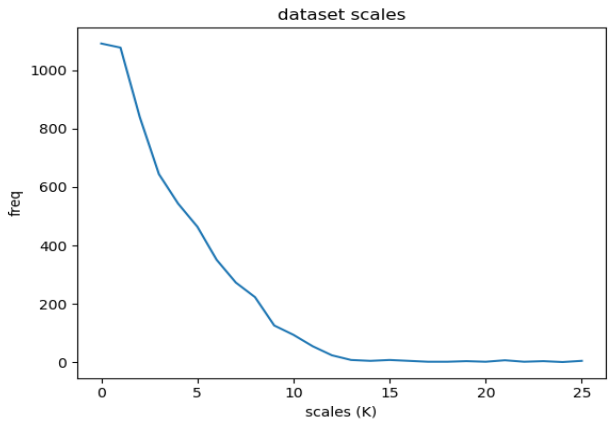


FIGURE 6.1: Pneumonia Scales of QaTa-COV19-v1, Y-axis represents the frequency, number of occurrences, of pneumonia with a particular area

QaTa-COV19 is a benchmark dataset for COVID-19 detection and Segmentation from CXR images. All models used for comparison are trained using QaTa-COV19-v1. Qata-COV19-v1 consists of 4603 COVID-19 CXRs and $120,013$ control group CXRs. A balanced number of samples for the two classes is used, namely 4603 CXR images for each class to train the models. Pneumonia Scales

of QaTa-COV19-v1 do not exhibit a uniform distribution. The scale of the Pneumonia can be defined as the number, area, of 8-neighbor-connected pixels labeled as COVID-19 pneumonia. QaTa-COV19-v1 provides a binary mask of 2951 COVID-19 CXR images which can be used for approximating the distribution of scales across the dataset. Fig. 6.1 illustrates the statistical distribution of QaTa-COV19 scales. The non-uniform distribution of the scales allows the CNN models to only recognize the small scales and not the large scales.

6.2 Evaluation of the Methodology I

Experiments are conducted on a Lenovo Z50-70 with Intel CORE i7-4510U CPU 2.00 GHz, 8GB RAM, NVIDIA GeForce 840M GPU; and with Python and PyTorch library.

6.2.1 Details of the Proposed Architecture

The Proposed architecture is composed of Convbase and Densebase. Convbase is composed of a 6 feature extraction modules *(FX)* preceded by batch normalization layer. Each FX module can be considered a sub-sequential model consisting of an RSB layer followed by Batch Normalization, Max-pooling, and LeakyReLU activation function. The Densebase is two fully connected layers that classify the Convbase output.

6.2.2 Hyperparameter Specification

All input chest X-ray images are resized to be 200×200 . After resizing the input images, these images are fed the Convbase model part which consists of 6 layers of residual separated block. Each residual separated block is followed with batch normalization and LeakyReLU [27] as an activation function. The output depth of each residual separated block is 4×16, 4×32, 4×64, 4×64, 4×64, and 4×16, respectively. The output of the Convbase model part is a 1D feature vector of 576 in length. The densebase model part consists of two hidden layers. Each layer has a size of 64 and the output layer of size 2. Each layer of Densebase layers is fully connected to its previous layer. The activation function used in the densebase model part is LeakyReLU. Table 6.1 summarizes the architecture hyperparameters.

TABLE 6.1: The proposed architecture hyperparameters

Layer Number	**Layer Size**	**Activation Function**
RSBLayer1	4 × 16	LeakyReLU
RSBLayer2	4 × 23	LeakyReLU
RSBLayer3	4 × 64	LeakyReLU
RSBLayer4	4 × 64	LeakyReLU
RSBLayer5	4 × 64	LeakyReLU
RSBLayer6	4 × 16	LeakyReLU
Flatten The Feature maps to 1D 576 feature vector		
LinearLayer1	64	LeakyReLU
LinearLayer2	64	LeakyReLU
LinearLayer3	2	Softmax

6.2.3 Network Training

The proposed CNN model is trained for 22 epochs. Adaptive Moment Estimation (Adam) optimizer [93] is a popular optimization technique for training deep networks. Adam optimizer is used during the training phase of the proposed CNN model. Both batch size and Adam optimizer learning rate are changed during the training phase if the training loss stops decreasing. Table 6.2 summarizes the parameter values used in the training phase of the proposed CNN model. Fig. 6.2(a) shows the progress for training and validation loss across each epoch. The difference between the training loss and validation loss through epochs shows that we did not memorize the dataset.

TABLE 6.2: The change of batch size and learning rate through the Training process

Epoch Number	**Batch Size**	**Learning Rate**
From 0 to 6	128	1e-3
From 7 to 12	256	1e-3
From 13 to 21	256	1e-4

6.2.4 Model Evaluation

To assess the efficiency of the proposed method, the proposed method is compared to recent state-of-the-art methods for detecting Covid-19 cases. Experiments are conducted with the same dataset and the corresponding hyperparameter of each work. All the methods depend on CNN. The comparison is performed using precision, sensitivity, F1-score, and accuracy [94]. In addition, the number of parameters used in the training phase is a very important comparison factor. Table 6.3 depicts the comparison between state-of-the-art methods and the proposed method. As shown in the comparison, the proposed method outperforms other methods achieving the maximum accuracy and the lowest parameter count.

TABLE 6.3: A performance comparison between the proposed method and state-of-the-art models.

Method	**PC**	**P(%)**	**S(%)**	**F1(%)**	**A(%)**
Proposed Method	0.15M	100.00	100.00	100.00	100.00
ResNet-34 [95]	21.8M	96.77	100.00	98.36	98.33
ACoS Phase I [96]	-	98.266	96.512	98.551	98.062
ResNet-50 [95]	25.6M	95.24	100.00	97.56	97.50
GoogleNet [95]	5M	96.67	96.67	96.67	96.67
VGG-16 [95]	138M	95.08	96.67	95.87	95.83
AlexNet [95]	60M	96.72	98.33	97.52	97.50
MobileNet-V2 [95]	3.4M	98.24	93.33	95.73	95.83
Inception-V3 [95]	24M	96.36	88.33	92.17	92.50
SqueezeNet [95]	1.25M	98.27	95.00	96.61	96.67

PC is Parameter count, P is precision, S is sensitivity

F1 is F1-score, and A is accuracy

6.3 Evaluation of the Methodology II

Methodology II is evaluated and compared against strong baselines and related works.

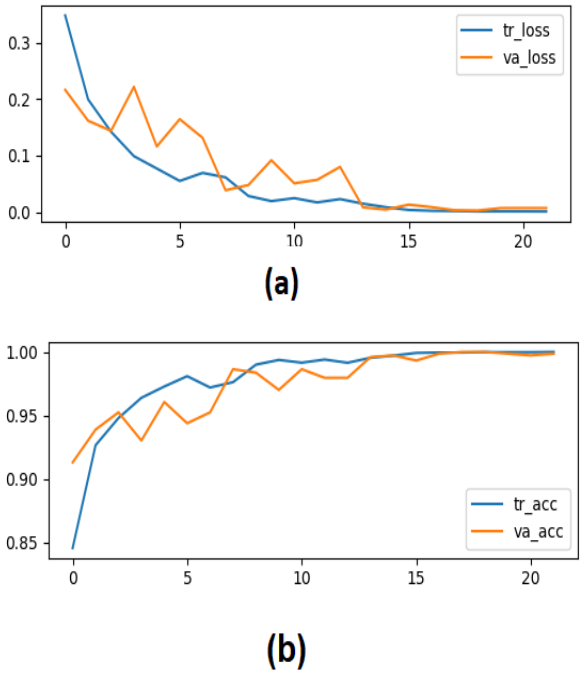


FIGURE 6.2: (a) The training loss and the validation loss of each epoch and (b) The training accuracy and the validation accuracy of each epoch.

6.3.1 Baseline Networks

Different architectures are trained to validate the effectiveness of the proposed method.

Spatial Pyramid Pooling (SPP-net) Based model

Four variants of SPP-net[91] is trained. All 4 variants have the same architecture but different SPP-layer. These variants of SPP-layer are as follows:

- full pyramid SPP of 8-levels using average-pooling as an aggregation function
- full SPP pyramid of 8-levels using max-pooling as an aggregation function
- single level SPP with 10-bins using average-pooling as an aggregation function
- single level SPP with 10-bins using max-pooling as an aggregation function

A Fixed Architecture is used for all SPP variant models with the same design principles of the proposed architecture. These architectures are the same

as the proposed architecture but DSWASPP is replaced by DC6 and SPP-1 layer is replaced with the corresponding SPP layer. DC6 is defined as six convolutional layers Densely connected. For SPP-net variants training a multiscale augmentation is added to the proposed augmentation process. Multi-scale augmentation is done by randomly sampling different 5 scales typically $\{320, 320 \pm 25, 320 \pm 50\}$.

Switchable Atrous Spatial Pyramid Pooling (SASPP-net) Based models

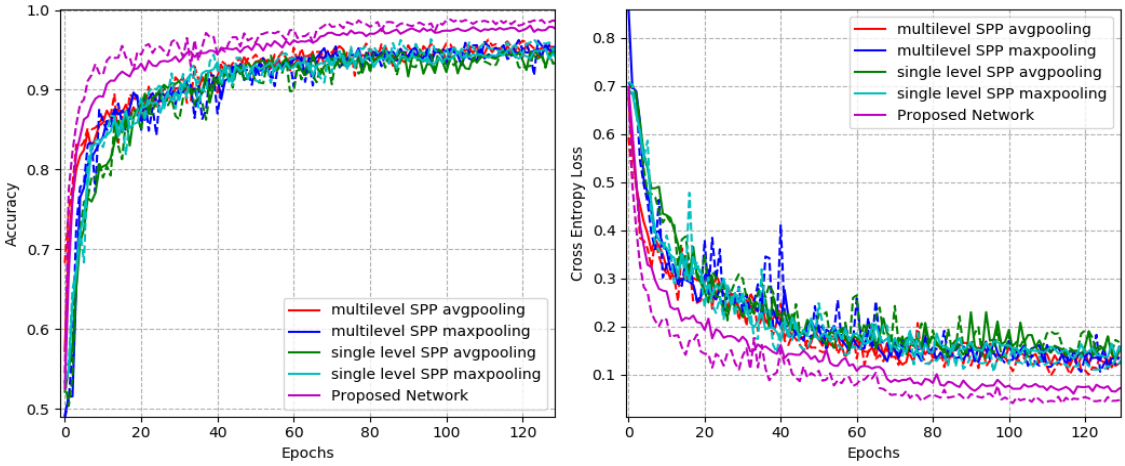


FIGURE 6.3: Training profiles of both SPP-net variants and the proposed network. For the same color solid line represents training statistics while the dashed line represents the validation statistics for the corresponding model. **left:** is the training accuracy. **Right:** is the training loss.

Another Baseline is introduced for comparison which is exactly as same as the proposed network but with a different Attention module structure and does not include a bottleneck within ASPP. This architecture is referred to as Switchable Atrous Spatial Pyramid Pooling (SASPP-net). The attention module structure is $Softmax(FC(GAP(X)))$ where: *X: is the input feature map, GAP: is a global average pooling, FC: is fully Connected layer performs a non-linear projection to $\mathbb{R}^4$ 4 values for the three scales and the input feature map.*

Table 6.4 summarizes the baseline models and the Corresponding parameter count.

TABLE 6.4: Baselines and their total number of parameters

Model Type	**Baseline CNN Architectures**	
	Variant	***Param. Count***
SPP	ML Average pooling	14,916,420
	ML max pooling	14,916,420
	SL Average pooling	14,490,436
	SL max pooling	14,490,436
SASPP		13,031,841

ML: Multilevel, **SL**: Single level

6.3.2 Models Training

The proposed architecture and baseline architecture are trained with the same hyperparameters. The dataset is split into 0.6, 0.2, and 0.2 for training, validation, and testing, respectively. For training a Cross Entropy Loss is used. All models trained with ADAM [93] optimizer with learning rate start by 10^{-3} and reduced every time validation loss plateau by multiplying by 10^{-1}. A Max Norm Constraint is used to clip the gradient value to the norm of 1 [23]. A batch size of 128 is used to calculate the gradient.

6.3.3 Reducing the overfitting

Overfitting is a critical problem for training large networks [23]. The proposed work has reduced the overfitting by using:

- Using Dropout with retrain probability of 0.5 [39].
- Using BatchNorm adds noise due to randomization introduced when constructing the minibatch [26].
- Using max norm constraint [23].
- Deep and thin architectures by design have an implicit regularization effect [22].
- Augmentation process i.e.) Texture augmentation [23].
- The use of small kernel size [21].

- Bottleneck in SWASPP module and the attention module.

During training no overfitting effects are observed.

6.3.4 Comparison with baselines

The proposed network is compared with the vanilla SPP-based Architecture and ASPP architecture.

Comparing with SPP-nets

Fig. 6.3 illustrates both training loss and training and validation accuracies and losses. Table 6.5 illustrates the testing accuracy for comparison between the SPP-nets baseline and the proposed architecture.

TABLE 6.5: Comparison between Proposed network and baseline SPP architectures

Model Name	Accuracy
SPP ML Average pooling	0.958
SPP ML max pooling	0.950
SPP SL Average pooling	0.927
SPP SL max pooling	0.957
Proposed Network	0.987

ML: Multilevel, **SL**: Single level

Comparing with SASPP

Fig. 6.4 illustrates the training and validation loss of training a SASPP baseline architecture. As shown in Fig. 6.4 SASPP is unable to generalize and start overfitting the training set. This comparison empirically shows the importance of the bottleneck introduced in the proposed architecture.

6.3.5 Comparing with the related works

To fairly compare with the related works proposed work is further trained. Fig. 6.5 shows the training and validation loss of the proposed network. Fig. 6.6 shows the of the training and validation accuracy.

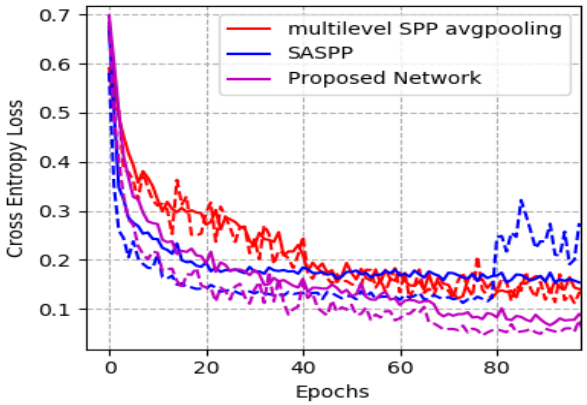


FIGURE 6.4: SASPP baseline architecture loss during both training, solid line, and validation, bashed line, compared with Proposed network and best performing SPP architecture.

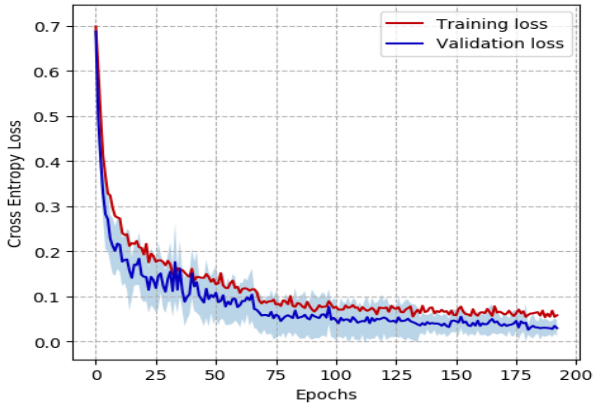


FIGURE 6.5: Cross entropy loss of the proposed architecture.

The proposed Network has a sensitivity, recall, and precision of 0.994 and 0.991 respectively on the validation set. Precision can be improved by investigating the precision-recall trade-off. Fig. 6.7 shows the trade-off between precision and recall for different thresholds. A threshold of 0.618 is used to improve the precision resulting in a sensitivity, recall, of 0.9903 and precision of 0.9956. Comparison metrics are defined as follows:

- *Accuracy*: is the ratio of correctly classified samples to the total number of samples
- *Sensitivity*: is the ratio of correctly classified Covid-19 samples to the total number of actual Covid-19 samples

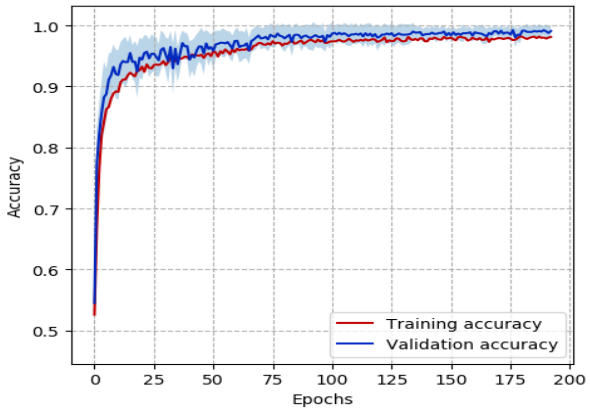


FIGURE 6.6: Training and validation accuracy of the proposed architecture.

TABLE 6.6: Comparison between Proposed network and Related works

Model Name	Accuracy	Sensitivity	Precision	Specificity	F1-score	Param. Count
Proposed	0.99294	**0.9903**	**0.9956**	0.9956	**0.9929**	**5,040,571**
SRC-Dalm[54]	0.985	0.886	-	0.993	-	-
SRC-Hom[54]	0.977	0.921	-	0.982	-	-
CRC-light[54]	0.973	0.955	-	0.974	-	-
DenseNet121*[54]	0.992	0.9714	-	0.9949	-	6,955,906
Inception-v3[54]	0.993	0.954	-	0.998	-	21,772,450
Modified MobileNetV2 [58]	0.98	0.98	0.97	-	0.97	-
ReCovNet-v2[53]	0.99726	0.98571	0.94262	0.9977	0.96369	-
ReCovNet-v1[53]	0.99824	0.9781	0.97438	0.99901	0.97624	-
DenseNet-121[53]	**0.9988**	0.97429	0.9932	**0.99974**	0.98365	6,955,906

- *Precision*: is the ratio of correctly, according to the ground-truth labels, classified Covid-19 samples to the total number of samples classified as Covid-19.
- *Specificity*: is the ratio of correctly, according to the ground-truth labels, classified non-COVID-19 to the total number of non-COVID-19.
- *F1-score*: is the harmonic mean of both Sensitivity and Precision.

$$F_1 = \frac{2 \times \text{Precision} \times \text{Sensitivity}}{\text{Precision} + \text{Sensitivity}}$$

- *Param. Count*: is the total number of the trainable parameters.

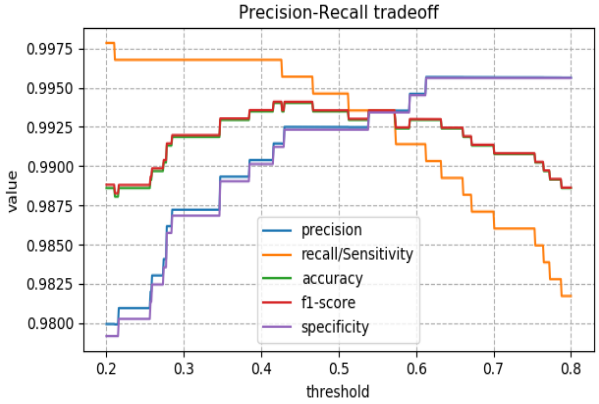


FIGURE 6.7: precision-recall trade-off of the proposed network.

Table 6.6 summarizes the comparison between the recent related works and the proposed architecture. The proposed architecture outperforms these works in many metrics. As their training and testing do not depend on a balanced number of samples, accuracy, and specificity are not good metrics for evaluation.

6.4 Summary

This chapter illustrates the superior performance of the proposed work I and II. In addition to the superior performance of the proposed works I and II, the chapter also provides a detailed analysis of the experimental results. The evaluation of proposed work I demonstrates that the use of spatially separable kernels and residual connection significantly improves the performance of the COVID-19 detection system. The batch normalization technique is found to be effective in maintaining network stability during the training process. The dynamic selection of hyperparameters such as batch size and learning rate further improves the performance of the proposed architecture I.

Furthermore, the proposed architecture I outperforms existing works for binary classification of chest X-ray images for normal and COVID-19 cases. This is particularly noteworthy as the proposed architecture has a low parameter count of only 150K trainable parameters compared to previous works. The achievement of a performance of 100% for accuracy, sensitivity, precision, and F1-score indicates the high accuracy and reliability of the proposed architecture.

Although the proposed architecture I performs exceptionally well, it does not address the scale variant nature of the CNN model. This issue is addressed in the proposed work II, which incorporates multiscale training approaches to obtain better quantitative results for CXR COVID-19 classification. The use of Atrous Spatial pyramid pooling enables the internal production of multiscale

feature maps, which are then fused using an attention module. To achieve a compact representation, a bottleneck dimension is introduced in both the multiscale feature extractor module and the attention module.

The proposed work II outperforms the current state-of-the-art architecture with lower parameter numbers. The recorded 0.9929 for the F1-score indicates the high performance of the proposed architecture. The detailed analysis and experimental results presented in this chapter provide valuable insights into the effectiveness of the proposed architectures and their potential for improving the accuracy and reliability of COVID-19 detection systems.

Chapter 7

Conclusion And Future Work

The COVID-19 pandemic has caused severe respiratory tract infections that have rapidly spread through contact with infected individuals, resulting in devastating loss of life and economic damage worldwide. The high rate of transmission has put tremendous pressure on healthcare systems to develop fast and accurate methods for diagnosing the disease. Convolutional Neural Networks (CNNs) have shown success in various computer vision tasks, but they are scale-variant and computationally expensive. In this thesis, we proposed novel architectures for multiscale feature extraction and classification, as well as a lightweight architecture for COVID-19 diagnosis.

The proposed lightweight CNN model, referred to as CNN-I, exploits spatial kernel separability to significantly reduce the number of training parameters and regularizes the model to only learn linear kernels. To maintain network stability and reduce overfitting, residual connections, and batch normalization are extensively used. We trained this lightweight architecture on the QaTa-Cov19 benchmark dataset, achieving 100% accuracy, sensitivity, precision, and F1-score with a parameter count of only 150K, which is significantly lower than other methods in the literature. As future work, attention and context attention can be explored to further enhance performance, and evaluating atrous convolution in the context of spatial separability may be beneficial.

Our second proposed architecture, CNN-II, learns multiscale features using a pyramid of shared convolution kernels with different atrous rates, making it scale-invariant. An attention-based mechanism is used to guide and select the correct scale for each input. CNN-II is an end-to-end trainable network that exploits a novel augmentation technique, Texture Augmentation, to reduce overfitting. This architecture achieved an F1-score of 0.9929 when tested on the QaTa-Cov19 benchmark dataset, with a total of 5,040,571 trainable parameters. We suggest that the SWASPP (Spatial Pyramid Atrous Spatial Pyramid

Pooling) can show great performance for segmentation, especially atrous convolution originating in the segmentation literature. Additionally, this work can be extended to classify various types of pneumonia.

In conclusion, this thesis proposes novel architectures for COVID-19 diagnosis that address the limitations of traditional CNN models. These architectures achieved high accuracy while reducing computational cost and parameter count. Further research can explore attention mechanisms and evaluate the use of atrous convolution in the context of spatial separability to improve performance. This work has the potential to improve COVID-19 diagnosis and aid in the development of fast and effective methods to combat future pandemics.

Chapter 8

Summary

The COVID-19 pandemic has caused severe respiratory tract infections that have rapidly spread through contact with infected individuals, resulting in devastating loss of life and economic damage worldwide. The high rate of transmission has put tremendous pressure on healthcare systems to develop fast and accurate methods for diagnosing the disease. Convolutional Neural Networks (CNNs) have shown success in various computer vision tasks, but they are scale-variant and computationally expensive. In this thesis, we proposed novel architectures for multiscale feature extraction and classification, as well as a lightweight architecture for COVID-19 diagnosis.

The proposed lightweight CNN model, referred to as CNN-I, exploits spatial kernel separability to significantly reduce the number of training parameters, and regularizes the model to only learn linear kernels. To maintain network stability and reduce overfitting, residual connections and batch normalization are extensively used. We trained this lightweight architecture on the QaTa-Cov19 benchmark dataset, achieving 100% accuracy, sensitivity, precision, and F1-score with a parameter count of only 150K, which is significantly lower than other methods in the literature. As future work, attention and context attention can be explored to further enhance performance, and evaluating atrous convolution in the context of spatial separability may be beneficial.

Our second proposed architecture, CNN-II, learns multiscale features using a pyramid of shared convolution kernels with different atrous rates, making it scale-invariant. An attention-based mechanism is used to guide and select the correct scale for each input. CNN-II is an end-to-end trainable network that exploits a novel augmentation technique, Texture Augmentation, to reduce overfitting. This architecture achieved an F1-score of 0.9929 when tested on the QaTa-Cov19 benchmark dataset, with a total of 5,040,571 trainable parameters. We suggest that the SWASPP (Spatial Pyramid Atrous Spatial Pyramid

Pooling) can show great performance for segmentation, especially atrous convolution originating in the segmentation literature. Additionally, this work can be extended to classify various types of pneumonia.

In conclusion, this thesis proposes novel architectures for COVID-19 diagnosis that address the limitations of traditional CNN models. These architectures achieved high accuracy while reducing computational cost and parameter count. Further research can explore attention mechanisms and evaluate the use of atrous convolution in the context of spatial separability to improve performance. This work has the potential to improve COVID-19 diagnosis and aid in the development of fast and effective methods to combat future pandemics.

- **Chapter 1**: briefly discussed the history of COVID-19 and the importance of automating COVID-19 detection.
- **Chapter 2**: includes required Background to understand the Thesis.
- **Chapter 3**: includes and illustrates the recent and related work in the COVID-19 detection literature.
- **Chapter 4**: presents the proposed work I which presents a lightweight classification model.
- **Chapter 5**: presents the proposed work II which includes the scale invariant model for COVID-19 classification.
- **Chapter 6**: illustrates the experimental results for both proposed work I and II and quantitative analysis of the proposed work I and II is provided.
- **Chapter 7**: concludes the thesis and provide planning for the future work as extension of the proposed approach

Bibliography

Publications

[97] M Zaki, Khalid Amin, and Ahmed Mahmoud Hamad. "COVID-19 Detection Based on Chest X-Ray Image Classification using Tailored CNN Model". In: *IJCI. International Journal of Computers and Information* 8.2 (2021), pp. 100–108.

[98] M Zaki, Khalid Amin, and Ahmed Mahmoud Hamad. "Multiscale aware classification of COVID-19 from Chest X-Ray using a spatially weighted atrous spatial pyramid pooling CNN". In: *submitted* ().

References

[1] Thamina Acter et al. “Evolution of severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) as coronavirus disease 2019 (COVID-19) pandemic: A global health emergency”. In: *Science of the Total Environment* 730 (2020), p. 138996.

[2] Tanu Singhal. “A review of coronavirus disease-2019 (COVID-19)”. In: *The indian journal of pediatrics* 87.4 (2020), pp. 281–286.

[3] David S Hui et al. “The continuing 2019-nCoV epidemic threat of novel coronaviruses to global health—The latest 2019 novel coronavirus outbreak in Wuhan, China”. In: *International journal of infectious diseases* 91 (2020), pp. 264–266.

[4] Sara Platto, Tongtong Xue, and Ernesto Carafoli. “COVID19: an announced pandemic”. In: *Cell Death & Disease* 11.9 (2020), pp. 1–13.

[5] Tung Thanh Le et al. “Evolution of the COVID-19 vaccine development landscape”. In: *Nat Rev Drug Discov* 19.10 (2020), pp. 667–668.

[6] Alireza Tahamtan and Abdollah Ardebili. “Real-time RT-PCR in COVID-19 detection: issues affecting the results”. In: *Expert review of molecular diagnostics* 20.5 (2020), pp. 453–454.

[7] Sana Salehi et al. “Coronavirus disease 2019 (COVID-19): a systematic review of imaging findings in 919 patients”. In: *Ajr Am J Roentgenol* 215.1 (2020), pp. 87–93.

[8] Fan Wu et al. “A new coronavirus associated with human respiratory disease in China”. In: *Nature* 579.7798 (2020), pp. 265–269.

[9] Zi Yue Zu et al. “Coronavirus disease 2019 (COVID-19): a perspective from China”. In: *Radiology* 296.2 (2020), E15–E25.

[10] KONRAD A Erickson, KR Mackenzie, and AJ Marshall. “Advanced but expensive technology. Balancing affordability with access in rural areas.” In: *Canadian family physician Medecin de famille canadien* 39 (1993), pp. 28–30.

[11] Ali Narin, Ceren Kaya, and Ziynet Pamuk. "Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks". In: *Pattern Analysis and Applications* 24.3 (2021), pp. 1207–1220.

[12] David J Brenner and Eric J Hall. "Computed tomography—an increasing source of radiation exposure". In: *New England journal of medicine* 357.22 (2007), pp. 2277–2284.

[13] Geoffrey D Rubin et al. "The role of chest imaging in patient management during the COVID-19 pandemic: a multinational consensus statement from the Fleischner Society". In: *Radiology* 296.1 (2020), pp. 172–180.

[14] Feng Shi et al. "Review of artificial intelligence techniques in imaging data acquisition, segmentation, and diagnosis for COVID-19". In: *IEEE reviews in biomedical engineering* 14 (2020), pp. 4–15.

[15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *nature* 521.7553 (2015), pp. 436–444.

[16] Yann LeCun et al. "Handwritten digit recognition with a back-propagation network". In: *Advances in neural information processing systems* 2 (1989).

[17] Dumitru Erhan et al. "Scalable object detection using deep neural networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 2147–2154.

[18] Ross Girshick et al. "Rich feature hierarchies for accurate object detection and semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 580–587.

[19] Pierre Sermanet et al. "Overfeat: Integrated recognition, localization and detection using convolutional networks". In: *arXiv preprint arXiv:1312.6229* (2013).

[20] Joseph Redmon et al. "You only look once: Unified, real-time object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[21] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[22] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems* 25 (2012).

[24] Matthew D Zeiler and Rob Fergus. "Visualizing and understanding convolutional networks". In: *European conference on computer vision*. Springer. 2014, pp. 818–833.

[25] Gao Huang et al. "Densely connected convolutional networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 4700–4708.

[26] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *International conference on machine learning*. PMLR. 2015, pp. 448–456.

[27] Kaiming He et al. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1026–1034.

[28] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. "Deep Sparse Rectifier Neural Networks". In: *JMLR W&CP: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2011)*. Apr. 2011.

[29] Josef Sivic and Andrew Zisserman. "Video Google: A text retrieval approach to object matching in videos". In: *Computer Vision, IEEE International Conference on*. Vol. 3. IEEE Computer Society. 2003, pp. 1470–1470.

[30] Kristen Grauman and Trevor Darrell. "The pyramid match kernel: Discriminative classification with sets of image features". In: *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*. Vol. 2. IEEE. 2005, pp. 1458–1465.

[31] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories". In: *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*. Vol. 2. IEEE. 2006, pp. 2169–2178.

[32] Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017).

[33] Meng-Hao Guo et al. "Attention mechanisms in computer vision: A survey". In: *Computational Visual Media* (2022), pp. 1–38.

[34] Long Chen et al. “Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 5659–5667.

[35] Jie Hu, Li Shen, and Gang Sun. “Squeeze-and-excitation networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 7132–7141.

[36] Sanghyun Woo et al. “Cbam: Convolutional block attention module”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 3–19.

[37] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. “Layer normalization”. In: *arXiv preprint arXiv:1607.06450* (2016).

[38] Randall Balestriero, Leon Bottou, and Yann LeCun. “The effects of regularization and data augmentation are class dependent”. In: *arXiv preprint arXiv:2204.03632* (2022).

[39] Nitish Srivastava et al. “Dropout: a simple way to prevent neural networks from overfitting”. In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958.

[40] Nanne Van Noord and Eric Postma. “Learning scale-variant and scale-invariant features for deep image classification”. In: *Pattern Recognition* 61 (2017), pp. 583–592.

[41] Clement Farabet et al. “Learning hierarchical features for scene labeling”. In: *IEEE transactions on pattern analysis and machine intelligence* 35.8 (2012), pp. 1915–1929.

[42] Guosheng Lin et al. “Efficient piecewise training of deep structured models for semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 3194–3203.

[43] Pedro F Felzenszwalb et al. “Object detection with discriminatively trained part-based models”. In: *IEEE transactions on pattern analysis and machine intelligence* 32.9 (2009), pp. 1627–1645.

[44] Dan Ciregan, Ueli Meier, and Jürgen Schmidhuber. “Multi-column deep neural networks for image classification”. In: *2012 IEEE conference on computer vision and pattern recognition*. IEEE. 2012, pp. 3642–3649.

[45] Liang-Chieh Chen et al. “Semantic image segmentation with deep convolutional nets and fully connected crfs”. In: *arXiv preprint arXiv:1412.7062* (2014).

[46] Bharath Hariharan et al. “Hypercolumns for object segmentation and fine-grained localization”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 447–456.

[47] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully convolutional networks for semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440.

[48] Tsung-Yi Lin et al. “Feature pyramid networks for object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2117–2125.

[49] Matthias Holschneider et al. “A real-time algorithm for signal analysis with the help of the wavelet transform”. In: *Wavelets*. Springer, 1990, pp. 286–297.

[50] Liang-Chieh Chen et al. “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs”. In: *IEEE transactions on pattern analysis and machine intelligence* 40.4 (2017), pp. 834–848.

[51] Alessandro Giusti et al. “Fast image scanning with deep max-pooling convolutional neural networks”. In: *2013 IEEE International Conference on Image Processing*. IEEE. 2013, pp. 4034–4038.

[52] Liang-Chieh Chen et al. “Rethinking atrous convolution for semantic image segmentation”. In: *arXiv preprint arXiv:1706.05587* (2017).

[53] Aysen Degerli et al. “Reliable COVID-19 Detection Using Chest X-ray Images”. In: *2021 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2021, pp. 185–189.

[54] Mete Ahishali et al. “Advance warning methodologies for covid-19 using chest x-ray images”. In: *IEEE Access* 9 (2021), pp. 41052–41065.

[55] John Wright et al. “Sparse representation for computer vision and pattern recognition”. In: *Proceedings of the IEEE* 98.6 (2010), pp. 1031–1044.

[56] Wei Li and Qian Du. “A survey on representation-based classification and detection in hyperspectral remote sensing imagery”. In: *Pattern Recognition Letters* 83 (2016), pp. 115–123.

[57] Tanaya Guha and Rabab K Ward. “Learning sparse representations for human action recognition”. In: *IEEE transactions on pattern analysis and machine intelligence* 34.8 (2011), pp. 1576–1588.

[58] Shamima Akter et al. “COVID-19 detection using deep learning algorithm on chest X-ray images”. In: *Biology* 10.11 (2021), p. 1174.

[59] Alassane Bonkano Abdoul-Razak et al. “Hybrid Machine and Deep Transfer Learning Based Classification Models for Covid 19 and Pneumonia Diagnosis Using X-ray Images”. In: *The International Conference on Information, Communication & Cybersecurity*. Springer. 2021, pp. 403–413.

[60] Tej Bahadur Chandra et al. “Coronavirus disease (COVID-19) detection in chest X-ray images using majority voting based classifier ensemble”. In: *Expert systems with applications* 165 (2021), p. 113909.

[61] GN Srinivasan and G Shobha. “Statistical texture analysis”. In: *Proceedings of world academy of science, engineering and technology*. Vol. 36. December. 2008, pp. 1264–1269.

[62] Walter Gómez, Wagner Coelho Albuquerque Pereira, and Antonio Fernando C Infantosi. “Analysis of co-occurrence texture statistics as a function of gray-level quantization for classifying breast ultrasound”. In: *IEEE transactions on medical imaging* 31.10 (2012), pp. 1889–1899.

[63] Navneet Dalal and Bill Triggs. “Histograms of oriented gradients for human detection”. In: *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR’05)*. Vol. 1. Ieee. 2005, pp. 886–893.

[64] Seyedali Mirjalili, Seyed Mohammad Mirjalili, and Andrew Lewis. “Grey wolf optimizer”. In: *Advances in engineering software* 69 (2014), pp. 46–61.

[65] Ioannis D Apostolopoulos and Tzani A Mpesiana. “Covid-19: automatic detection from x-ray images utilizing transfer learning with convolutional neural networks”. In: *Physical and engineering sciences in medicine* 43 (2020), pp. 635–640.

[66] Lawrence O Hall et al. “Finding covid-19 from chest x-rays using deep learning on a small dataset”. In: *arXiv preprint arXiv:2004.02060* (2020).

[67] Ali Narin, Ceren Kaya, and Ziynet Pamuk. “Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks”. In: *Pattern Analysis and Applications* 24 (2021), pp. 1207–1220.

[68] Yu-Xing Tang et al. “Automated abnormality classification of chest radiographs using deep convolutional neural networks”. In: *NPJ digital medicine* 3.1 (2020), p. 70.

[69] Shervin Minaee et al. “Deep-COVID: Predicting COVID-19 from chest X-ray images using deep transfer learning”. In: *Medical image analysis* 65 (2020), p. 101794.

[70] Parnian Afshar et al. “Covid-caps: A capsule network-based framework for identification of covid-19 cases from x-ray images”. In: *Pattern Recognition Letters* 138 (2020), pp. 638–643.

[71] Md Manjurul Ahsan et al. “Covid-19 symptoms detection based on nasnetmobile with explainable ai using various imaging modalities”. In: *Machine Learning and Knowledge Extraction* 2.4 (2020), pp. 490–504.

[72] Ezz El-Din Hemdan, Marwa A Shouman, and Mohamed Esmail Karar. “Covidx-net: A framework of deep learning classifiers to diagnose covid-19 in x-ray images”. In: *arXiv preprint arXiv:2003.11055* (2020).

[73] Tulin Ozturk et al. “Automated detection of COVID-19 cases using deep neural networks with X-ray images”. In: *Computers in biology and medicine* 121 (2020), p. 103792.

[74] Asif Iqbal Khan, Junaid Latief Shah, and Mohammad Mudasir Bhat. “CoroNet: A deep neural network for detection and diagnosis of COVID-19 from chest x-ray images”. In: *Computer methods and programs in biomedicine* 196 (2020), p. 105581.

[75] Tej Bahadur Chandra et al. “Coronavirus disease (COVID-19) detection in chest X-ray images using majority voting based classifier ensemble”. In: *Expert systems with applications* 165 (2021), p. 113909.

[76] Boran Sekeroglu and Ilker Ozsahin. “<? covid19?> Detection of COVID-19 from Chest X-Ray Images Using Convolutional Neural Networks”. In: *SLAS TECHNOLOGY: Translating Life Sciences Innovation* 25.6 (2020), pp. 553–565.

[77] Mohammad Khalid Pandit et al. “Automatic detection of COVID-19 from chest radiographs using deep learning”. In: *Radiography* 27.2 (2021), pp. 483–489.

[78] Julián D Arias-Londoño et al. “Artificial intelligence applied to chest X-ray images for the automatic detection of COVID-19. A thoughtful evaluation approach”. In: *Ieee Access* 8 (2020), pp. 226811–226827.

[79] Mehmet Yamac et al. “Convolutional sparse support estimator-based COVID-19 recognition from X-ray images”. In: *IEEE Transactions on Neural Networks and Learning Systems* 32.5 (2021), pp. 1810–1820.

[80] Linda Wang, Zhong Qiu Lin, and Alexander Wong. "Covid-net: A tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images". In: *Scientific reports* 10.1 (2020), pp. 1–12.

[81] Asmaa Abbas, Mohammed M Abdelsamea, and Mohamed Medhat Gaber. "Classification of COVID-19 in chest X-ray images using DeTraC deep convolutional neural network". In: *Applied Intelligence* 51 (2021), pp. 854–864.

[82] Suat Toraman, Talha Burak Alakus, and Ibrahim Turkoglu. "Convolutional capsnet: A novel artificial neural network approach to detect COVID-19 disease from X-ray images using capsule networks". In: *Chaos, Solitons & Fractals* 140 (2020), p. 110122.

[83] N Narayan Das et al. "Automated deep transfer learning-based approach for detection of COVID-19 infection in chest X-rays". In: *Irbm* 43.2 (2022), pp. 114–119.

[84] Mengjie Hu et al. "Learning to recognize chest-Xray images faster and more efficiently based on multi-kernel depthwise convolution". In: *IEEE Access* 8 (2020), pp. 37265–37274.

[85] Aras M Ismael and Abdulkadir Şengür. "Deep learning approaches for COVID-19 detection based on chest X-ray images". In: *Expert Systems with Applications* 164 (2021), p. 114054.

[86] K Shankar et al. "An optimal cascaded recurrent neural network for intelligent COVID-19 detection using Chest X-ray images". In: *Applied soft computing* 113 (2021), p. 107878.

[87] Roberto Rigamonti et al. "Learning separable filters". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2013, pp. 2754–2761.

[88] Christian Szegedy et al. "Inception-v4, inception-resnet and the impact of residual connections on learning". In: *Thirty-first AAAI conference on artificial intelligence*. 2017.

[89] Shibani Santurkar et al. "How does batch normalization help optimization?" In: *Advances in neural information processing systems* 31 (2018).

[90] Wenjie Luo et al. "Understanding the effective receptive field in deep convolutional neural networks". In: *Advances in neural information processing systems* 29 (2016).

[91] Kaiming He et al. “Spatial pyramid pooling in deep convolutional networks for visual recognition”. In: *IEEE transactions on pattern analysis and machine intelligence* 37.9 (2015), pp. 1904–1916.

[92] Mete Ahishali et al. “Advance warning methodologies for covid-19 using chest x-ray images”. In: *IEEE Access* 9 (2021), pp. 41052–41065.

[93] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[94] Mohammad Hossin and Md Nasir Sulaiman. “A review on evaluation metrics for data classification evaluations”. In: *International journal of data mining & knowledge management process* 5.2 (2015), p. 1.

[95] Soumya Ranjan Nayak et al. “Application of deep learning techniques for detection of COVID-19 cases using chest X-ray images: A comprehensive study”. In: *Biomedical Signal Processing and Control* 64 (2021), p. 102365.

[96] Tej Bahadur Chandra et al. “Coronavirus disease (COVID-19) detection in chest X-ray images using majority voting based classifier ensemble”. In: *Expert systems with applications* 165 (2021), p. 113909.

[97] M Zaki, Khalid Amin, and Ahmed Mahmoud Hamad. “COVID-19 Detection Based on Chest X-Ray Image Classification using Tailored CNN Model”. In: *IJCI. International Journal of Computers and Information* 8.2 (2021), pp. 100–108.

[98] M Zaki, Khalid Amin, and Ahmed Mahmoud Hamad. “Multiscale aware classification of COVID-19 from Chest X-Ray using a spatially weighted atrous spatial pyramid pooling CNN”. In: *submitted* ().

موجز الرسالة

الجائحة الناجمة عن فيروس كورونا (COVID-19) تسببت في التهابات شديدة في المجاري التنفسية انتشرت بسرعة من خلال الاتصال بالأفراد المصابين، مما أدى إلى فقدان فادح للأرواح وتدمير اقتصادي على نطاق عالمي. وقد وضعت معدلات الانتقال العالية ضغطًا هائلًا على أنظمة الرعاية الصحية لتطوير طرق سريعة ودقيقة لتشخيص المرض. وقد أظهرت الشبكات العصبية التكرارية (CNNs) نجاحًا في مهام مختلفة في الرؤية الحاسوبية، ولكنها تعتبر متغيرة المقياس ومكلفة حسابيًا. في هذه الرسالة، اقترحنا تصميمات جديدة لاستخراج المعالم والتصنيف متعددة المقاييس، بالإضافة إلى تصميم خفيف الوزن لتشخيص COVID-19.

يستغل النموذج CNN-I الخفيف المقترح الفصلية المصفوفية للنواة الفضائية لتقليل عدد معاملات التدريب بشكل كبير، ويتم تنظيم النموذج ليتعلم فقط النوى الخطية. وتستخدم اتصالات البقاء والتعادل على مجموعات كبيرة للحفاظ على استقرار الشبكة وتقليل التكيف المفرط. قمنا بتدريب هذا التصميم الخفيف على مجموعة البيانات المرجعية QaTa-Cov19، وحققنا دقة وحساسية ودقة و F1-score بنسبة 100٪. بعدد معاملات تدريب قدره 150 ألف فقط، وهو أقل بكثير من الأساليب الأخرى في الأدبيات. وكعمل مستقبلي، يمكن استكشاف الاهتمام والاهتمام بالسياق لتعزيز الأداء بشكل أفضل، ويمكن أن يكون تقييم الانحراف الزمني في سياق الفصلية الفضائية مفيداً.

التصميم الثاني الذي قدمناه، CNN-II، يتعلم الميزات متعددة المقياسات باستخدام هرم من نوى التبعية المشتركة بمعدلات انحراف مختلفة، مما يجعله مقاييسًا. يتم استخدام آلية قائمة على الاهتمام لتوجيه واختيار المقياس الصحيح لكل إدخال. CNN-II هو شبكة قابلة للتدريب من البداية إلى النهاية تستغل تقنية التعزيز الجديدة، التعزيز النسيجي، لتقليل التكيف المفرط. حقق هذا التصميم F1-score بنسبة 0.9929 عند اختباره على مجموعة بيانات المرجعية QaTa-Cov19، مع إجمالي 5،040،571 معاملًا قابلًا للتدريب. نقترح أن SWASPP (التجويف الهرمي الفضائي التجويف الهرمي الفضائي للتجميع) يمكن أن يظهر أداءً عظيمًا للتشريح، وخاصة الانحراف الزمني الناشئ في الأدبيات الخاصة بالتشريح. بالإضافة إلى ذلك، يمكن توسيع هذا العمل لتصنيف أنواع مختلفة من الالتهاب الرئوي.

في الخلاصة، تقدم هذه الرسالة العلمية تصاميم جديدة لتشخيص COVID-19 تتعامل مع القيود التي تواجه نماذج CNN التقليدية. حققت هذه التصاميم دقة عالية مع تقليل التكلفة الحسابية وعدد المعاملات. يمكن للأبحاث المستقبلية استكشاف آليات الانتباه وتقييم استخدام الانحراف الزمني الناشئ في السياق التبعي لتحسين الأداء. لدى هذا العمل القدرة على تحسين تشخيص COVID-19 والمساعدة في تطوير طرق سريعة وفعالة لمكافحة الأوبئة المستقبلية.

وتقسم الرساله على النحو التالى:

- **الفصل الاول**: يحتوى على مقدمة عن مرض كوفيد19 والمشكلة التي تبحثها الرسالة. أيضا يتم عرض الطريقة المقترحه بأختصار و الاسهامات التى تقدمها الرسالة فى هذا المجال.
- **الفصل الثاني**: يعرض خلفية عامة عن الخوارزميات التي سيتم استخدامها لاقتراح حل للتحسين التنبؤ مرض كوفيد19.
- **الفصل الثالث**: هذا الفصل يعرض الطرق المقترحة سابقا والنتائج التي توصلوا اليها ومميزات وعيوب كل طريقة منهم.
- **الفصل الرابع**: يقدم وصفًا تفصيليًا عن اطار العمل المقترح الاول للتنبؤ بمرض كوفيد19 وأيضا كيف يتم تطبيقه على أحد البيانات الموجوده.
- **الفصل الخامس**: يقدم وصفًا تفصيليًا عن اطار العمل المقترح الثانى للتنبؤ بمرض كوفيد19.
- **الفصل السادس**: يوضح ويناقش النتائج التي حققتها المقترحات.
- **الفصل السابع**: يحتوي على عرض لما تم إنجازه في الرساله وكذلك بعض النقاط المقترحة لدراسات المستقبلية.

نبذة الرسالة

كوفيد19 هو التهاب حاد في الجهاز التنفسي. يمكن أن ينتشر كوفيد19 الناجم عن SARS-CoV-2 بسهولة من خلال الاتصال بشخص مصاب. إن التزايد الشديد للعدوى بفيروس SARS-CoV-2 لم يهدر الأرواح فحسب، بل أدى أيضًا إلى إلحاق أضرار جسيمة بالنظم المالية في كل من البلدان النامية والمتقدمة. هذا المعدل العالى فى الانتشار يضع ضغط العالي على أنظمة الرعاية الصحية مما يزيد من الحاجة إلى طرق سريعة لتشخيص هذا المرض. تُظهر الشبكات العصبية التلافيفية (CNN) نجاحًا كبيرًا في مهام الرؤية الحاسوبية المختلفة. ومع ذلك ، يعد CNN نموذجًا متغيرًا scale-invariant ومكلفًا من الناحية الحسابية. في هذه الرسالة، تم اقتراح هياكل جديدة لاستخراج الميزات متعددة النطاقات وتصنيفها ونموزج خفيفة الوزن فى الحسابات لتشخيص كوفيد19. يستغل النموذج I المقترح وهو نموذج CNN خفيف الوزن فى الحسابات قابلية فصل kernel المكانية لتقليل عدد متغيرات التدريب إلى حد كبير وتنظيم النموذج لتعلم kernel خطية فقط. علاوة على ذلك ، يستخدم هذا النموذج الاتصال BatchNorm Residual وتطبيع الدُفعات على نطاق واسع للحفاظ على استقرار الشبكة أثناء عملية التدريب وتزويد النموذج بتأثير التنظيم Regularization لتقليل التجهيز الزائد Overfitting . يتعلم CNN II المقترح ميزات متعددة النطاقات Multiscale باستخدام هرم Convolution Atrous المشتركة بمعدلات Dilation مختلفة. يستخدم CNN آلية قائمة على الانتباه Attention تُستخدم لتوجيه واختيار المقياس الصحيح لكل إدخال. إن شبكة CNN II المقترحة هي شبكة تدريب شاملة وتستغل تقنية زيادة جديدة ، وهي Augmentation Texture ، لتقليل التجهيز الزائد. يتم تدريب البنية خفيفة الوزن باستخدام مجموعة البيانات المعيارية QaTa-Cov19 التي تحقق 100 ٪ من الدقة والحساسية والدقة ودرجة F1 مع عدد معلمات منخفض جدًا (150K) مقارنة بالطرق الأخرى في الاعمال السابقه. حققت الطريقة المقترحة II 0.9929 لدرجة F1 التي تم اختبارها على مجموعة البيانات المعيارية QaTa-Cov19 بإجمالي 5،040،571 متغير قابلة للتدريب.

السيرة الذاتية لمقدم الرسالة

- **الوظيفة:** معيد بقسم تكنولوجيا المعلومات
- **جهة العمل:** كلية الحاسبات والمعلومات – جامعة المنوفية
- **تاريخ الميلاد:** 2 / فبراير / 1997
- **العنوان** : سملا - مركز قطور - محافظه الغربيه
- **الدرجة العلمية:** بكالوريوس الحاسبات والمعلومات تخصص تكنولوجيا المعلومات
- **التقدير:** امتياز مع مرتبة الشرف
- **جهة منح الدرجة العلمية:** كلية الحاسبات والمعلومات – جامعة المنوفية
- **تاريخ التخرج:** مايو 2019
- **تاريخ التسجيل لدرجة الماجستير:** 25 / فبراير / 2021