applied sciences

MDPI

Article

COVID-19 Diagnosis in Chest X-rays Using Deep Learning and Majority Voting

Marwa Ben Jabra [1], Anis Koubaa [1,2], Bilel Benjdira [1,3,*], Adel Ammar [1] and Habib Hamam [4]

[1] Robotics and Internet-of-Things Unit (RIoTU) Lab, Prince Sultan University, Riyadh 12435, Saudi Arabia; mbenjbara@riotu-lab.org (M.B.J.); akoubaa@psu.edu.sa (A.K.); aammar@psu.edu.sa (A.A.)
[2] CISTER, INESC-TEC, ISEP, Polytechnic Institute of Porto, 4200-465 Porto, Portugal
[3] SEICT Lab, LR18ES44, Enicarthage, University of Carthage, Tunis 1054, Tunisia
[4] Faculty of Engineering, University of Moncton, Moncton, NB E1A 3E9, Canada; Habib.Hamam@umoncton.ca
* Correspondence: bbenjdira@psu.edu.sa

Abstract: The COVID-19 disease has spread all over the world, representing an intriguing challenge for humanity as a whole. The efficient diagnosis of humans infected by COVID-19 still remains an increasing need worldwide. The chest X-ray imagery represents, among others, one attractive means to detect COVID-19 cases efficiently. Many studies have reported the efficiency of using deep learning classifiers in diagnosing COVID-19 from chest X-ray images. They conducted several comparisons among a subset of classifiers to identify the most accurate. In this paper, we investigate the potential of the combination of state-of-the-art classifiers in achieving the highest possible accuracy for the detection of COVID-19 from X-ray. For this purpose, we conducted a comprehensive comparison study among 16 state-of-the-art classifiers. To the best of our knowledge, this is the first study considering this number of classifiers. This paper's innovation lies in the methodology that we followed to develop the inference system that allows us to detect COVID-19 with high accuracy. The methodology consists of three steps: (1) comprehensive comparative study between 16 state-of-the-art classifiers; (2) comparison between different ensemble classification techniques, including hard/soft majority, weighted voting, Support Vector Machine, and Random Forest; and (3) finding the combination of deep learning models and ensemble classification techniques that lead to the highest classification confidence on three classes. We found that using the Majority Voting approach is an adequate strategy to adopt in general cases for this task and may achieve an average accuracy up to 99.314%.

Keywords: COVID-19; X-ray; deep learning; classification; majority voting; Pneumonia; VGGNet; EfficientNet; ResNet; MobileNet; inception; densenet

Citation: Ben Jabra, M.; Koubaa, A.; Benjdira, B.; Ammar, A.; Hamam, H. COVID-19 Diagnosis in Chest X-rays Using Deep Learning and Majority Voting. *Appl. Sci.* **2021**, *11*, 2884. https://doi.org/10.3390/app11062884

Academic Editor: Anton Civit

Received: 14 February 2021
Accepted: 17 March 2021
Published: 23 March 2021

1. Introduction

Since December 2019, the world has been plagued with uncertainty and devastation relating to a novel virus, SARS-CoV-2 (Severe Acute Respiratory Syndrome Corona Virus 2), which causes the coronavirus disease 2019, COVID-19 (Corona Virus Disease 2019). The COVID-19 pandemic has had a profound economic and social impact on most countries. As of 8 May 2020, the virus has claimed almost 300,000 lives and infected almost four million people throughout the world.

SARS-CoV-2 is not the deadliest virus in contemporary history. Ebola is significantly more deadly, reaching a fatality rate of 50% of infected people. Coronaviruses that cause Severe Acute Respiratory Syndrome (SARS) and Middle East Respiratory Syndrome (MERS) result in death in approximately 10% and 30% to 40% of cases, respectively. In fact, what makes COVID-19 particularly dangerous is that seeing that it does not immediately ravage the body, it remains active within the host for a longer period of time, therefore increasing the probability of contamination. The virus can attack the upper respiratory airways, often exhausting the host's immune system.

Researchers around the world were urged to tackle challenges related to this virus, which remains an enigma for the most part. Ongoing intensive works include and are not limited to the sequencing of the virus, screening, related medical treatment, and the necessary vaccine. Our contribution concerns the screening of the virus.

This research's motivation is to provide a reliable diagnosis system to support the decision-making of medical experts in the detection COVID-19 virus. Since the coronavirus mainly attacks the respiratory system, the diagnosis of chest X-rays has emerged as a viable solution for the detection of COVID-19 infection. In this paper, we contribute to the state-of-the-art by a comprehensive study that compares and combines 16 classifiers to develop a reliable inference system that can detect the COVID-19 virus from chest X-rs with high confidence.

1.1. COVID-19 Diagnosis in Chest X-rays Images

Several approaches have been used to screen for the virus that causes COVID-19. We opted for screening by analyzing the medical image. Medical image analysis includes image acquisition, detection, segmentation, recognition, classification, diagnosis, and follow-up.

We focused on virus screening through image recognition. This recognition is undertaken through classification among various viruses. We profited from the progress conducted on deep learning [1–7]. Deep Learning is a sub-field of machine learning dealing with algorithms in tune with the structure and function of the brain-known as artificial neural networks. Deep learning builds features automatically based on training data. It combines feature extraction and classification. For feature extraction and image classification, the Convolutional Neural Network (CNN) turned out to be the neural network offering the most promising avenue for deep learning. This avenue branches off in several structures of the network, such as AlexNet, VGGNet, ResNet, Inception, and EfficientNet. One structure may be implemented by more than one algorithm. The main interesting quality of deep learning is that it can be composed and extended in various ways to solve a variety of more complex tasks. By using this quality, we contributed, among others, to the adaptation of a list of deep learning algorithms to our specific application of COVID-19 detection.

In this paper, the main challenges are the identification of Coronavirus cases in blurred X-ray images and the differentiation of these cases of COVID-19 from other pneumonia cases, like MERS and SARS [8], bearing in mind that they have a high degree of similarity. Second, the COVID-19 virus does not have a fixed shape, circular, for example, inside the human tissue. Third, the decision resulting from the detection process is very delicate and can put human beings at risk. False-Negative means, in our situation, a person infected by the virus, who is declared by our process safe and sound, while the patient may be at risk of death, and even worse: before dying, he/she may infect thousands of people. Thus, the rate of False-Negatives should be literally zero.

The originality of our work lies in the following aspects: First, we composed a dataset enabling deep training within COVID-19 related images, and we improve the performance of the dataset by removing duplicate images giving our deep neural network models additional opportunities to learn unbiasedly the different patterns existing in the data. Second, we considered and trained 16 state-of-the-art deep learning models to classify X-ray images into three classes, normal, pneumonia, and COVID-19. Third, we selected the five best classifiers and combined them using five voting approaches (hard/soft majority, weighted voting, Support Vector Machine, and Random Forest) improve the classification accuracy. We found that the ensemble classification to with the hard voting approach achieves the best accuracy up to 99.314%, by leveraging the combination of the classifiers. To the best of our knowledge, this is the first work that evaluates 16 classifiers and 5 voting approaches for the classification of COVID-19 from X-rays.

1.2. Related Works

In the literature, there are several works that used deep learning classifiers to detect patients infected with COVID-19 [9,10]. Xu et al. [11] found that the characteristics of Computed Tomography (CT) imaging of COVID-19 are different from other types of viral pneumonia (such as Influenza-A viral pneumonia). They used multiple CNN models to classify CT images, calculate the infection probability of COVID-19, and assist in the early screening of patients with COVID-19. They collected a total of 618 CT samples: 219 from 110 patients with COVID-19; 224 CT samples from 224 patients with Influenza-A viral pneumonia; and 175 CT samples from healthy people. Then, they pre-processed the CT images to extract effective pulmonary regions. Then, they used a 3D CNN model based on ResNet18 to segment multiple candidate image cubes. They chose a 3D image classification model to be able to categorize all the image patches. The location attention classification model uses the relative distance from-edge as extra weight for the model. This classification model aims to learn the relative location information of the patch on the pulmonary image. They acquired a total of 11,871 image patches, including 2634 COVID-19, 2661 Influenza-A-viral-pneumonia, and 6576 irrelevant-to-infection. Finally, they used the Noisy or Bayesian function to calculate the infection type (COVID-19, Influenza-A-viral-pneumonia, or no-infection-found) and the total confidence score of the CT case. They only compared the average f1-score for the first two classes, which showed an enhancement of 4.7%, with an overall classification accuracy of 86.7% for all three groups.

Because there is no automatic tool to quantify the infection volume for COVID-19 patients clinically, Shan et al. [12] proposed to develop a Deep Learning-based system called "VB Net" neural network for automatic segmentation and quantification of COVID-19 infection regions in chest CT [13]. This system also aims to accurately estimate the shapes, volumes, and percentage of the infection region. The "VB Net" model is a combination between the V-Net model and the bottleneck model. The V-Net extracts global image features using down-sampling and convolution operations, and the bottleneck model integrates fine-grained image features using up-sampling and convolution operations. Compared with V-Net, the speed of VB-Net is much higher because of the bottleneck structure. The system is trained using 249 COVID-19 patients' data and validated on 300 new COVID-19 patients. To accelerate the delineation of COVID-19 CT images used for training, which is very time-consuming, they proposed a human-in-the-loop (HITL) strategy to generate the training samples iteratively. This method assists radiologists to refine the automatic annotation of each case. To evaluate the performance of the DL (Deep Learning) based system, the Dice similarity coefficient, the differences of volume, and the percentage of infection (POI) are calculated between automatic and manual segmentation results on the validation set. The proposed system gave dice similarity coefficients of 91.6 $\pm$ 10.0% between automatic and manual segmentation and a mean POI estimation error of 0.3% for the whole lung on the validation dataset. The proposed human in the loop strategy reduces the delineation time to 4 min after three iterations of model updating. The segmentation accuracy of deep learning models was evaluated using the Dice similarity coefficient on the entire 300 validation set. It has improved from 85.1 $\pm$ 11.4%, to 91.0 $\pm$ 9.6%, and 91.6 $\pm$ 10.0% with more training data added. The improved segmentation accuracy greatly reduces human intervention and, thus, significantly reduces the time of annotation and labeling.

Many studies confirm that among the characteristics of the patients infected with COVID-19 is that they present abnormalities in their chest X-ray images [14,15]. Motivated by the need for faster interpretation of radiography images, Wang et al. [16] proposed a deep convolutional neural network design (COVID-Net), to detect the COVID-19 cases from chest radiography X-ray images. They used the open-source COVIDx dataset; it comprises 16,756 chest radiography images from 13,645 patient cases from two open access data repositories. More specifically, the COVIDx dataset contains only 76 radiography images from 53 COVID-19 patient cases, while there are significantly more patient cases and corresponding radiography images with Normal and Non-COVID-19 pneumonia. In total, there are 8066 normal patient cases and 5526 cases of non-COVID-19 pneumonia patients.

The COVID-Net network architecture uses a lightweight residual projection-expansion-projection-extension (PEPX) design pattern. The first-stage projection is composed of 1×1 convolutions for projecting input features to a lower dimension. The expansion stage is composed of 1×1 convolutions for expanding features to a higher dimension that is different than that of the input features. The Depth-wise representation uses efficient 3×3 depth-wise convolutions for learning spatial characteristics to minimize computational complexity. The second-stage projection is composed of 1×1 convolutions for projecting features back to a lower dimension. Finally, an extension is composed of 1×1 convolutions that extend channel dimensionality to a higher dimension to produce the final features. The COVID-Net balances accuracy and computational complexity by achieving 92.4% test accuracy, while requiring just 2.26 billion MAC (Multiplier Accumulator)operations to perform case predictions.

In another approach, Duran-Lopez et al. [17] proposed to diagnose COVID-19 cases from X-ray images using a set of pre-processing algorithms followed by a designed CNN (COVID-XNet) in order to distinguish COVID-19 cases from normal cases at an average accuracy of 94.43% and an AUC (Area Under Curve) of 0.988. They also went deeper by analyzing the extracted features from COVID-XNet using the Class Activation Maps (CAM). This helped to localize precisely the COVID-19 infected areas inside the screened lungs. The localization accuracy was assessed qualitatively by a lung specialist and confirmed to be efficient and accurate.

Leveraging fractional-order (FO) calculus techniques, Sahlol et al. [18] proposed a COVID-19 X-ray classification method that uses a pre-trained CNN (Inception [19]) as feature extractor, followed by an improved swarm-based meta-heuristic optimization technique (Marine Predators Algorithm [20]) to select only relevant features. They achieved up to 99.6% accuracy on the binary classification problem on the dataset made public by Chowdhury et al. [21]. While we used the same dataset, the problem that we address in this paper is more challenging since it also considers a third class of other pneumonia cases. More recently, in order to mitigate the problem of the small size of available COVID-19 datasets, Karakanis and Leontidis [22] used a conditional generative adversarial network (cGAN [23]) for data augmentation. Accordingly, they generated realistic synthetic images only for the under-represented COVID-19 class, since the two other classes had a sufficient number of original images. Then, they tested two ad hoc lightweight deep-learning models on the augmented dataset. They obtained an accuracy of 98.7% and 98.3% on the binary and 3-class problems, respectively, on a small balanced dataset (275 images for each class) that they extracted from originally unbalanced datasets made public by References [24,25]. In a similar approach, Zebin and Rezvy [26] used a different type of generative adversarial network (a CycleGAN [27]) for augmenting the number of COVID-19 images, then tested several pre-trained CNN-based feature extractors. They attained a maximum accuracy of 96.8% with EfficientNet-B0 architecture [28] on a selected dataset where COVID-19 images were taken from Reference [24], while normal and other pneumonia images were taken from Reference [25]. Nevertheless, such a heterogeneous dataset may introduce some bias in the results. Moreover, we will show in Section 3.3 that we can reach a higher accuracy without resorting to supplementary synthetic images.

To differentiate COVID-19 cases from other pneumonia cases, Farooq et al. [29] proposed to build open source and open access chest X-rays datasets and presented an accurate Convolutional Neural Network framework [30]. They also used an updated version of the COVIDx dataset recently made public by the authors of the COVID-Net [16] previously described. It consists of a total of 5941 chest radiography images from 2839 patients with four classes. There are 68 COVID-19 radiographs from 45 COVID-19 patients. There were a total of 1203 patients with negative pneumonia: normal class, 931 patients with bacterial pneumonia, and 660 patients with non-COVID-19 viral pneumonia cases. To solve the imbalanced classification problem caused by the small number of COVID-19 images, they proposed to use data augmentation techniques. The transformations used included vertical flips of the training images, random rotation of the images (maximum rotation angle was

15 degrees), and lighting conditions. They chose to employ the ResNet50 model for the classification task because it provides a good trade-off between performance and number of parameters, has proved faster training, and it is possible to produce images with different sizes than the training images. The ResNet50 model is pre-trained on the ImageNet dataset [31] and fine-tuned with the COVIDx dataset. The input images are resized to $128 \times 128 \times 3$, $224 \times 224 \times 3$, and $299 \times 299 \times 3$ pixels and are employed in different training stages. For training a high-performance network with very few epochs, they used the transfer learning techniques introduced in Fastai [32]. This technique replaces the head of the trained model by another containing a sequence of Adaptive average/max pooling, batch normalization, drop out, and linear layers. The resultant network is called COVID-ResNet. This approach achieved a state-of-the-art accuracy of 96.23% on the COVIDx dataset with only 41 epochs and 25.6 M parameters.

To study the application of the COVID-19 detection application based on deep learning models from the chest X-ray images, Minaee et al. [33] started by preparing a dataset of 5000 chest X-rs from the publicly available datasets. Then, they used Transfer learning on a subset of 2000 radiograms to train 4 CNN models, including ResNet18, ResNet50, SqueezeNet, and DenseNet-121, to identify COVID-19 disease in the analyzed chest X-ray images. Finally, they evaluated these models on the 3000 images. Most of these networks achieved a sensitivity rate of 98% ($\pm$3%) and a specificity rate of around 90%.

Table 1 summarizes the datasets, algorithms, and results of the most similar related works on COVID-19 detection, compared to the present paper. In this table, row 1 will refer to the work introduced by Reference [11]; row 2 will refer to the work introduced by Reference [12], row 3 will refer to the work introduced by Reference [16], row 4 will refer to the work introduced by Reference [29], row 5 will refer to the work introduced by Reference [33], and the last row will describe the output generated by our method.

The remainder of the paper is organized as follows. Section 2 describes the methods and the materials used in this study: the characteristics of the COVID-19 dataset, the data collection and the cleaning processes, and, finally, the 16 deep learning models selected. Section 3 discusses the main results. Section 4 concludes the paper and outlines future works.

Table 1. Comparison of our paper with the related works.

Ref.	Dataset Used	Algorithms	Main Results			
				Accuracy	Recall	Precision
1 [11]	A total of 618 CT samples: 219 CT samples with COVID-19; 224 CT samples with Influenza-A viral pneumonia; and 175 CT samples from healthy people. 528 CT samples for training 90 CT samples for validation	3D CNN model based on the classical ResNet18 network structure.	Normal	86.7%	90.0%	93.1%
			Non-COVID-19		83.3%	86.2%
			COVID-19		86.7%	81.3%
2 [12]	300 CT images of COVID-19 cases for validation. 249 CT images of COVID-19 cases for training.	The "VB Net" model is a combination between the V-Net model and the bottleneck model for automatic segmentation and quantification	Normal	91.6%	-	-
			Non-COVID-19		-	-
			COVID-19		-	-
3 [16]	COVIDx dataset : 16,756 chest-X-ray images from 13,645 patient cases: 76 images from 53 COVID-19 patient cases, 8066 patient normal cases, 5526 patient non-COVID-19 pneumonia cases	The COVID-Net network architecture uses a lightweight residual (PEPX) design pattern.	Normal	93.3%	95.0%	90.5%
			Non-COVID-19		94.0%	91.3%
			COVID-19		91.0%	98.9%
4 [29]	A total of 5941 posteroanterior chest-X-ray images from 2839 patients: 68 COVID-19 images from45 COVID-19 patients. 1203 patients with Normal class, 931 patients with a bacterial pneumonia 660 patients with non-COVID-19 viral pneumonia cases.	The ResNet50 model for classification.	Normal	96.2%	96.5%	99.1%
			Non-COVID-19		93.9%	92.7%
			COVID-19		100.0%	100.0%
5 [33]	COVID-X-ray-5k: It contains 2084 training and 3100 test images. - 2084 images for training divided into 84 COVID-19 and 2000 Non-COVID - 3100 images for testing divided into 100 COVID-19 and 3000 Non-COVID.	4 CNN models, including ResNet18, ResNet50, SqueezeNet, and DenseNet-121, to identify COVID-19 disease in the analyzed chest X-ray images.	Normal	89.5%	-	-
			Non-COVID-19		-	-
			COVID-19		-	-
Our paper	The chest X-ray dataset contains 2911 images divided into: 237 COVID-19 positive images, 1338 normal images, 1336 viral pneumonia images. 2328 images for the training, 291 images for the validation, 292 images for testing.	MobileNet-V2, Xception, Inception-V3, DenseNet-201, VGG16, Resnet-50 (V1 and V2), Resnet 101, and EfficientNet (B0, B1, B2, B3, B4, B5, B6, and B7), and Ensemble classification using voting.	Normal	99.31%	100.0%	98.52%
			Non-COVID-19		98.50%	100.0%
			COVID-19		100.0%	100.0%

2. Materials & Methods

In this paper, we propose to detect and differentiate the COVID-19 cases from other pneumonia and normal cases using deep learning algorithms based on chest X-ray images. Our proposed method is composed of 5 steps. In the first step, we started by preparing the Chest X-ray Dataset. In the second step, we trained the selected classification algorithms on the constructed dataset. The selected classification algorithms were: MobileNetV2, Xception, InceptionV3, DenseNet-201, VGG16, ResNet50 (V1 and V2), and ResNet11 EfficientNet (B0, B1, B2, B3, B4, B5, B6, and B7). In the third step, we selected the five best-performing algorithms and tested five different voting approaches to pick out the best strategy to consider in general cases. In this section, we will explain these three steps in more detail.

2.1. Dataset Preparation

In this paper, we used the Chest X-ray Dataset for the detection of COVID-19 cases that was recently made public by Chowdhury et al. [21]. This dataset was made by a team of researchers from Qatar University, Doha, and the University of Dhaka, Bangladesh, along with their collaborators from Pakistan and Malaysia in collaboration with medical doctors. The inclusion exclusion criteria is common for all COVID-19 dataset.

The dataset is composed of chest X-ray images that merely present the anterior-posterior views because based on board-certified radiologist advice. Only anterior-posterior images are kept for COVID-19 prediction since the other type of images is not suitable for this purpose [33].

It contains chest X-ray images for COVID-19 positive cases, along with Normal and Viral Pneumonia images. It consists of a total of 2992 chest radiography images with three classes: 306 COVID-19 positive images, 1341 normal images, and 1345 viral pneumonia images.

To improve the Chest X-ray Dataset, we removed duplicate images found in the original dataset. We found that it contains 77 duplicate images. The new dataset contains 2911 images divided into 237 COVID-19 positive images, 1338 normal images, and 1336 viral pneumonia images. As observed, the COVID-19 cases are significantly lower than other classes, making it an imbalanced classification problem. Figure 1 presents some images of the chest X-ray dataset.

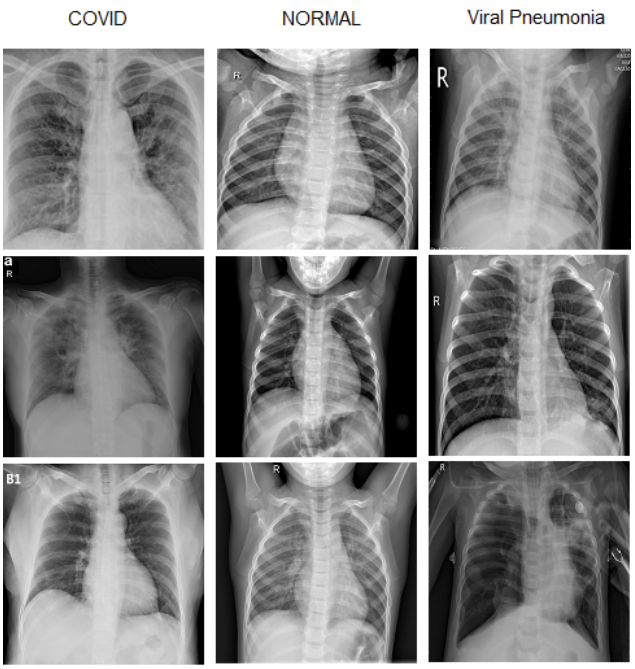

Figure 1. Sample images from the chest X-ray dataset.

For the evaluation of our deep learning model, we split our dataset into 2328 images for the training, 291 images for validation (dev), and 292 images for testing. For the training, we performed data augmentation on the dataset and fixed the input size of the image to $224 \times 224 \times 3$.

2.2. Training of the Selected Algorithms

Among the state of the art algorithms in image classification, we selected 16 classifiers which are: MobileNetV2 [34], Xception [35], InceptionV3 [19], DenseNet-201 [36], VGG16 [37], ResNet (ResNet50V1, ResNet50V2, ResNet11) [38], and EfficientNet (B0, B1, B2, B3, B4, B5, B6, and B7) [28]. These classifiers are considered among the most popular CNN architectures used in literature based on the recent survey made by Khan et al. [39]. We wanted to consider this large number of classifiers to be able to reach the maximum

possible accuracy on COVID-19 diagnosis task independently from the chosen CNN architecture. For every selected model, we used the pre-trained weights on the ImageNet dataset as a start point for the training. In fact, the constructed dataset is so small to be sufficient to learn discriminative features for general visual patterns. Big datasets, like ImageNet, help the model to learn better general visual patterns that exist in image data. Using pre-trained weights on ImageNet for training small datasets helps the model to converge faster and easier. Although, in our case, the type of chest X-ray Images is different from the type of images existing in ImageNet. But, in literature, there are no big datasets of chest X-ray images that we may use to pre-train our model on. ImageNet was the most adequate dataset in our case. During the training of our model, all the layers were set to be trainable. We did not freeze any part of the model. In fact, our dataset's domain is notably different from the domain of ImageNet, and all the parameters of the model should be tuned to fit our dataset.

In every model, we changed the last layers of the classification part by a proposed head model composed of 7 layers. The first layer is an average pooling layer with a size of 4×4 to reduce the number of parameters, followed by a ReLU activation layer (Rectified Linear Unit) that increases the non-linearity in the images. The third layer is a batch normalization layer that improves the speed, performance, and stability of our model, followed by a ReLU activation layer. We placed the dropout 0.5 after the activation function that sets a number of hidden units to 0 with a probability of 0.5. The sixth and last layer before the classification layer is a batch normalization layer aiming to improve the whole model. Finally, the last layer is a Softmax layer with three outputs corresponding to 3 different classes (COVID-19, Normal, and Viral Pneumonia). Figure 2 presents our proposed model combined with the feature extractor of the selected model concatenated to the designed head model.

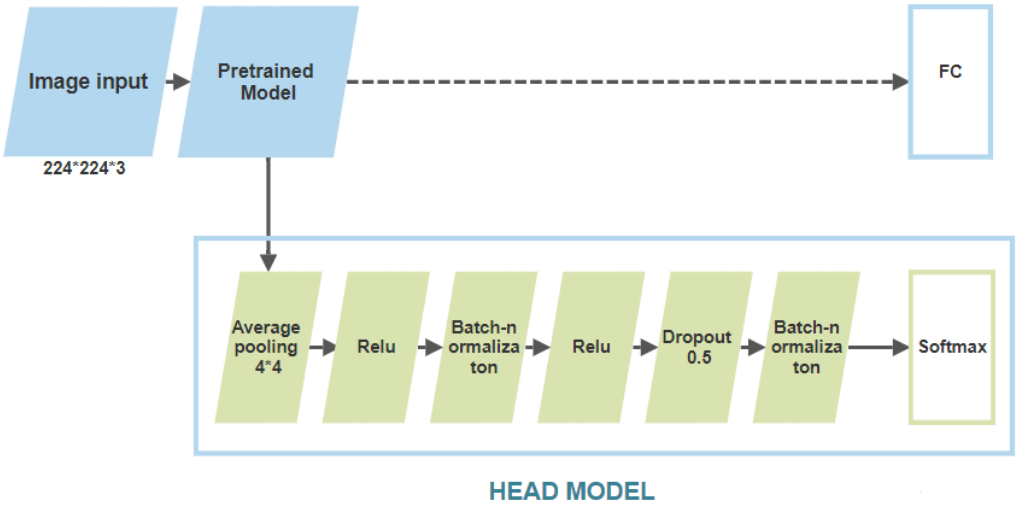

Figure 2. The proposed model architecture for training the COVID-19 chest X-ray dataset.

2.2.1. MobileNet

MobileNetV2 [34] is the second version of MobileNet architecture. This architecture contains two types of blocks. One is a residual block with a stride of 1. The other is a block with a stride of 2 for downsizing. For each block, there are three layers. The first layer is 1×1 convolution with ReLU6, the second layer is a depth-wise convolution, and the third layer is another 1×1 convolution but without any non-linearity. The architecture of MobileNetV2 is displayed in Figure 3. Every line represents a series of layers that are repeated n times, c is the number of output channels, s is the stride, and t is the expansion factor.

Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

Figure 3. MobileNet architecture [34].

To adapt the MobileNetV2 architecture for our application, we replaced the head FC (Fully Connected) layer by the designed head in our proposed architecture (Figure 2).

2.2.2. Xception

The Xception [35] architecture has 36 convolutional layers forming the feature extraction base of the network. The 36 convolutional layers are structured into 14 modules, all of them have linear residual connections around them, except for the first and last modules. In short, the Xception architecture is a linear stack of depth-wise separable convolution layers with residual connections. The data first goes through the entry flow, then through the middle flow, which is repeated eight times, and finally through the exit flow. Note that all Convolution and Separable Convolution layers are followed by batch normalization. All Separable Convolution layers use a depth multiplier of 1. A representation of the Xception architecture is illustrated in Figure 4. Data goes first into the Entry flow. Then, it reiterates into the middle flow for eight times before going into the exit flow.

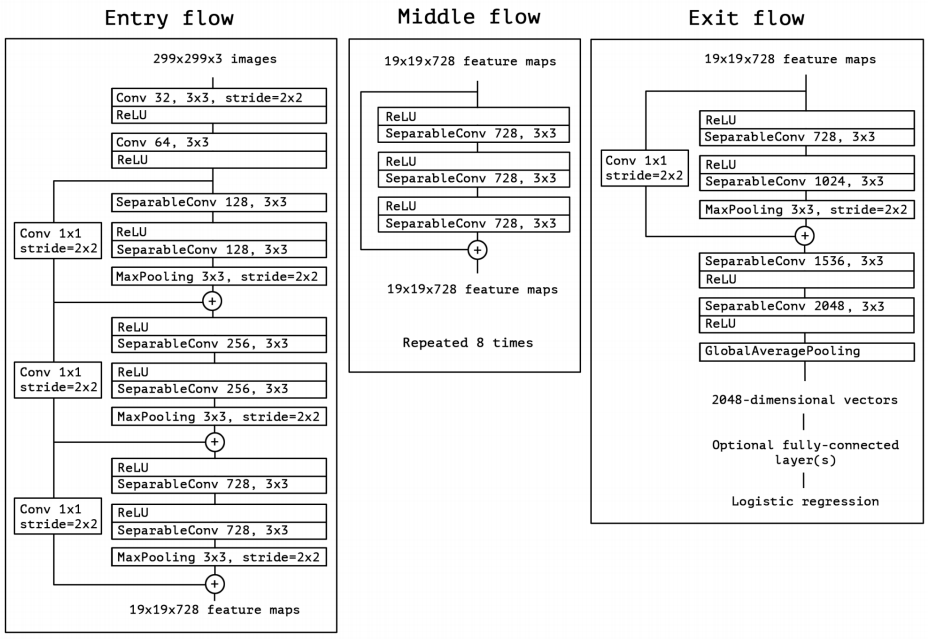

Figure 4. Xception architecture [35].

2.2.3. Inception

The InceptionV3 [19] is the last version of the Inception architecture. It allows us to increase the depth and the width of the deep learning network, while simultaneously maintaining the computational cost constant. It works as a multi-level feature generator by

computing 1 × 1, 3 × 3, and 5 × 5 convolutions with 42 layers deep. This allows the model to use multiple scales of kernels on the image and to get results from all of them. All such outputs are stacked along the channel dimension and used as input to the next layer. A representation of the of the InceptionV3 architecture is made in Figure 5.

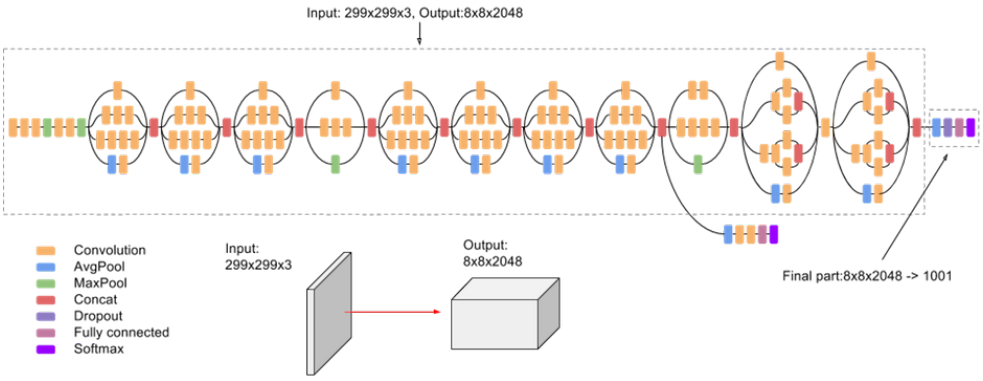

Figure 5. InceptionV3 architecture [40].

2.2.4. DenseNet

The DenseNet architecture, introduced by Huang et al. [36], is a network architecture where each layer is directly connected to every other layer in a feed-forward. The feature maps of all preceding layers are treated as separate inputs for each layer, whereas its own feature maps are passed on as inputs to all subsequent layers. This simplifies the connectivity pattern between layers introduced in other architectures. This makes it lower in the number of parameters than an equivalent traditional CNN, as there is no need to learn redundant feature maps. There are multiple variants of DenseNet following the number of layers. For example, DenseNet-201 corresponds to a variant where the number of layers with trainable weights is 201 (excluding batch normalization layers). A representation of one DenseNet architecture based on three Dense blocks is made in Figure 6.

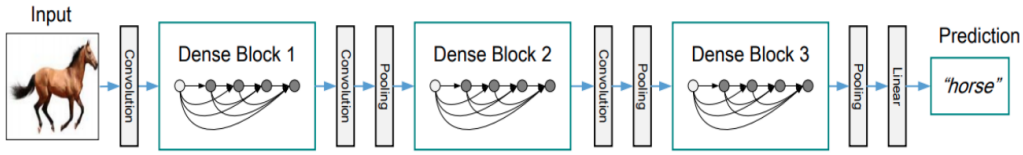

Figure 6. DenseNet architecture [36].

2.2.5. VGGNet

In the VGG16 [41] architecture, the input to the first convolutional layer is of fixed size 224 × 224 × 3 RGB (Red-Green-Blue) image. The image is passed through a stack of convolutional layers, where the filters were used with a very small receptive field: 3 × 3. One of the configurations also utilizes 1 × 1 convolution filters, which can be seen as a linear transformation of the input channels. The convolution stride is fixed to 1 pixel; the spatial padding of convolutional layer input is such that the spatial resolution is preserved after convolution, the padding is 1-pixel for 3 × 3 convolutional layers. Spatial pooling is carried out by five max-pooling layers, which follow some of the convolutional layers. Max-pooling is performed over a 2 × 2 pixel window, with stride 2. A representation of the VGG architectures is made in Figure 7.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 7. VGG architectures [41].

2.2.6. ResNet

The ResNet [38] model comes with a residual learning framework to simplify the training of deeper networks. The architecture is based on network layers' reformulation as a residual block added to the layer inputs. The ResNet network has four stages. It takes as input an image having height and width as multiple of 32 and channel width as 3 (224 × 224 × 3). Every ResNet architecture performs the initial convolution and max-pooling using 7 × 7 and 3 × 3 kernel sizes, respectively. Afterward, Stage 1 of the network has 3 Residual blocks. Every Residual block contains three layers. The kernels' size to perform the convolution operation in a residual block of stage 1 is 64, 64, and 128, respectively. The convolution operation in the Residual Block is performed with stride 2. Hence, the input size will be reduced to half in terms of height and width, but the channel width will be doubled. Figure 8 describes the most used ResNet architectures.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figure 8. ResNet architectures [38].

In the current paper, we chose three variants of ResNet (ResNet50-V1, ResNet50-V2, and ResNet11). Two major differences exist between ResNet50–V1 and ResNet50–V2.

The first difference is that ResNet50-V2 has removed the last non-linearity, therefore clearing the input path to output in the form of identity connection. The second difference is that ResNet50-V2 applies Batch Normalization and ReLU activation to the input before the multiplication with the weight matrix (convolution operation), while ResNet50-V1 performs the convolution, followed by Batch Normalization and ReLU activation.

ResNet11 contains 51 convolutional layers more that ResNet50. It also has 7.6 billion FLOPS (Floating Point Operations per Second) instead of 3.8 billion FLOPS for the ResNet50 model.

2.2.7. EfficientNet

EfficientNets [28] are a list of classifiers introduced recently in 2019 and based on AutoML and Compound Scaling. AutoML is used to develop a mobile-size baseline network (EfficientNet-B0). Then, the compound scaling method is used to scale up this baseline to obtain EfficientNet-B1 to B7. The Compound Scaling method scales uniformly all dimensions of depth, width, and resolution using a simple yet highly effective compound coefficient. The depth of layers should increase 20%, the width 10%, and the image resolution 15% to keep things as efficient as possible, while expanding the implementation and improving the accuracy. Alpha, beta, and gamma are the scaling multipliers for depth, width, and resolution, respectively. They are obtained using a grid search. Phi is a user-specific coefficient. It is a real number that controls resources. Below are the equations of depth, weight, and resolution based on Phi :

$$\textbf{Depth} : d = \alpha^{\phi}, \tag{1}$$

$$\textbf{Width} : w = \beta^{\phi}, \tag{2}$$

$$\textbf{Resolution} : r = \gamma^{\phi}, \tag{3}$$

$$while : \ \alpha.\beta^2.\gamma^2 \approx 2; \ \ \alpha \geq 1, \beta \geq 1 \, and \, \gamma \geq 1. \tag{4}$$

EfficientNet-B0 architecture is a mobile sized architecture having 11M trainable parameters. Its architecture is described in Figure 9, where every row is a separate stage i in the network. Every stage i is characterized by a number of layers $\hat{L}_i$, an input resolution size $< \hat{H}_i, \hat{W}_i >$ and an output channels size $\hat{C}_i$.

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

Figure 9. EfficientNet-B0 architecture [28].

It uses seven inverted residual blocks. Squeeze and excitation blocks are used along with the swish activation function. EfficientNet uses 7 MBConv blocks. Every MBConv block takes two inputs. The first is data, and the second is block arguments. The data is received from the last layer. The block arguments are a collection of attributes to be used inside an MBConv block, like input filters, output filters, expansion ratio, squeeze ratio, etc. The expansion phase aims to expand the layer to make it wide. The depth-wise convolution phase applies a depth-wise convolution using the kernel size mentioned in the block arguments. The Squeeze and excitation phase extracts the global features using the global average pooling. Then, it squeezes the numbers of channels using the squeeze

ratio. The Output phase applies convolution operation using the output filters mentioned in the block arguments.

2.3. Ensemble Classification

As demonstrated in the Experimental section, we got different accuracies for the 16 selected classifiers when testing them on the Chest X-ray Dataset. But, we noted that five classifiers generally outperform the others in this task. This affirmation is concluded after testing the classifiers on two different datasets (a validation and a test set). No one of them is to be selected as the best classifier in all cases (the best classifier tested in the validation set is different from the best classifier in the test set). Hence, we decided to combine results generated by every classifier following five different ensemble classification methods. First, we selected the Voting approach because it is the straightforward solution to generate final-stage classification from different predictions. We included both the soft and the hard approaches to assess the validity of both of them. Then, we selected three of the top used machine learning algorithms (Random Forests, SVM and Neural Networks) to estimate if there is a more accurate combination between the different predictions. We selected these classifiers similarly to many studies that only considered them in their research works [42–47]. Below is a more detailed description of every selected approach:

- **Majority Voting using the hard approach:** As shown in Figure 10, this method acts by summing the per class labels associated with every classifier for the input image. Then, it gives the final label to the class that has the greatest number of labels (votes) among the classifiers. If there are equal votes for two different classes, we chose to assign the final label to the class with the least index. Other strategies can be used to solve this special case, as well.
- **Majority Voting using the soft approach:** As shown in Figure 10, this method acts by summing the per class values of the probability vector generated by every classifier for the input image. Then, it gives the final label to the class that has the greatest probability sum. Equal probabilities sum is an almost impossible case for the soft approach.
- **Weighted voting using a Neural Network:** Here, we designed a more dedicated voting approach in order to give a learned weight for every classifier prediction. In fact, manually giving a weight for every classifier is not practical. To solve this problem, we decided to assign the weights using a Neural Network. The Neural Network is trained on the validation set and tested on the test set. In the end, every classifier will be assigned a conditional weight that depends on other classifiers to deduce the most accurate label for the input image.
- **SVM (Support vector machine)-based voting:** To deduce the right classification of the input image, an SVM is trained to deduce the right classification of the input image by only seeing the vector of labels assigned by the top classifiers. The training of the SVM is made on the validation set and tested on the test set.
- **Random Forests-based voting:** The Random Forests algorithm acts by building a number of decision trees during the training and generating as output the mode of the assigned classes by the individual trees. The Random Forests method has the advantage of avoiding the habit of overfitting for the normal decision tree. Here, we do the same; many decision trees are built to estimate the right label based on the labels made by the classifiers. Then, we deduce the mode of the estimations made by these decision trees. This mode will be chosen as the final label assigned to the input image.

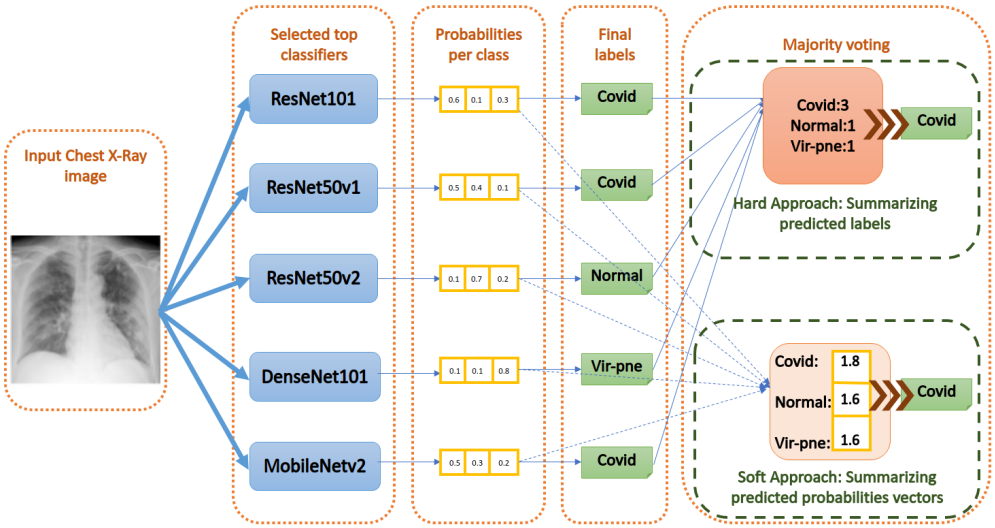

Figure 10. The ensemble classification model.

We tested every voting approach on the test set. For the two Majority Voting methods (the hard and the soft approaches), they do not need any training; they were tested directly on the test dataset. For the other three methods, we do the training of the algorithm on the validation dataset before testing it on the test dataset.

As demonstrated in the experimental section, the majority voting methods (the hard and the soft approaches) outperformed clearly the other methods. They were chosen as the best strategy to adopt for the COVID-19 diagnosis from the chest X-ray images. These two methods are illustrated in Figure 10.

3. Results

In this section, we describe the experiments run to evaluate the proposed approaches. After that, we discuss the findings.

3.1. Experimental Setup

Concerning the deep learning classifiers, we used the Tensorflow 2.1 Library [48]. We used the default Python API of the library. Models are instantiated using the Keras [49] default implementation inside Tensorflow. Concerning the Ensemble voting, we implemented the majority voting approaches (soft and hard approaches) in Python language. We used Scikit-Learn [50] library for the SVM and the Random Forests models. We used Keras [49] for the Neural Network-based Ensemble Voting. All the experiments were made in Python [51] Language, and we used Jupyter Lab [52] for easy assessment of the results. We performed the training and the testing on Google Colab Professional account. The GPUs used were P100 and T4. The size of the RAM was 100 GB. For all the algorithms used, we performed the training using the Adam optimizer and the Cross-Entropy loss function. The image input sizes for all the networks are of (224 $\times$ 224) pixels. For the learning rate, we started by 1.00 $\times 10^{-4}$, and then we made some tuning by increasing the value of the learning rate to 1.00 $\times 10^{-5}$ for all the models, except the VGG16 and EfficientNet-B0. Only for these two models, we noticed that the learning rate increase did not improve the convergence of the results.

For the number of epochs, we started by 200 epochs. Then, we increased it or decreased it depending on the convergence results and the stabilization of the Training Loss and the Accuracy Curve. In Table 2, we presented the training epoch number and the epochs number of best convergence for each deep learning model used in the experiments. The batch size is a hyperparameter of gradient descent representing the number of training samples fed to the network in one iteration before updating its parameters. Its value depends on the size of the model, the GPU memory, and the convergence of the results.

Table 2. Trainig parameters of the deep learning models.

Model	Learning Rate	Batch Size	Epochs Number of Best Convergence	Number of Epochs
MobileNetV2	1.00×10^{-5}	200	760	1000
Xception	1.00×10^{-5}	100	471	500
InceptionV3	1.00×10^{-5}	100	369	400
DenseNet-201	1.00×10^{-5}	64	270	300
VGG16	1.00×10^{-4}	64	169	200
ResNet50V1	1.00×10^{-5}	100	152	200
ResNet50V2	1.00×10^{-5}	100	231	300
ResNet11	1.00×10^{-5}	100	425	500
EfficientNet-B0	1.00×10^{-4}	32	224	300
EfficientNet-B1	1.00×10^{-5}	32	152	200
EfficientNet-B2	1.00×10^{-5}	32	280	300
EfficientNet-B3	1.00×10^{-5}	32	373	400
EfficientNet-B4	1.00×10^{-5}	32	350	400
EfficientNet-B5	1.00×10^{-5}	32	415	500
EfficientNet-B6	1.00×10^{-5}	16	252	300
EfficientNet-B7	1.00×10^{-5}	16	290	300

3.2. Performance Evaluation and Metrics

For the evaluation of our proposed algorithms, we used six metrics based on the following parameters:

- **True Positives (TP):** It represents the number of images belonging to a class "X," and the model predicts correctly that they belong to the class "X". For example, the input image is of class "Normal" and the model predicts correctly that it is of class "Normal".
- **True Negatives (TN):** It represents the number of images that do not belong to a class "X" and the model predicts correctly that they do not belong to the class "X". For example, the input image is not "COVID", and the model predicts correctly that it is not of the class "COVID".
- **False Positives (FP):** It represents the number of images belonging to a class "X" and the model falsely predicts that it belongs to another class different from "X". For example, the input image is "COVID" and the model falsely predicts it as "Normal".
- **False Negatives (FN):** It represents the number of images that do not belong to a class "X" and the model falsely predicts that they belong to the class "X". For example, the input image is not "COVID", and the model predicts it falsely as "COVID".

The batch size is a hyperparameter of gradient descent representing the number of training samples fed to the network in one iteration before updating its parameters. Its value depends on the size of the model, the GPU memory, and the convergence of the results.

The four metrics used for the evaluation are:

$$\textbf{Accuracy} = (TP + TN)/(TP + TN + FP + FN), \tag{5}$$

$$\textbf{Precision} = TP/(TP + FP), \tag{6}$$

$$\textbf{Recall} = TP/(TP + FN), \ and \tag{7}$$

$$\textbf{F1score} = \frac{2 * Precision * Recall}{(Precision + Recall)}. \tag{8}$$

For our problem: the COVID-19 diagnosis in chest X-ray images, all these defined metrics have a significant meaning and interpretation. The accuracy measures the degree of right predictions among the total predictions (right and false) of the model. We will consider during our study the overall accuracy of the model to be able to judge the global performance of the model overall the classes.

Concerning the precision, the recall, and the F1 score, we will consider only the class "COVID". This is more fruitful for our study and analysis. The precision will reflect the model's ability to only detect "COVID" cases without falsely classifying images that belong

to other class as "COVID" cases. The clinical impact of low precision is the increase in the number of classifying safe people (Normal and Viral Pneumonia cases) as COVID. It will engender more false alarms and add more surcharge for the COVID medical staff as they will attentively care for persons that did not suffer from COVID.

On the other hand, the recall (i.e., sensitivity) of the class COVID will reflect the model's ability to detect all the COVID cases that exist without assigning them to other classes. The clinical impact of a low recall will be dangerous. In fact, a lower recall rate means a higher risk to assign COVID cases to other classes and to prevent them from the special care they need. They will be more exposed to complications, and they will be at a higher risk of death. In addition, during this, they will, unconsciously, infect other people with the disease.

Besides, the F1 score of the class COVID will measure the strength of the model in treating the COVID cases (to successfully detect all the COVID cases in the dataset and not assign any non-COVID case to the class COVID). In fact, it is an equal combination between the precision and the recall metrics.

3.3. Results

For the evaluation of the proposed algorithms, we compared the values of the four metrics (Overall Accuracy, Precision for the class COVID, Recall for the class COVID, and F1 score for the class COVID) for every algorithm described in Section 3. We tested these algorithms on the chest X-ray constructed dataset: the training set is composed of 2328 images, the validation set is composed of 291 images, and the test set is composed of 292 images. Tables 3 and 4 present the metrics of every algorithm on the training set and the test set respectively. In addition, they represent the mean, the standard deviation, the Confidence Level (95.0%), and the Confidence Interval (95.0%) for all the models. The results are presented in both of the tables in an ascending order following the accuracy metrics.

Table 3. Evaluation metrics of the algorithms on the train dataset.

ALGORITHM USED	Accuracy	Precision	Recall	F1-Score
EfficientNet-B0	0.9549	0.66197	0.93925	0.79325
EfficientNet-B6	0.97466	0.84821	1	0.91787
EfficientNet-B2	0.98969	0.89623	1	0.94527
EfficientNet-B4	0.99012	0.96447	1	0.98191
EfficientNet-B3	0.99527	0.96447	1	0.98191
EfficientNet-B5	0.99871	1	1	1
VGG16	0.99914	1	1	1
EfficientNet-B1	0.99957	1	1	1
ResNet50V2	0.99957	1	1	1
DenseNet-201	0.99957	1	1	1
Xception	0.99957	1	0.99474	0.99736
MobileNetV2	1	1	1	1
InceptionV3	1	1	1	1
ResNet50V1	1	1	1	1
ResNet11	1	1	1	1
EfficientNet-B7	1	1	1	1
Mean	0.99379	0.95845	0.99587	0.97609
SD	0.01235	0.09054	0.01515	0.05416
Confidence Level (95.0%)	0.00658	0.04825	0.00807	0.02886
Confidence Interval (95.0%)	0.98721–1	0.91020–1	0.98779–1	0.94723–1

Table 4. Evaluation metrics of the algorithms on the test dataset.

ALGORITHM USED	Accuracy	Precision	Recall	F1-Score
EfficientNet-B0	0.92466	0.55814	1	0.71642
EfficientNet-B3	0.96233	0.82759	1	0.90566
EfficientNet-B4	0.96233	0.82759	1	0.90566
VGG16	0.96575	1	1	1
EfficientNet-B2	0.97603	0.85714	1	0.92308
EfficientNet-B5	0.97603	1	1	1
EfficientNet-B6	0.97603	0.88889	1	0.94118
EfficientNet-B7	0.97603	0.96	1	0.97959
Xception	0.97945	1	1	1
InceptionV3	0.98288	1	1	1
EfficientNet-B1	0.9863	1	1	1
ResNet11	0.9863	1	1	1
DenseNet-201	0.9893	1	1	1
ResNet50V1	0.98973	0.96	1	0.97959
ResNet50V2	0.99315	1	1	1
MobileNetV2	0.99658	1	1	1
Majority Voting (hard approach)	0.99315	1	1	1
Majority Voting (soft approach)	0.99315	1	1	1
Weighted Voting using Neural Networks	0.98630	1	1	1
SVM-based voting	0.98973	1	1	1
Random Forests-based voting	0.98630	1	1	1
Mean	0.97945	0.94663	1	0.969104
SD	0.01602	0.10743	0	0.066476
Confidence Level (95.0%)	0.00729	0.048903	0	0.03026
Confidence Interval (95.0%)	0.97215–0.98674	0.89773–0.99553	1–1	0.93884–0.99936

In Table 3, all the deep learning models had made an accuracy superior to 0.95, and 13 out of them had made an accuracy superior to 0.99. Although these good results, we cannot judge the models' performance until we see the accuracy on the test set. In fact, good accuracy in the training set, coupled with lower performance on the test set, reflects that the model suffers from overfitting, making it inefficient to use in real cases.

Table 5 presents the Inference time of the algorithms, which is exactly the time we need to detect COVID-19 in chest X-ray images. The average inference time is 1 ms.

Table 5. Inference time of the algorithms on the test dataset.

ALGORITHM USED	Inference Time (ms)
EfficientNet-B0	2
EfficientNet-B3	2
EfficientNet-B4	1
VGG16	0.879
EfficientNet-B2	2
EfficientNet-B5	2
EfficientNet-B6	2
EfficientNet-B7	2
Xception	2
InceptionV3	1
EfficientNet-B1	2
ResNet11	0.996
DenseNet-201	2
ResNet50V1	1
ResNet50V2	1
MobileNetV2	2

As it can be observed in Table 4, the general performance of the models has decreased. This reflects a weak degree of overfitting in most classifiers. Based on the overall accuracy, we can see that some classifiers were more prone to overfitting (like EfficientNet-B7), whereas others had no overfitting (like MobileNetV2 and ResNetV2).

In Figure 11, we tried to emphasize more the overall accuracy of the different deep learning models on the test set. We can see that the best classifier is MobileNetV2, which and accuracy of 0.99658.

In Figure 12, we show the plots of accuracy and loss of the top 5 best classifiers (MobileNetV2, ResNet50V1, DenseNet-201, ResNet11 and ResNet50V2). The plots are drawn for the training and the validation sets of the Chest X-ray Datasets.

These figures demonstrate that all the top models converge efficiently on the training set from the few first epochs. In fact, 100 epochs were sufficient for all the models to converge. However, in the validation set, more epochs were needed to reach the convergence stage, especially for MobileNetV2, ResNet50V1, and ResNet11. Moreover, the overfitting degree was weak in all models (convergence of the Accuracy on the training set is close to its convergence on the validation set). The least prone model to overfitting was ResNet11.

Concerning the other metrics, the Precision, the Recall, and the F1 score measured for the class COVID, quasi-optimal were obtained. Beginning by the Precision metric, almost the top classifiers gave a precision of 1 (except ResNet50V1, which gave 0.96). This is illustrated in Figure 13. Hence, this means clinically, that, based on the datasets, the top models have a strong ability to not classify non-COVID cases as COVID cases. This prevents the Medical staff from giving expensive care to individuals that did not suffer from COVID in reality.

Concerning the Recall metric, it is more the most important metric to consider for the COVID diagnosis. The higher this metric is, the higher the model ability to detect all the COVID cases in the dataset. All the 16 deep learning classifiers were successful in getting the full recall score: 1. The clinical impact of this fact is that the risk of labeling COVID cases as safe is almost zero in real cases. Almost every COVID patient will be detected successfully by them and, therefore, will be assigned the right care from the medical staff. These patients will be at a lower risk of death and at a lower risk of infecting others with this disease.

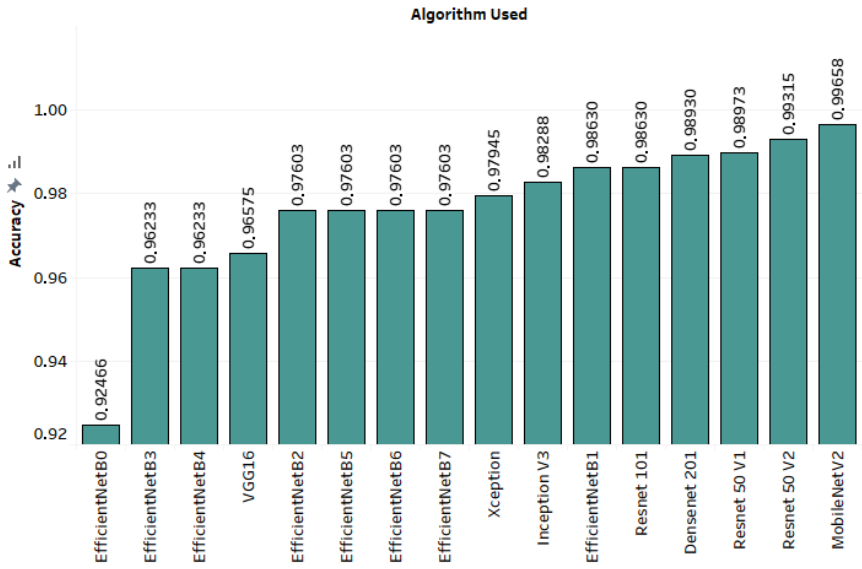

Figure 11. Overall Accuracy of the deep learning models on the test set.

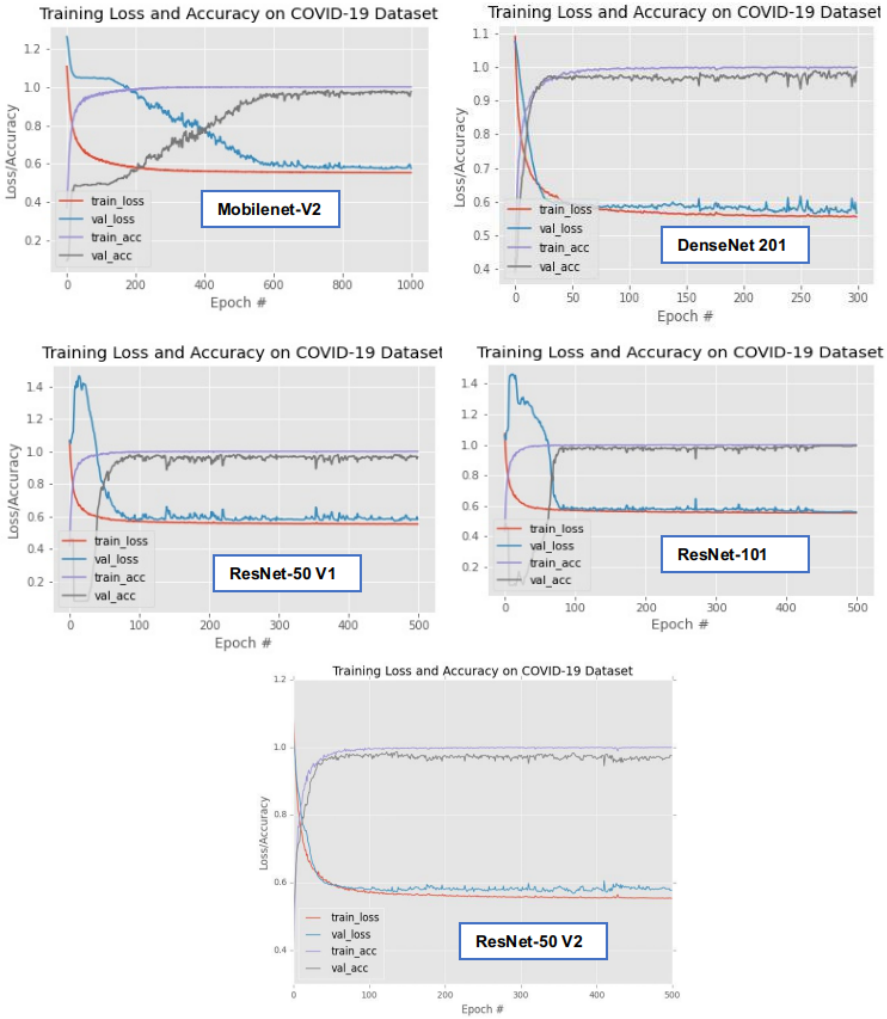

Figure 12. Training Loss and Accuracy for MobileNetV2, DenseNet-201, ResNet50V1, ResNet11, and ResNet50V2.

Concerning the F1 score, it is a metric that combines precision and recall and gives an idea about the model's strength regarding the class COVID. As the recall is 1 for models, this metric will be 1 when the precision metric is 1. For other models, the F1 score will decrease following the error rate in the precision metric. According to Figure 14, all the top selected classifiers got an F1 score of 1, except the ResNet50V1 which got 0.9796.

Finally, we provided in Figure 15 the confusion matrices of the best five models. We can see that both models achieve the perfect performance of 100% for the COVID-19 class. We can see that the performance of all the models for the class COVID (Class of index 1) is 100% in all the models, except the ResNet50V1. If we limit the confusion matrices on only two classes ("COVID"/"non-COVID" cases), we will get an overall accuracy of 100% in four of the top selected models.

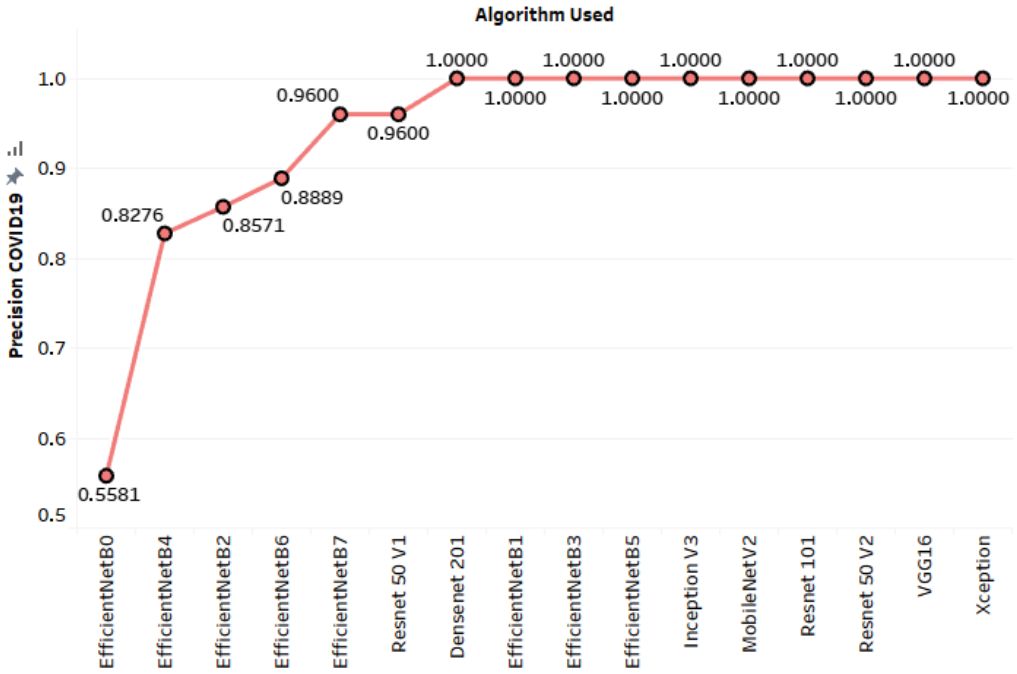

Figure 13. Precision.

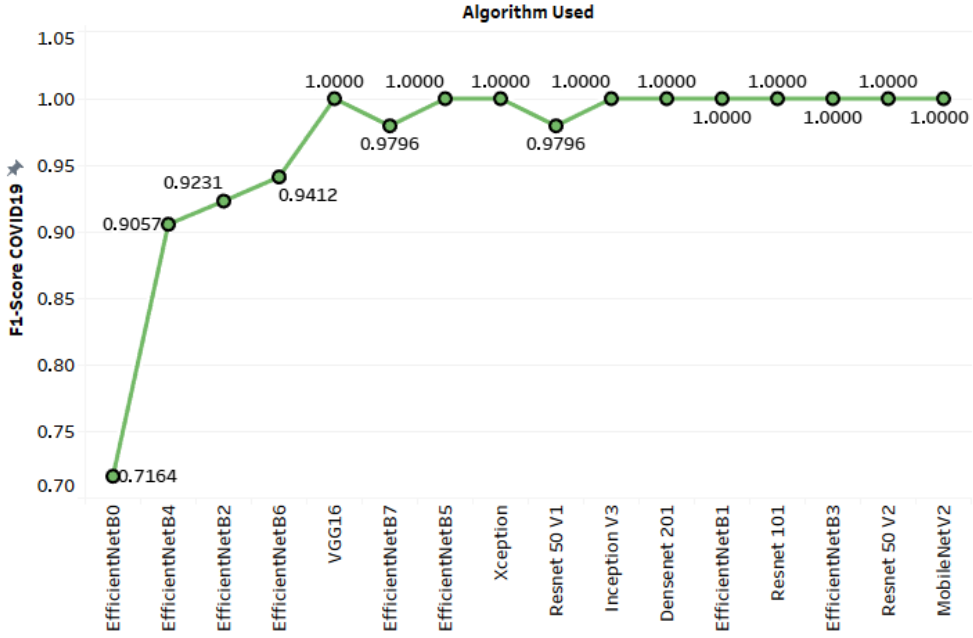

Figure 14. F1 score.

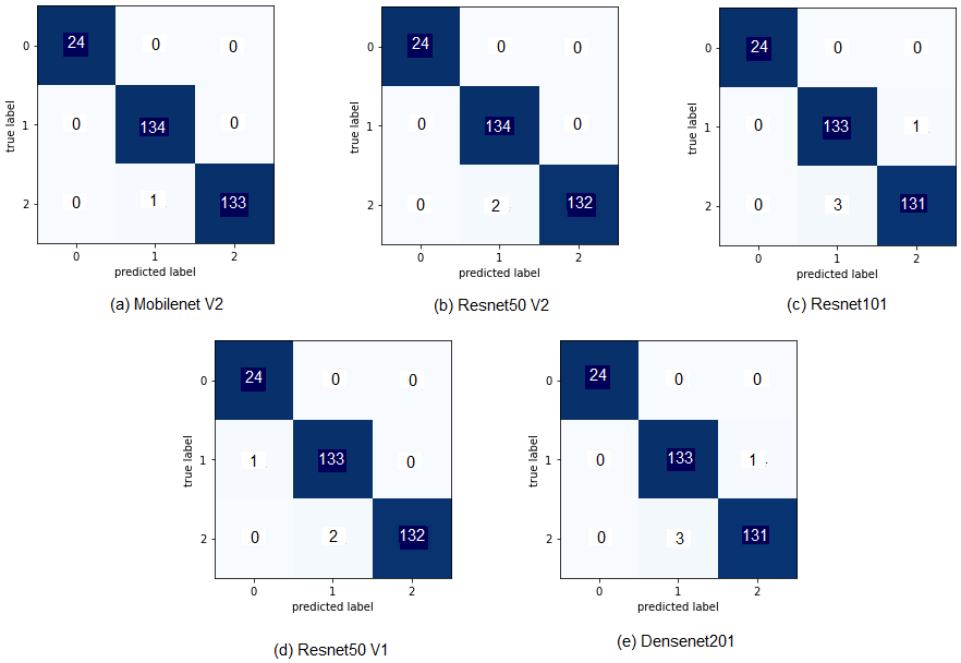

Figure 15. Confusion matrix of the best five models on the chest X-ray test dataset 0: 'COVID-19', 1: 'NORMAL', 2: 'Viral Pneumonia'.

3.4. Ensemble Voting Appraoches

Based on the overall accuracy on the dataset, we selected the best five classifiers, which are: MobileNetV2, ResNet50V2, ResNet50V1, DenseNet-201, and ResNet11. In order to improve more the accuracy, we decided to apply 5 of Ensemble voting approaches, which are already described in Section 3.

Concerning the first two ensemble voting approaches: Majority Voting using the hard approach and the Majority Voting using the soft approach, they do not need training. They are applied statically without any preliminary training. However, for the other three: The Weighted Voting using Neural Networks, the SVM-based voting, and the Random Forests-based voting, we trained the voting model on the validation dataset before applying it on the test set. The weighting approach is learned independently on the validation dataset before applying it to the test dataset.

We can see in the table that, through the Ensemble voting approaches, the Majority Voting (both hard and soft approaches) are the best compared to other voting approaches. They gave us slightly lesser accuracy than the top best classifier (0.00343 compared to the best classifier: MobileNetV2).

Concerning the other voting approaches (Weighted voting using Neural Networks, SVM-based voting, and Random Forests-based voting), they are less efficient than the Majority Voting Approaches. This means that there is not an optimal weighted combination of the classifiers labeling that works in all cases. To emphasize this fact, we compared the accuracy on the validation set (where the models are trained) to their accuracies tested in the test set; we also compared the mean, the standard deviation, the Confidence Level (95.0%), and the Confidence Interval (95.0%) of the accuracy on the validation set with those of the accuracy tested on the test set. All the results are presented in Table 6.

Table 6. Evaluation metrics of 3 ensemble classification algorithms (Neural Networks, SVM and Random Forests on the test and the validation dataset.

ALGORITHM USED	Accuracy in the Validation Set	Accuracy in the Test Set
Weighted Voting using Neural Networks	0.99658	0.98630
SVM-based voting	0.99658	0.98973
Random Forests-based voting	0.99315	0.98630
Mean	0.99543	0.98744
SD	0.00198	0.00198
Confidence Level (95.0%)	0.00491	0.00491
Confidence Interval (95.0%)	0.99051–1	0.98252–0.99236

We can see in Table 6, although the three models were successful in learning, a good representation of weighted combination between the different classifiers. This representation is not optimal in all cases and changes from one dataset to another. In fact, the high accuracies of the top selected classifiers (above 98%) make the mission for these models more difficult. Hence, we avoided using these approaches in the rest of this study and kept focusing on the Majority Approaches.

Returning to Table 4, the Majority Voting (both hard approach and soft approach) gave us the same metrics. This means that all the selected deep learning classifiers have strong discriminative capabilities. They detect the final predicted class with a very strong probability compared to other classes. In fact, the soft approach gives us better results because it works on the probability and not the final associated label. To emphasize this fact, we tested the Majority Voting approaches on three classifiers that are not among the best (EfficientNet-B7, EfficientNet-B6, and EfficientNet-B5), and we calculated the mean, the standard deviation, the Confidence Level (95.0%) and the Confidence Interval (95.0%) of these different models for every considered metric. We got the results presented in Table 7.

Table 7. Comparison between the soft and the hard approaches for Majority voting implemented on three classifiers (EfficientNet-B5, B6 and B7).

ALGORITHM USED	Accuracy	Precision	Recall	F1-Score
EfficientNet-B5	0.97603	1	1	1
EfficientNet-B6	0.97603	0.88889	1	0.94118
EfficientNet-B7	0.97603	0.96	1	0.97959
Majority Voting (hard approach)	0.98630	0.96	1	0.97959
Majority Voting (soft approach)	0.98973	1	1	1
Mean	0.98082	0.96177	1	0.98007
SD	0.00667	0.04538	0	0.02401
Confidence Level (95.0%)	0.00828	0.05635	0	0.02982
Confidence Interval (95.0%)	0.97253–0.98911	0.90541–1.00000	1–1	0.95025–1

As seen in Table 7, the Accuracy of the soft approach is better than the hard approach. And both approaches outperform clearly the three selected classifiers. In fact, the classifiers have lesser discriminative capability than the top best. Hence, we can affirm that the Majority Voting approaches work better when the classifier's performance is less than the optimal. Moreover, when we have the less discriminative capability; the soft approach works better than the hard approach, in general.

Going further, we analyzed the measures of the ROC (Receiver Operating Characteristic) AUC (Area Under Curve). Since our problem is a multi-classification problem, we followed two strategies to convert it into a binary classification problem. The first is the OVO (One versus One) approach, in which we divided the dataset into multiple sub-

datasets. In each one of them, we only consider only one class versus another. The second approach used is the OVR (One versus the Rest). In OVR, we split the dataset into multiple sub-datasets where we consider only one class versus all the rest. Then, for every one of the two approaches, we averaged the different obtained scores using two methods: macro average score and prevalence weighted average. The scores are calculated for the top selected classifiers (DenseNet-201, ResNet50V1, ResNet50V2, MobileNetV2, and ResNet11) and the Soft Majority Approach. We cannot measure the AUC score for the Hard Majority Approach because the predicted label in this case is not obtained from a probability. The Results are presented in Table 8.

Table 8. AUC scores for DenseNet-201, ResNet50V1, ResNetV2, MobileNetV2, ResNet11, and the Soft Majority Voting among them.

	AUC Scores Following the OVO Scheme		**AUC Scores Following the OVR Scheme**	
	Macro Average	**Prevalence-Weighted Average**	**Macro Average**	**Prevalence-Weighted Average**
DenseNet-201	0.995056	0.993194	0.995056	0.993194
ResNet50V1	0.999103	0.998765	0.999103	0.998765
ResNet50V2	0.997229	0.996185	0.997229	0.996185
MobileNetV2	0.999843	0.999783	0.999843	0.999783
ResNet11	0.992285	0.989379	0.992285	0.989379
Soft Majority Voting	0.999827	0.999762	0.999827	0.999762

We can see there that the Soft Majority Voting and the MobileNetV2 have the best AUC scores than all the others. The margin between these two methods is not statistically significant compared to their margin with the others classifiers. In fact, this margin is only 0.2% of the total margin between the best and the least performing algorithms. Hence, this confirms the validity of choosing the Majority Voting approach in general cases to avoid performance variability of the classifiers among different test sets. In fact, the performance of MobileNetV2 was remarkably lower than the Soft Majority Approach on another test set (see Table 11), while the Soft Majority Approach was always the best performing algorithm on it.

3.5. Discussion

All the steps performed during this study have a remarkable impact on the efficiency of the classifiers. We started with the chest X-ray image pre-processing and the data augmentation. For image pre-processing, we proposed to remove all duplicate images from the original dataset (77 duplicate images were removed). Although the number of images has decreased in this case, it improves the dataset's performance because having duplicate images in the dataset creates a problem for two reasons. First, it introduces bias into your dataset, giving the deep neural network additional opportunities to learn patterns specific to the duplicates. Second, it hurts your model's ability to generalize to new images outside of what it was trained on. For the data augmentation, All of the pre-trained models were large enough to be overfitted easily on this dataset. To avoid this, we resized the images to $224 \times 224 \times 3$, and we included the random rotation of the images, as data augmentation has an effect to prevent overfitting.

Then, we selected the most powerful deep learning algorithms for the images classification to study in our task: MobileNetV2, Xception, InceptionV3, DenseNet-201, VGG16, ResNet50 (V1 and V2), ResNet11, and EfficientNet (B0, B1, B2, B3, B4, B5, B6, and B7), We fine-tuned these models by adding the proposed head model composed by one average pooling layer with a size of 4×4, two ReLU activation layers, two batch normalization layers, dropout 0.5, and, finally, a Softmax layer.

All proposed models demonstrated attractive results, with an accuracy rate of around 98%. Moreover, all the methods have 100% recall on the test set. A higher recall value means a lower number of False-Negative (FN) cases, which is very important in the diagnosis of COVID-19 cases. A patient who has a negative result is actually infected and will have a

normal life without taking any precautionary measures, which can cause the infection of other persons, which is very dangerous.

We compared our results with state of the art tested on the original chest X-ray test dataset. Wang et al. [16] used COVID-Net network architecture, which has a lightweight residual (PEPX) design pattern. They obtained a sensitivity of 88.6% and a precision of 91.33%, with an accuracy of 92.4%. Farooq et al. [29] used an implementation of the ResNet50 model, pretrained on the ImageNet dataset. They obtained a sensitivity of 96.9% and a precision of 96.8%, with an accuracy of 96.32%. The comparison is given in Table 9, and we can see that our best models have outperformed other state of the art methods.

Table 9. Comparative results for each model on test Accuracy.

Algorithm	Accuracy
3D CNN (ResNet18) [11]	86.7%
VBNet [12]	91.6%
COVID-Net [16]	93.3%
ResNet50 [29]	96.23%
4 CNN models [33]	89.5%
ResNet50V2	99.315%
MobileNetV2	99.658%
Majority Voting	99.315%

As demonstrated previously, the top selected classifiers gave us quasi-optimal results when tested on the chest X-ray test dataset. All of them gave 100% Accuracy in treating COVID cases, except the ResNet50V1. However, to be able to generalize, we need more experiments. This is why we tested all the classifiers and the Majority Voting again in another dataset, which is the Validation dataset. The results are provided in Table 10.

Table 10. Evaluation metrics of the algorithms on the validation dataset.

ALGORITHM USED	Accuracy	Precision	Recall	F1-Score
EfficientNet-B0	0.88316	0.44681	0.91304	0.6
EfficientNet-B6	0.93471	0.6875	0.95652	0.8
EfficientNet-B4	0.94502	0.7931	1	0.88462
EfficientNet-B3	0.95189	0.82143	1	0.90196
EfficientNet-B2	0.95189	0.74194	1	0.85185
EfficientNet-B1	0.95876	0.88462	1	0.93878
InceptionV3	0.96564	1	1	1
ResNet50V1	0.96564	0.88	0.95652	0.91667
EfficientNet-B5	0.96564	1	1	1
VGG16	0.96907	0.95833	1	0.97872
ResNet50V2	0.97595	95652	0.95652	0.95652
MobileNetV2	0.97595	0.92	1	0.95833
Xception	0.97945	1	0.95652	97778
DenseNet-201	0.98625	1	1	1
ResNet11	0.99313	1	1	1
EfficientNet-B7	0.99313	1	1	1
Majority Voting (hard approach)	0.99313	1	1	1
Majority Voting (soft approach)	0.99313	1	1	1

From Table 10, we can deduce that the top 5 classifiers selected previously are among the best on the validation dataset. Although the top-performing algorithm is different from

test to validation datasets. The best classifier in the test set was MobileNetV2, with an accuracy of 0.99658. But it gave lesser performance when tested on the validation dataset: 0.97595. However, the Majority Voting approach gave in the validation set the best accuracy: 0.99313. Going deeper, we studied the average overall accuracy of every algorithm on both sets (the test set and the validation set). The results are provided in Table 11. The results are ordered by descending order following the average accuracy calculated by averaging every algorithm's accuracy on the test and the validation sets.

Based on the results provided in Table 11, we note that the best method to use for the problem treated in this paper is the Majority Voting method based on combinating results from 5 classifiers: MobileNetV2, ResNet50V2, ResNet50V1, DenseNet-201, and ResNet11. In fact, this method gave us the best average accuracy on both test and validation sets. Moreover, it gives us 100% accuracy regarding the class COVID on both validation and test sets (100% precision, 100% recall, and 100% F1 score on both sets). The average accuracy of the Majority Voting approach is remarkably better than any other classifier tested on the Chest X-ray Dataset (even the top selected classifiers in the test set).

Table 11. Average Accuracy of the classifiers and the Majority Voting methods on both the test and the validation dataset.

Model	Accuracy on the Test Set	Accuracy on the Validation	Average Accuracy
Majority Voting (hard approach)	0.99315	0.99313	0.99314
Majority Voting (soft approach)	0.99315	0.99313	0.99314
ResNet11	0.9863	0.99313	0.989715
DenseNet-201	0.9893	0.98625	0.987775
MobileNetV2	0.99658	0.97595	0.986265
EfficientNet-B7	0.97603	0.99313	0.98458
ResNet50V2	0.99315	0.97595	0.98455
Xception	0.97945	0.97945	0.97945
ResNet50V1	0.98973	0.96564	0.977685
InceptionV3	0.98288	0.96564	0.97426
EfficientNet-B1	0.9863	0.95876	0.97253
EfficientNet-B5	0.97603	0.96564	0.970835
VGG16	0.96575	0.96907	0.96741
EfficientNet-B2	0.97603	0.95189	0.96396
EfficientNet-B3	0.96233	0.95189	0.95711
EfficientNet-B6	0.97603	0.93471	0.95537
EfficientNet-B4	0.96233	0.94502	0.953675
EfficientNet-B0	0.92466	0.88316	0.90391

Going deeper, we decided to test the statistical significance of our introduced approach. In other words, we need to statistically reject the hypothesis that assumes that the Majority Voting superiority came only by chance in our experiments. So, we decided first to define explicitly the Null Hypothesis H_0 that we want to reject by calculating the p-value. As we need to statistically quantify the superiority of one algorithm over the other, we considered H_0 as the hypothesis that the classification Method does not belong to the top 5% margin of the difference between the top and the least performing algorithm on a selected dataset. The Alternate Hypothesis H_1 will be then defined as the hypothesis that the classification method belongs to the top 5% margin of the difference between the top and the least performing algorithms on a selected dataset. To calculate the p-value, we converted Table 11 into a more understandable way to estimate the superiority of one algorithm over the others. On every selected dataset, we calculated the increase in accuracy of every method proportionally to the margin between the top and the least algorithm. Results are written in Table 12.

The probability that one method belongs to the top 5% of the margin between the best and the least performing algorithm on one dataset is: $p_0 = 0.05$. Based on the

observations made on the test and the validation sets, the probability that the Majority Voting belongs to the top 5% margin on both the test and the validation set is p-value $= P(H_0) = p_0 \times p_0 = 0.0025$. The common α value used as the threshold for p-value is 0.05 [53]. In our case, p-value $= 0.0025 << 0.05$, which is strongly sufficient to reject the Null Hypothesis H_0. This strongly disprove the hypothesis that the Majority Voting superiority among other classifiers is explained by chance. Moreover, calculating the p-value for other methods is not enough to reject the Null Hypothesis H_0 for them. This reinforces more the statistical significance of the superiority of the Majority Voting over all the other classifiers.

Table 12. Accuracy improvement of the selected methods proportionally to the margin between the best and the least recorded accuracy on the test and the validation sets.

Model	Test Set	Validation Set
Majority Voting (hard approach)	95%	100%
Majority Voting (soft approach)	95%	100%
ResNet11	86%	100%
DenseNet-201	90%	94%
MobileNetV2	100%	84%
EfficientNet-B7	71%	100%
ResNet50V2	95%	84%
Xception	76%	88%
ResNet50V1	90%	75%
InceptionV3	81%	75%
EfficientNet-B1	86%	69%
EfficientNet-B5	71%	75%
VGG16	57%	78%
EfficientNet-B2	71%	62%
EfficientNet-B3	52%	62%
EfficientNet-B6	71%	47%
EfficientNet-B4	52%	56%
EfficientNet-B0	0%	0%

Hence, we suggest that the Majority Voting approach is the most efficient method to use in general cases when we do not have ideas about the targeted dataset. For all other classifiers, the accuracy changes from one dataset to another. No classifier is able to be adopted for every dataset. This study emphasizes the efficiency of the Voting approaches (especially the Majority Voting approach) in treating some sensitive tasks, like COVID diagnosis.

Our study has some limitations to be targeted in the next research works. One of them is that the dataset is not associated with data about the subjects who participated in the study. This fact obliged us to be limited to descriptive statistics and prevented us from using inferential statistics tools. Data, like gender, age, and clinical symptoms, could strengthen our medical analysis method for better adoption in real cases. The dataset was also not associated with the PCR test result (Polymerase chain reaction test) for every sample image. PCR test is considered by many as the gold standard for COVID-19 diagnosis. Calculation of the agreement rate between our method and the PCR test allows us to judge more the potential of our method for a prevalent and widespread adoption in the actual condition of the COVID-19 pandemic. All these limitations can be targeted in other studies to go deeper into our method's clinical interpretation.

4. Conclusions

In this study, we targeted the COVID diagnosis task from the chest X-ray images. We began by preparing the dataset to be used. We selected the deep learning models to best tested among the current state of the art algorithms in image classification. We modified their architecture to add our designed head model. We performed data augmentation and made the training of all the selected classifiers on the processed dataset. We found

very encouraging results when testing on the test set. All the classifiers got an accuracy of around 98%. The recall was 100% for all of them, which has an important clinical advantage. This means the labeling of COVID cases to other classes is almost zero, which reduces the risk of non-detecting COVID cases from their chest X-rays. To go beyond in improving the accuracy, we selected the top-performing classifiers on the test set and designed five different Ensemble Voting methods. To reinforce our findings, we made the experiments on two different sets (the test set and the validation set). We found that the best approach to be adopted for COVID diagnosis is the Majority Voting method based on the results given by the top selected classifiers on the test set: MobileNetV2, ResNet50V2, ResNet50V1, DenseNet-201, and ResNet11. The Majority Voting gave us an average accuracy of 0.99314 with 100% accuracy regarding the COVID class when tested on the test and the validation set. To avoid the classifiers' performance change from one test set to another, we propose the Majority Voting as the best strategy to follow in general cases. This study emphasizes more on the utility of the Majority Voting in treating sensitive and important tasks, like COVID-19 diagnosis.

In future work, we need to invest more in the voting approaches by studying its performance on larger datasets. Moreover, we need to go deeper in studying the soft approach as it gives better results than the hard approach in many cases. Finally, we need to overcome the cases where true labels are voted by a minority to tune the Majority Voting to better performance.

Author Contributions: A.K., A.A., and B.B. designed the method. M.B.J. and B.B. implemented the method and wrote the paper. A.K., A.A., and H.H. contributed to the supervision of the work, analysis of the method, and paper writing. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Robotics and Internet of Things Lab of Prince Sultan University.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Acknowledgments: We want to acknowledge The Robotics and Internet of Things Lab of Prince Sultan University for funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
2. Koubâa, A.; Ammar, A.; Benjdira, B.; Al-Hadid, A.; Kawaf, B.; Al-Yahri, S.A.; Babiker, A.; Assaf, K.; Ras, M.B. Activity Monitoring of Islamic Prayer (Salat) Postures using Deep Learning. In Proceedings of the 2020 6th Conference on Data Science and Machine Learning Applications (CDMA), Riyadh, Saudi Arabia, 4–5 March 2020; pp. 106–111.
3. Benjdira, B.; Ouni, K.; Al Rahhal, M.M.; Albakr, A.; Al-Habib, A.; Mahrous, E. Spinal Cord Segmentation in Ultrasound Medical Imagery. *Appl. Sci.* **2020**, *10*, 1370. [CrossRef]
4. Benjdira, B.; Ammar, A.; Koubaa, A.; Ouni, K. Data-Efficient Domain Adaptation for Semantic Segmentation of Aerial Imagery Using Generative Adversarial Networks. *Appl. Sci.* **2020**, *10*, 1092. [CrossRef]
5. Benjdira, B.; Bazi, Y.; Koubaa, A.; Ouni, K. Unsupervised Domain Adaptation Using Generative Adversarial Networks for Semantic Segmentation of Aerial Images. *Remote Sens.* **2019**, *11*, 1369. [CrossRef]
6. Alhichri, H.; Bazi, Y.; Alajlan, N.; Bin Jdira, B. Helping the Visually Impaired See via Image Multi-labeling Based on SqueezeNet CNN. *Appl. Sci.* **2019**, *9*, 4656. [CrossRef]
7. Alhichri, H.; Jdira, B.B.; Alajlan, N. Multiple Object Scene Description for the Visually Impaired Using Pre-trained Convolutional Neural Networks. In *International Conference on Image Analysis and Recognition*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 290–295.
8. Koubaa, A. Understanding the covid19 outbreak: A comparative data analytics and study. *arXiv* **2020**, arXiv:2003.14150.
9. Corman, V.M.; Landt, O.; Kaiser, M.; Molenkamp, R.; Meijer, A.; Chu, D.K.; Bleicker, T.; Brünink, S.; Schneider, J.; Schmidt, M.L.; et al. Detection of 2019 novel coronavirus (2019-nCoV) by real-time RT-PCR. *Eurosurveillance* **2020**, *25*, 2000045. [CrossRef] [PubMed]
10. Huang, C.; Wang, Y.; Li, X.; Ren, L.; Zhao, J.; Hu, Y.; Zhang, L.; Fan, G.; Xu, J.; Gu, X.; et al. Clinical features of patients infected with 2019 novel coronavirus in Wuhan, China. *Lancet* **2020**, *395*, 497–506. [CrossRef]

11. Xu, X.; Jiang, X.; Ma, C.; Du, P.; Li, X.; Lv, S.; Yu, L.; Chen, Y.; Su, J.; Lang, G.; et al. Deep learning system to screen coronavirus disease 2019 pneumonia. *arXiv* **2020**, arXiv:2002.09334.
12. Shan, F.; Gao, Y.; Wang, J.; Shi, W.; Shi, N.; Han, M.; Xue, Z.; Shen, D.; Shi, Y. Lung infection quantification of covid-19 in ct images with deep learning. *arXiv* **2020**, arXiv:2003.04655.
13. Xie, X.; Zhong, Z.; Zhao, W.; Zheng, C.; Wang, F.; Liu, J. Chest CT for typical 2019-nCoV pneumonia: Relationship to negative RT-PCR testing. *Radiology* **2020**, *296*. [CrossRef]
14. Ucar, F.; Korkmaz, D. COVIDiagnosis-Net: Deep Bayes-SqueezeNet based Diagnostic of the Coronavirus Disease 2019 (COVID-19) from X-ray Images. *Med. Hypotheses* **2020**, *140*, 109761. [CrossRef] [PubMed]
15. Wang, W.; Xu, Y.; Gao, R.; Lu, R.; Han, K.; Wu, G.; Tan, W. Detection of SARS-CoV-2 in different types of clinical specimens. *Jama* **2020**, *323*, 1843–1844. [CrossRef]
16. Wang, L.; Wong, A. COVID-Net: A tailored deep convolutional neural network design for detection of COVID-19 cases from chest radiography images. *arXiv* **2020**, arXiv:2003.09871.
17. Duran-Lopez, L.; Dominguez-Morales, J.P.; Corral-Jaime, J.; Vicente-Diaz, S.; Linares-Barranco, A. COVID-XNet: A custom deep learning system to diagnose and locate COVID-19 in chest X-ray images. *Appl. Sci.* **2020**, *10*, 5683. [CrossRef]
18. Sahlol, A.T.; Yousri, D.; Ewees, A.A.; Al-Qaness, M.A.; Damasevicius, R.; Abd Elaziz, M. COVID-19 image classification using deep features and fractional-order marine predators algorithm. *Sci. Rep.* **2020**, *10*, 1–15. [CrossRef]
19. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 2818–2826.
20. Faramarzi, A.; Heidarinejad, M.; Mirjalili, S.; Gandomi, A.H. Marine Predators Algorithm: A nature-inspired metaheuristic. *Expert Syst. Appl.* **2020**, *152*, 113377. [CrossRef]
21. Chowdhury, M.E.; Rahman, T.; Khandakar, A.; Mazhar, R.; Kadir, M.A.; Mahbub, Z.B.; Islam, K.R.; Khan, M.S.; Iqbal, A.; Al-Emadi, N.; et al. Can AI help in screening Viral and COVID-19 pneumonia? *arXiv* **2020**, arXiv:2003.13145.
22. Karakanis, S.; Leontidis, G. Lightweight deep learning models for detecting COVID-19 from chest X-ray images. *Comput. Biol. Med.* **2021**, *130*, 104181. [CrossRef] [PubMed]
23. Mirza, M.; Osindero, S. Conditional generative adversarial nets. *arXiv* **2014**, arXiv:1411.1784.
24. Cohen, J.P.; Morrison, P.; Dao, L.; Roth, K.; Duong, T.Q.; Ghassemi, M. Covid-19 image data collection: Prospective predictions are the future. *arXiv* **2020**, arXiv:2006.11988.
25. Chest X-Ray Images (Pneumonia). Available online: https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia (accessed on 2 March 2021).
26. Zebin, T.; Rezvy, S. COVID-19 detection and disease progression visualization: Deep learning on chest X-rays for classification and coarse localization. *Appl. Intell.* **2021**, *51*, 1010–1021. [CrossRef]
27. Zhu, J.Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2223–2232.
28. Tan, M.; Le, Q.V. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv* **2019**, arXiv:1905.11946.
29. Farooq, M.; Hafeez, A. Covid-resnet: A deep learning framework for screening of covid19 from radiographs. *arXiv* **2020**, arXiv:2003.14395.
30. Ng, M.Y.; Lee, E.Y.; Yang, J.; Yang, F.; Li, X.; Wang, H.; Lui, M.M.S.; Lo, C.S.Y.; Leung, B.; Khong, P.L.; et al. Imaging profile of the COVID-19 infection: Radiologic findings and literature review. *Radiol. Cardiothorac. Imaging* **2020**, *2*, e200034. [CrossRef]
31. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
32. Howard, J.; Gugger, S. Fastai: A layered API for deep learning. *Information* **2020**, *11*, 108. [CrossRef]
33. Minaee, S.; Kafieh, R.; Sonka, M.; Yazdani, S.; Soufi, G.J. Deep-covid: Predicting covid-19 from chest x-ray images using deep transfer learning. *arXiv* **2020**, arXiv:2004.09363.
34. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4510–4520.
35. Chollet, F. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1251–1258.
36. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
37. Qassim, H.; Feinzimer, D.; Verma, A. Residual squeeze vgg16. *arXiv* **2017**, arXiv:1705.03004.
38. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.
39. Khan, A.; Sohail, A.; Zahoora, U.; Qureshi, A.S. A survey of the recent architectures of deep convolutional neural networks. *Artif. Intell. Rev.* **2020**, *53*, 5455–5516. [CrossRef]
40. Google. Advanced Guide to Inception v3 on Cloud TPU. Available online https://cloud.google.com/tpu/docs/inception-v3-advanced?hl=en (accessed on 3 March 2021).
41. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**, arXiv:1409.1556.

42. Al-Mukhtar, M. Random forest, support vector machine, and neural networks to modelling suspended sediment in Tigris River-Baghdad. *Environ. Monit. Assess.* **2019**, *191*, 1–12. [CrossRef]
43. Han, T.; Jiang, D.; Zhao, Q.; Wang, L.; Yin, K. Comparison of random forest, artificial neural networks and support vector machine for intelligent diagnosis of rotating machinery. *Trans. Inst. Meas. Control* **2018**, *40*, 2681–2693. [CrossRef]
44. Liu, M.; Wang, M.; Wang, J.; Li, D. Comparison of random forest, support vector machine and back propagation neural network for electronic tongue data classification: Application to the recognition of orange beverage and Chinese vinegar. *Sens. Actuators B Chem.* **2013**, *177*, 970–980. [CrossRef]
45. Raczko, E.; Zagajewski, B. Comparison of support vector machine, random forest and neural network classifiers for tree species classification on airborne hyperspectral APEX images. *Eur. J. Remote Sens.* **2017**, *50*, 144–154. [CrossRef]
46. Nitze, I.; Schulthess, U.; Asche, H. Comparison of machine learning algorithms random forest, artificial neural network and support vector machine to maximum likelihood for supervised crop type classification. In Proceedings of the 4th GEOBIA, Rio de Janeiro, Brazil, 7–9 May 2012; Volume 35.
47. Martinez-Castillo, C.; Astray, G.; Mejuto, J.C.; Simal-Gandara, J. Random forest, artificial neural network, and support vector machine models for honey classification. *eFood* **2019**, *1*, 69–76. [CrossRef]
48. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. TensorFlow: A system for large-scale machine learning. *arXiv* **2016**, arXiv:1605.08695.
49. Chollet, F. Keras. 2015. Available online: https://github.com/fchollet/keras (accessed on 21 March 2021).
50. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Müller, A.; Nothman, J.; Louppe, G.; et al. Scikit-learn: Machine Learning in Python. *arXiv* **2012**, arXiv:1201.0490.
51. Van Rossum, G.; Drake, F.L., Jr. *Python Tutorial*; Centrum voor Wiskunde en Informatica: Amsterdam, The Netherlands, 1995.
52. Jupyter Lab Project. Available online: https://jupyter.org/ (accessed on 4 March 2021).
53. Fisher, R.A. *The Design of Experiments*; Oliver & Boyd: Edinburgh, UK, 1937.