International Journal of Reliable and Quality E-Healthcare
Volume 12 • Issue 1

A New Classification Model Based on Transfer Learning of DCNN and Stacknet for Fast Classification of Pneumonia Through X-Ray Images

Jalal Rabbah, University of Hassan II, Morocco*
https://orcid.org/0000-0001-5615-7638

Mohammed Ridouani, University of Hassan II, Morocco

Larbi Hassouni, University of Hassan II, Morocco
https://orcid.org/0000-0002-6219-4280

ABSTRACT

Coronavirus has spread worldwide, with over 688 million confirmed cases and 6.8 million deaths. The results could be important as containment restrictions begin to be relaxed and we are not immune to new strains. They underscore the need to introduce increasingly effective techniques to deal with such a spread and help identify new infections more quickly, at a reasonable cost and with a minimum error rate. Machine learning models constitute a new approach, used increasingly in this field. In this proposed work, the authors built a classification model named CovStacknet based on StackNet metamodeling methodology combined with the deep convolutional neural network as the basis for feature extraction from x-ray images. Firstly, the proposed model used VGG16 as a transfer learning of deep convolutional neural networks and achieved an accuracy score of 98%. Secondly, the proposed model is extended to evaluate four other deep convolutional neural networks, ResNet-50, Inception-V3, MobileNet-V2 and DenseNet, and ResNet-50, has achieved the best performance.

KEYWORDS

Customer ChurnStacknet, Deep Convolutional Neural Networks, DeepInsight, Halving, Machine Learning

INTRODUCTION

Coronavirus is an infectious disease caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2). Considering the degree of its spread worldwide, World Health Organization Director-General declared the 2019-nCoV outbreak a public health emergency of international concern on 30 January 2020 and a pandemic on 11 March 2020. The World Health Organization leads international coordination to country readiness to research and innovation to limit transmission, provide early care, communicate critical information, and minimize social and economic impacts.

DOI: 10.4018/IJRQEH.326765 *Corresponding Author

The choice of Covid-19 to name the novel coronavirus disease is to guard against using other names that might be inaccurate or stigmatizing. Since the early weeks of the pandemic, thousands of researchers tried to stop the Covid -19 outbreak by developing easy-to-apply diagnostics, accelerating existing vaccine candidates, and preventing infection. Common symptoms include fever, cough, fatigue, shortness of breath, and loss of sense of smell. Complications may include pneumonia and acute respiratory distress syndrome. Until now, Covid -19 has had no immunization or treatment. However, numerous continuous clinical preliminaries are assessing potential medicines. More than 688 million Covid-19 infected cases were confirmed in more than 230 countries until March 2023, including more than 6.8 million deaths, 660 million recovered, and 20 million active cases (World Health Organization, 2023). The standard test for current infection with SARS-CoV-2 uses RNA testing of respiratory secretions collected using a nasopharyngeal swab, though it is possible to test other samples. This test uses real-time reverse transcription polymerase chain response (i.e., RT-PCR), which detects the presence of viral RNA fragments (Wang et al., 2020). The test procedure is manual and is facing many issues, namely the shortage of tests available worldwide, it may fail to identify infected patients without symptoms and is a time-consuming process (Zhou et al., 2020). Because of the primary involvement of the respiratory system, chest computerized tomography (CT) is strongly recommended in suspected Covid-19 cases for both initial evaluation and follow-up.

Chest radiographs (CXR) are of little diagnostic value in the early stages, whereas CT findings may be present before symptom onset. Ground glass opacities (GGO) pattern is the most common finding in Covid-19 infections. They are usually multifocal, bilateral, and peripheral. Still, in the early phase of the disease, the GGO may present as a unifocal lesion, most commonly located in the inferior lobe of the right lung (Chassagnon et al., 2020). There are widespread bilateral GGO with a posterior predominance. Sometimes there are thickened interlobular and intralobular lines in combination with a ground glass pattern. CXR images can assist in the early detection of suspected Covid-19 cases, but the overlap with other infectious and inflammatory lung diseases can lead to misdiagnosis. It is, therefore, essential to take the necessary precautions to avoid the financial and health consequences of such a false screening. The call to artificial intelligence has become critical, more particularly in times of crisis such as the Covid-19 pandemic. Indeed, the increased growth of detected and suspected cases has implied a demand far exceeding the capacity of health institutions, even for the best health systems in the world, such as the United States, Italy or Spain.

BACKGROUND

CXR has emerged as a highly suitable domain for developing deep learning algorithms for automatic interpretation (Chassagnon et al., 2020). Notably, training deep learning algorithms for pneumonia detection has shown superior performance compared to the average performance of four radiologists (Rajpurkar et al., 2017). The application of deep learning in implementing automated radiology reporting models has gained significant attention in recent studies (Monshi et al., 2020). Convolutional neural networks (CNNs) integration has been identified as a standard approach for image analysis in this context. Regarding Covid-19 pneumonia imaging, Lomoro et al. (2020) investigated the imaging features of emerging Covid-19 pneumonia using chest ultrasound, Chest X-ray and Computed Tomography. They found that the spectrum of chest imaging manifestations of Covid-19 includes consolidations and hazy increased opacities on CXR, as well as multifocal GGO with consolidations on CT. Twenty-six original studies supported this conclusion. Simonyan and Zisserman (2014) categorized Covid-19 CT findings into various stages, highlighting that only 9% of the patients in the intermediate stage (3-5 days since symptom onset) exhibited negative chest CT. Kedia and Katarya (2021) developed a deep learning model, COVNet, for Covid-19 detection using CNN to pursue improved diagnostic accuracy for CT. COVNet, based on the ResNet50 backbone, demonstrated excellent performance on the test set, achieving an area under the curve (AUC) of 96%,

a sensitivity of 90%, and a specificity of 96%. The model utilized a series of CT slices as input, with resulting feature maps fed into a fully connected layer to generate probability scores for each class.

In this study, the authors aimed to develop a new prediction model called CovStackNet, which combines deep CNN (DCNN) and ensemble learning. This approach leverages the performance and flexibility offered by stacked generalization, which has proven effective across various use cases. The proposed model aims to detect pneumonia cases from chest X-ray images with a high level of certainty and differentiate Covid-19 infections from other forms of pneumonia. The study consisted of two main parts: The first part involved using VGG16 pretrained DCNN for feature extraction. In contrast, the second part comprised a comparative study of the impact of transfer learning algorithms on the model's performance. The authors tested various DCNN architectures in this comparative analysis, including ResNet-50, Inception-V3, MobileNet-V2, and DenseNet. The structure of this paper is as follows: The second section provides a concise problem description, followed by an outline of the analytical methods employed in the third section. The fourth section presents the authors' methodology, while the fifth section showcases the experimental results. The sixth section summarises the comparison results of transfer learning algorithms. Finally, the seventh section offers concluding remarks.

PROBLEM DESCRIPTION

Medical imaging has improved the diagnosis and treatment of numerous medical conditions. CXR is among the most requested exams; it uses small amounts of radiation to produce two-dimensional pictures of the body's organs, tissues, and bones, and it can help spot abnormalities or diseases of the airways, blood vessels, bones, heart, and lungs. The current best practice advises that chest CT is not used to diagnose Covid-19, but many studies show that it may help to diagnose, or at least triage, Covid-19 patients. CXR of infected cases demonstrates characteristic pneumonia like patterns that can help diagnose. CXR remains on the frontline imaging modality of choice for anyone suspected of Covid-19 infection because it is cheap, readily available, and easily cleaned. It is also known worldwide and may be considered the only solution accessible in several regions that do not have the means to perform many PCR tests. However, a significant drawback of using X-rays to test for Covid-19 without the dedicated test kits is that X-ray examination requires a radiology master and takes a considerable time, which is valuable when hundreds of people are sick. Therefore, developing an automated analysis system is essential to save medical professionals valuable time. Machine learning techniques with a big data approach are suitable for building efficient and robust models to extract sound knowledge from medical imaging data. This study used pre-trained DCNNs (Simonyan & Zisserman, 2014) combined with a StackNet (Wolpert, 1992) to build a machine-learning classification model and classify the CXR images.

ENSEMBLE LEARNING USING STACKNETS

In machine learning, the authors refer to ensemble learning as the construction of a metamodel grouping together with several machine learning algorithms to obtain better performance than that of single algorithm models. In other words, the resulting "ensemble" also represents a supervised learning algorithm since it can be trained and used afterwards to generate predictions. From a statistical point of view, the hypothesis represented by the "ensemble" is not necessarily contained in the species of the beliefs constituting it. In practice, "ensemble" models generate better results with a significant diversity of base algorithms.

There are several methods of "ensemble" learning. Bagging (bootstrap aggregation) (Breiman, 1996) uses weak learners in parallel based on a given number of bootstrap sets and combines their results deterministically. Boosting (Freund et al., 1999) performs sequential learning of many weak learners and applies weight to training tuples to help the next classifier perform better. Wolpert (1992) defined Stacking as an ensemble learning method that combines a set of models built by different

algorithms. The first step of Stacking is to train diverse weak learners on the same dataset, called the training set. The resulting vectors are stacked to a new dataset representing all first-level predictions. Once the predictions dataset is filled, a metamodel is trained to predict the target dataset.

StackNets are a kind of generalization on neural networks (Breiman, 1996; Wolpert, 1992) to improve accuracy in machine learning. The activation function can be any supervised machine learning algorithm (i.e., classifiers, regressors or ensemble metamodel). Stacknets aim to improve the accuracy of classifiers or reduce training errors; in this study, the authors used a software implementation of Stacknets in Java.

Since it is not always possible to use backpropagation to train Stacknets, as is the case for neural networks, a forward training methodology is essential. In this study, the authors used the K-fold training cross-validation to train their Stacknet.

The following example refers to a sample of *n* independent observations, *Xi* the input vectors, and *Yi* the target vectors ($i = 1, 2 \ldots, n$). It is possible to predict the targets using different types of learners, such as logistic regression, decision trees, support vector machines, and k-nearest neighbors. However, finding the optimal learner isn't easy; in some cases, learners overfit the training set data. The example includes the following variables:

- The classification model set, which is defined as follows:

$$M_{L,i}: \{ (i=1,2, \ldots, K_L), K_L \text{ number of learners of layer L } \} \quad (1)$$

- The vector of learners in layer *L*, which is equal to the following:

$$V_L: \{(M_{L,1}, M_{L,2}, \ldots, M_{L,KL),} \text{ L (L=0,1, …,N-1), N the number of layers} \quad (2)$$

- The output $O_{L+1,i}$ of the learner i in the layer L+1, could be expressed as follows:

$$\forall\, 0 \le L \le N\text{-}1, \forall\, 1 \le i \le K_L+1, O_{L+1,i} = M_{L+1,i} (O_{1,L}, O_{2,L}, O_{3,L}, \ldots, O_{P,L}) \quad (3)$$

where $P = K_L$. $M_{L+1,i}$ are the metamodels of layer L+1 to combine all previous models' predictions (Wolpert, 1992).

StackNet models are known to be very accurate and robust; they have won many competitions and got top ten results on Kaggle, for example.

DEEP CONVOLUTIONAL NEURAL NETWORK FOR FEATURES EXTRACTION

DCNNs have revolutionized the field of image classification and recognition. They can learn basic shapes in the first and more complicated forms in deeper layers. This approach was inspired in 1962 by stimuli of the visual cortex (Hubel & Wiesel, 1962); this work constituted a breakthrough in computer vision. The proposed model used the pre-trained VGG16 (Figure 1) DCNN to extract and reveal x-ray features hidden in the original images. For this purpose, the authors' features extractor used only the convolution layers to extract a 25,088-length features vector for each input image. These features can identify different levels of image representation. The authors' model does not use the fully connected VGG16 classification layers.

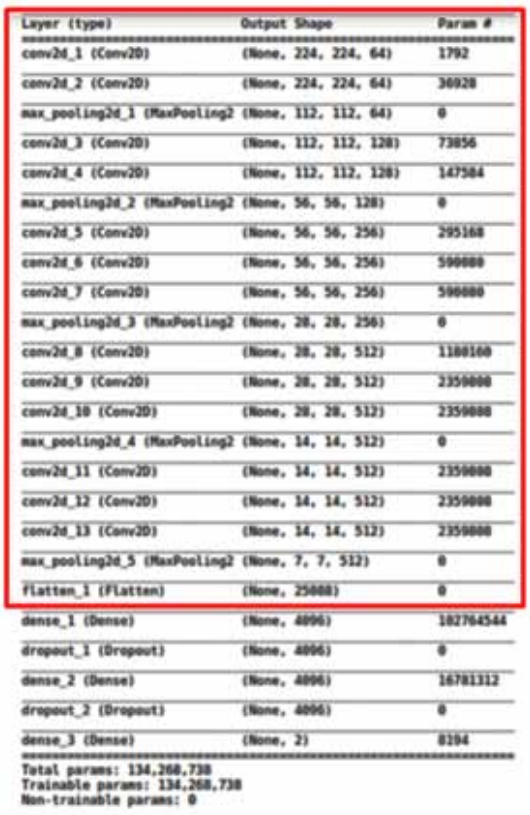
The authors' model consists of 20 pre-trained layers. The first layer is the input layer, taking a (Red, Green, Blue) fixed size of 224 x 224 x 3 pixels image. The following 16 layers combine Convolution + ReLU and Max Pooling layers. These layers are part of Simonyan and Zisserman's (2014) pretrained VGG16 model and are trained on the ImageNet dataset (Deng et al., 2009). ImageNet

contains around 15 million annotated images from 22,000 categories, and VGG16 achieved 92.7% accuracy on ImageNet. Therefore, the authors used the VGG16 model as a base model, as depicted in Figure 1 for feature extraction. Then, they applied a transfer learning model using a Stacknet architecture trained on two X-ray image datasets (Bashir, 2020; Mooney, 2018). The authors built their model based on the TensorFlow library (Abadi, Agarwal et al., 2016) backend to perform tensor operations. Tensor Flow is Google's new artificial intelligence learning system generation based on Disbelief (Abadi, Barham et al., 2016); it analyzes complex data structures into artificial neural networks. In addition, the authors used Keras API (Chollet et al., 2015), a high-level neural network API written in Python, and they used JupyterLab (2022) as the editor.

INPUT DATA: X-RAY IMAGE DATASETS

The first dataset (DS1) (Mooney, 2018) consists of 5,216 X-ray images, including 4,273 having pneumonia (bacterial and viral pneumonia cases) and 1,583 standard cases. In this study, the authors chose an approach that allowed building efficient models even with limited data. The second dataset (DS2) (Bashir, 2020) is a Kaggle open database. The third dataset (DS3) (Miller,2020) is a recently open-sourced database containing CXR pictures of patients suffering from several pneumonia infections, including the Covid-19 disease. The authors' objective was to build a classifier to predict from a CXR scan whether a patient has the virus (Figure 2).

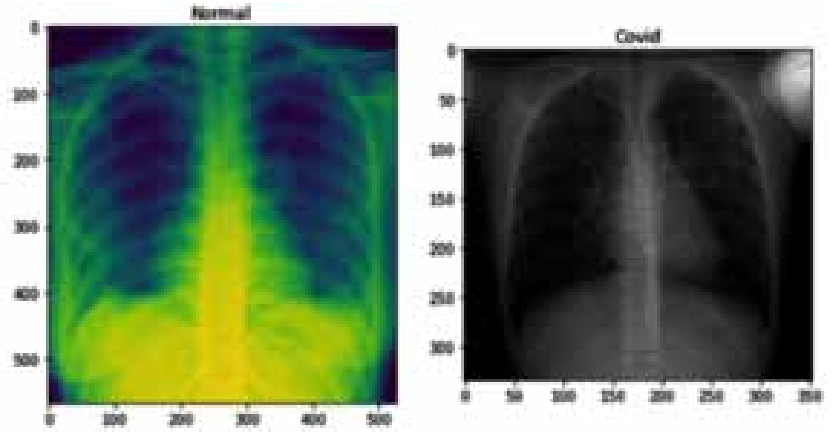
Figure 1. Model summary of the features extraction using VGG16 as a base model



Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 224, 224, 64)	1792
conv2d_2 (Conv2D)	(None, 224, 224, 64)	36928
max_pooling2d_1 (MaxPooling2	(None, 112, 112, 64)	0
conv2d_3 (Conv2D)	(None, 112, 112, 128)	73856
conv2d_4 (Conv2D)	(None, 112, 112, 128)	147584
max_pooling2d_2 (MaxPooling2	(None, 56, 56, 128)	0
conv2d_5 (Conv2D)	(None, 56, 56, 256)	295168
conv2d_6 (Conv2D)	(None, 56, 56, 256)	590080
conv2d_7 (Conv2D)	(None, 56, 56, 256)	590080
max_pooling2d_3 (MaxPooling2	(None, 28, 28, 256)	0
conv2d_8 (Conv2D)	(None, 28, 28, 512)	1180160
conv2d_9 (Conv2D)	(None, 28, 28, 512)	2359808
conv2d_10 (Conv2D)	(None, 28, 28, 512)	2359808
max_pooling2d_4 (MaxPooling2	(None, 14, 14, 512)	0
conv2d_11 (Conv2D)	(None, 14, 14, 512)	2359808
conv2d_12 (Conv2D)	(None, 14, 14, 512)	2359808
conv2d_13 (Conv2D)	(None, 14, 14, 512)	2359808
max_pooling2d_5 (MaxPooling2	(None, 7, 7, 512)	0
flatten_1 (Flatten)	(None, 25088)	0
dense_1 (Dense)	(None, 4096)	102764544
dropout_1 (Dropout)	(None, 4096)	0
dense_2 (Dense)	(None, 4096)	16781312
dropout_2 (Dropout)	(None, 4096)	0
dense_3 (Dense)	(None, 2)	8194

Total params: 134,268,738
Trainable params: 134,268,738
Non-trainable params: 0

Figure 2. Samples of the labeled x-ray images from DS2



DATA PREPROCESSING

Three main approaches are possible concerning data augmentation to prevent overfitting while using transfer learning in CNNs:

- Using the pre-trained model as a feature extractor, without any weights tuning, and training the authors' classifier.
- Fine-tuning the pre-trained model, using the actual weights as initial values.
- Using the pre-trained model directly as a final classifier.

In this study, the authors used the first approach. They used the VGG16 convolutional layers to extract features from row X-ray images. The generated vector contained 25,088 real values for each input image; the authors considered these values features for their StackNet classifier. During this phase, they carried out several transformations of the generated features vector so that it could be used to construct the classification model. Indeed, the first standardized the features by removing the mean and scaling to unit variance.

FEATURE ENGINEERING

During their analysis, the researchers studied feature selection to reduce the features' dimensionality and have a relevant number which would be helpful for effective learning without overfitting.

This approach has a considerable simplification impact since the complexity of the problem increases exponentially with the number of dimensions, so reducing them typically saves processing time and improves output quality and predictive performance.

The authors used several feature selection approaches and eliminated the features with a minimum score for all algorithms. The first method they used was filter-based, in which they calculated the variance of each feature and then removed features with the variation below a given threshold (Figure 4). A feature generally has very little predictive power when it does not vary much within itself.

The authors' second method for feature selection was feature importance with a forest of trees; the algorithm suggests more informative features to consider during the classification task. For example, the researchers simplified the complexity by ten during their first training scenario using the DS1 dataset (Figure 5).

Figure 3. Overview of the training and test data (the extracted features vector reduced to two components using principal component)

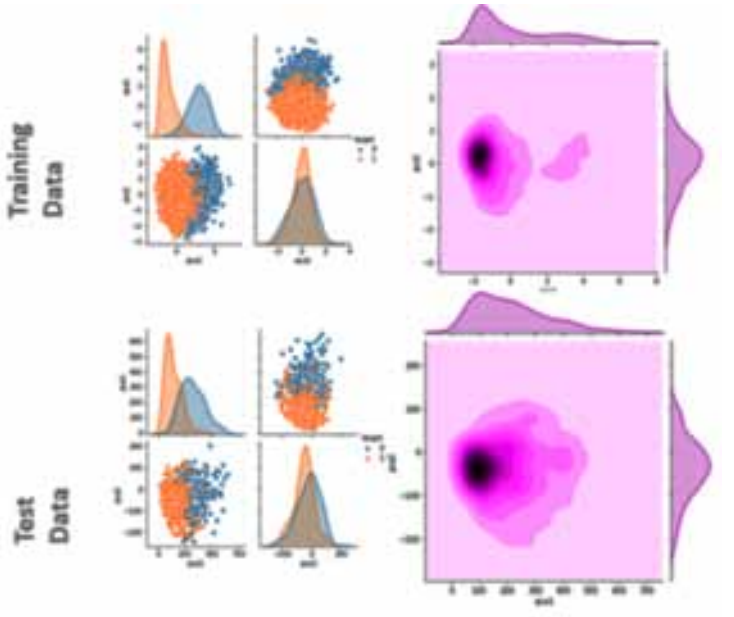
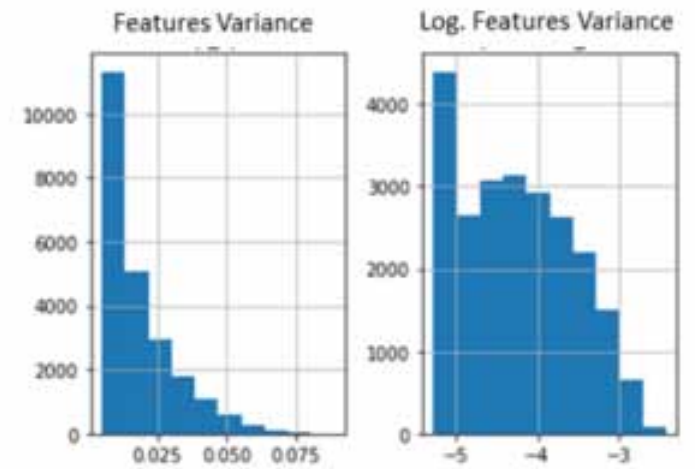


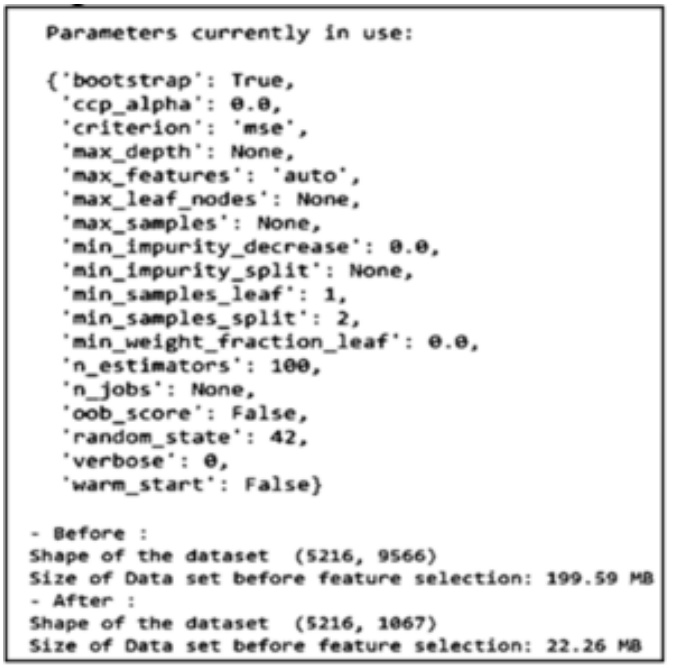
Figure 4. Features variance and logarithm of variance distribution



IMBALANCED CLASS HANDLING

This type of configuration is often encountered in the case of supervised learning problems when a significant imbalance exists in the number of observations of the target classes. X-ray images are one of the situations where the number of infected cases is relatively rare. This must be considered during the data analysis process, from the data loading to the validation of the optimal model to be retained. One of the most famous techniques for handling imbalanced datasets is the synthetic minority oversampling technique (SMOTE) (Chawla et al., 2002).

Figure 5. Features selection using random forest regressor sample parameters and output performance using DS1



This algorithm oversamples the examples in the minority class in the training dataset using the k-nearest neighbors' algorithm.

COVSTACKNET ARCHITECTURE

After the features extractions from x-ray images, the authors used the VGG16 flatten layer, whose role is to apply a flattening transformation on the tensor converting the two-dimensional matrix of features into a vector that can be fed into the authors' Stacknet. Figure 6 shows the full computing algorithm.

Many machine learning models allow for building a Stacknet (i.e., boosting, bagging, and neural networks) (Kim et al., 2020). The authors decided to combine different machine learning algorithms they had selected from a large set of algorithms (logistic regression, linear discriminant analysis, k-nearest neighbors classifier, decision tree classifier, Gaussian naïve bias, support vector classifier, adaptive boosting classifier, gradient boosting classifier, random forest classifier, extra trees classifier, and bagging classifier).

Using the grid search cross-validation approach, they optimized the classifiers via hyperparameters estimation (Figure 7). This technique is recommended to avoid overfitting and keep an almost constant predictive performance between training and unseen data by combining hyperparameters in the grid. Then, the researchers tested the model they had obtained on a test dataset (X_test, y_test); the scores determined the optimal parameters combination of the most efficient model.

Before passing the prepared data (nx1 vector) to their StackNet, the researchers made a SMOTE sampling, as explained above. This technique handles the imbalanced dataset that characterizes the authors' problem. They generated three datasets at this stage: Training, validation, and testing.

To build a robust and efficient model, the authors opted for using various models based on different algorithms; they implemented their solution using the Scikit-Learn package of the Python

Figure 6. Algorithm for automatic detection and classification of pneumonia

Algorithm 1 : Automatic Detection and Classification of Pneumonia

Input:
o_1 → Training Set (n features, s_1 samples)
o_2 → Testing Set (n features, s_2 samples)
K → number of folds
τ → Variance Threshold
n → o_1 number of features
m → Number of $level_0$ models
I → Maximum number of iterations
f = n/m → number of features by feature subset (split)
M_0 → $Level_0$ base model
M_1 → $Level_1$ models' list
M_2 → $Level_2$ models' list

Output:
m_{opt}, K_{opt}, τ_{opt}, M_{opt}: Best model hyper-parameters

Begin:
1) Determine M_0, M_1, M_2
2) Set K, τ, m value ranges K_r, τ_r, m_r respectively
3) Convert training set images into 224x224

for i=0 to I do
4) Randomly select K_i, τ_i & mi from K_r, τ_r & m_r
5) Apply Variance threshold features selection on o1 & o2 using τ_i
6) Split the new_o_1 from step 5 into mi features subsets
7) Train mr instances of M_0 using K_i-folds
8) Return $S_{0,i}$ scores vector of iteration i (mr x (#Scores)
9) Return $P_{0,i}$, M_0 Stacked predictions matrix for features subsets
10) Stacking of new_o_1 and $P_{0,i}$ (Called ReStacking Operation)
11) Train level1 models using K_i-folds
12) Return $S_{1,i}$ scores matrix of iteration i (card(M_1) x(#Scores))
13) Return $P_{1,i}$, M_1 Stacked predictions matrix for step 10 output
14) Stacking of step 10 output and $P_{1,i}$
15) Train $level_2$ meta-model using K_i-folds
16) Return $S_{2,i}$ scores vector of iteration i (1 x(#Scores))
17) Return $P_{2,i}$, M_2 Stacked predictions matrix for step 14 output
18) Stacking of step 10 output and $P_{2,i}$

end for
19) Return Argmax ($S_{2,i}$ (K_i, τ_i, m_i)) $0 \le i \le I$

end

language (Pedregosa et al., 2011). They also used an implementation of Stacknet called pystacknet (Michailidis et al., 2017; Michailidis & Soni, 2018).

The Stacknet comprised five levels of estimators (from 0 to 4) (Figures 14 and 15): Level 0 included 100 logistic regression classifiers applied on a split of random subsets of the original input features; each level from 1 to 4 included four estimators.

The proposed architecture enables extracting features from a labeled x-ray image training dataset through a CNN, followed by classification using a Stacknet architecture (Figure 9). The end-to-end algorithm for X-ray image classification involves feature extraction, feature engineering, and a two-step Stacknet classification phase. The experimental results section details this architecture and its training process.

COMPARATIVE STUDY ARCHITECTURES

In this subsection, the authors compare five pre-trained CNNs for the feature extraction step using the same model architecture (Figure 8): VGG-16, ResNet-50, Inception-V3, MobileNet-V2, and DenseNet.

Figure 7. Models' training, validation, and test configuration

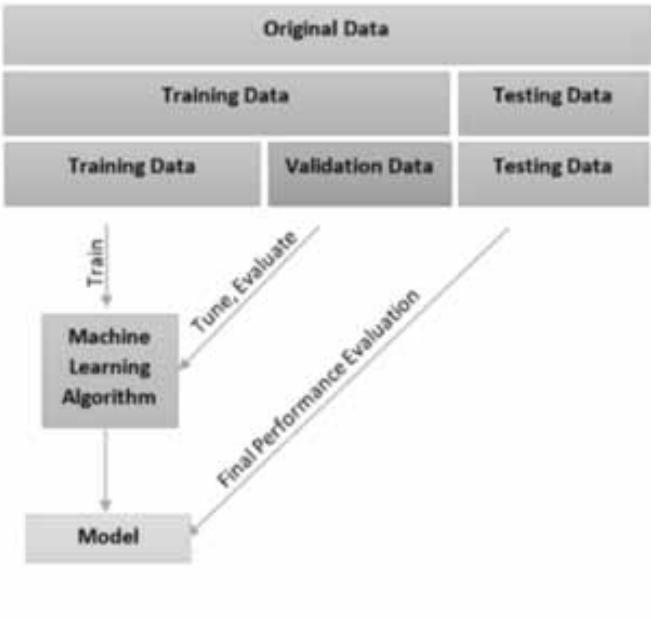
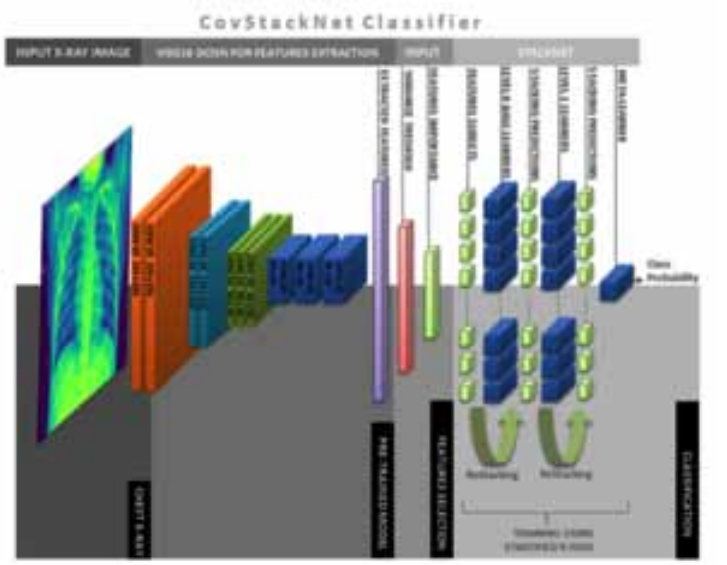
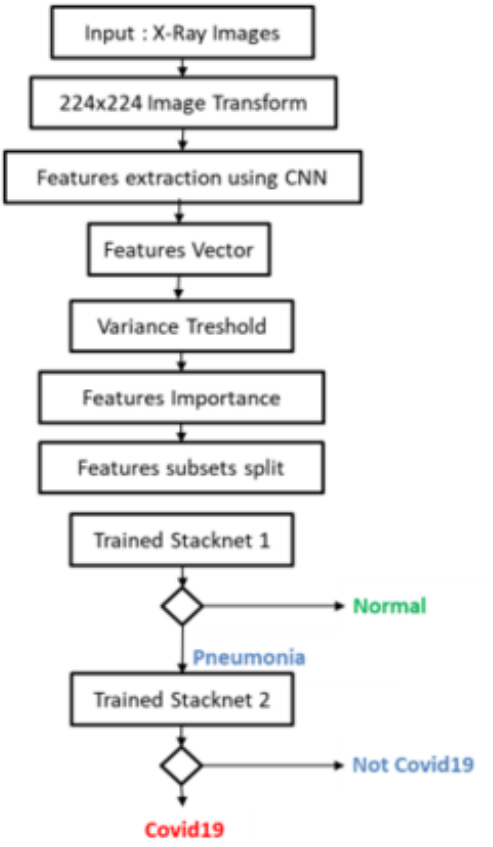


Figure 8. CovStacknet model architecture



ResNet-50 (or residual network 50 layers deep) is a type of deep network based on residual learning (Sandler et al., 2018) that considers inputs as a reference to make network training easier. Inception-V3 (Simonyan & Zisserman, 2014) by Google is the third version of a series of deep learning convolutional architectures based on depthwise separable convolution layers. Version 3 mainly focuses on burning less computational power by modifying the previous Inception architectures (Szegedy et al., 2016).

Figure 9. CovStacknet model architecture



MobileNet-V2 (Sandler et al., 2018) is a lightweight CNN with 53 layers (52 convolutions and one fully connected layer). DenseNet (Huang et al., 2017) is a kind of CNN where every layer obtains additional inputs from all preceding layers and passes on its feature maps to all subsequent layers. The concept of concatenation gives the model a higher computational and memory efficiency.

The authors used the same model architecture to conduct the comparative study, changing just the features extraction layer (Figure 10). Table 1 shows the dataset the authors used. After the feature extraction step using the five pre-trained CNNs, the authors applied the variance threshold algorithm to reduce dimensionality and selected the most pertinent features. Table 2 shows the number of features before and after the feature selection step. Next, the researchers split the dataset into training and validation subsets and applied SMOTE sampling to handle unbalanced datasets. Finally, they trained and validated the model using the StackNet architecture (Figure 11) with intermediate-level prediction restacking. In level 0 of the StackNet, the researchers divided the data into 100 subsets; each subset had the total number of features divided by 100.

EVALUATION METRICS

To evaluate their approach and estimate the generalization accuracy of their models on future data, the authors used a set of metrics calculated based on the confusion matrix (Figure 12). This allowed

Figure 10. CovStackNet architecture for features extraction comparative study

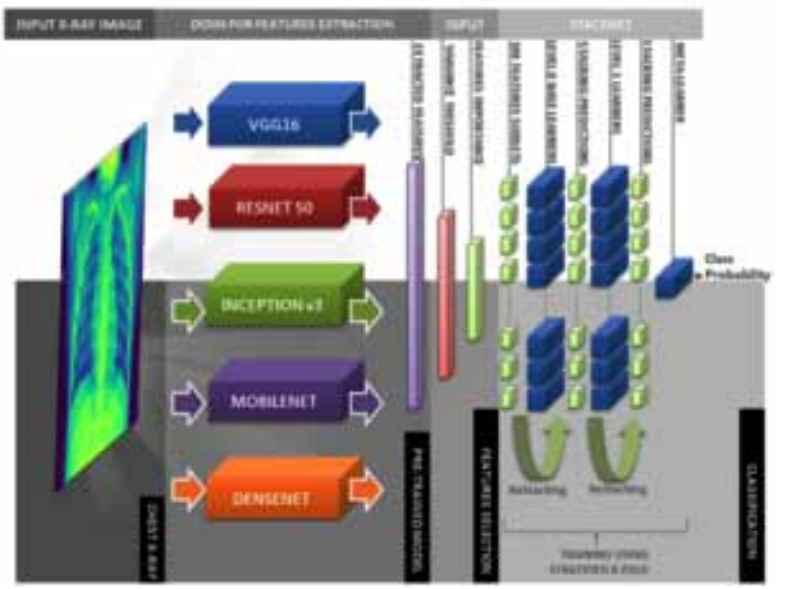


Table 1. Initial x-ray images content

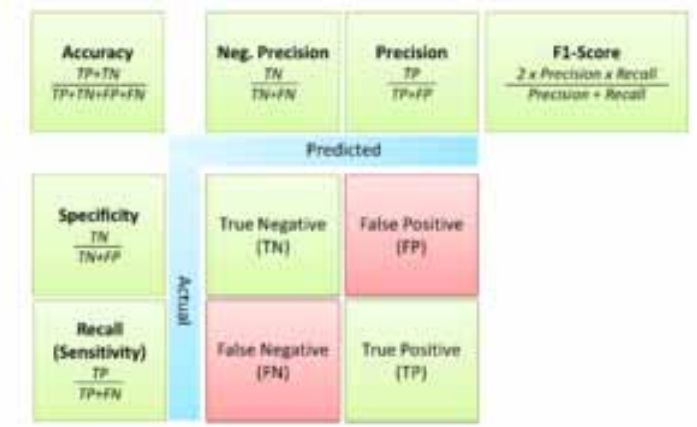
Scenario	S1	S2		S3
Dataset	DS1	DS2	DS3	DS1 + DS2 + DS3
Normal	1583	54	147	0
Pneumonia	4273			2779
Covid-19	0	108	526	634
	5216	162	673	3413

Table 2. Performance summary for the features extraction comparative study

CNN	N° of features		CovStackNet classification performance						
	Before	After	Data set	Accuracy	Precision	Recall	F1 Score	Specificity	Execution Time
VGG16	25088	1879	Training	99.2%	99.7%	98.7%	99.2%	99.7%	260.42 s
			Validation	99.1%	98.6%	97.4%	98.0%	99.6%	258.35 s
ResNet 50	2048	320	Training	99.8%	99.9%	99.7%	99.8%	99.9%	94.94 s
			Validation	99.1%	98.6%	97.4%	98.0%	99.6%	93.68 s
Inception V3	2048	519	Training	98.0%	99.8%	96.3%	98.0%	99.8%	191.60 s
			Validation	96.5%	92.8%	91.6%	92.2%	97.9%	200.76 s
MobileNet V2	1024	394	Training	99.8%	99.8%	99.7%	99.8%	99.8%	125.00 s
			Validation	98.1%	98.6%	92.9%	95.7%	99.6%	113.05 s
DenseNet	1024	118	Training	99.8%	100%	99.7%	99.8%	100%	64.46 s
			Validation	98.1%	96.7%	94.8%	95.7%	99.1%	60.75 s

Figure 11. StackNet architecture used to build covstacknet classifier

Figure 12. Confusion matrix and performance metrics used for comparative analyses



them to experiment with two approaches to evaluate their model performance. The set of metrics was as follows:

- **Classification Accuracy:** The percentage of correct predictions compared to the set of all predictions.
- **Receiver Operating Characteristic Curve (ROC):** It allows comparing the true positive rate and the false positive rate for different classification thresholds for a classifier.
- **Area Under Curve (AUC):** It calculates the entire two-dimensional areas underneath the ROC curve (Figure 7).
- **Precision:** It gives the level of precision of the model on positive predictions (i.e., the portion of correct positive predictions).
- **Recall:** It gives the model's precision level on the rate of true positive predictions compared to the overall number of actual positives.
- **F-Measure:** Also called F1-score, it balances precision and recall. This measure gives more precise information than the accuracy score, in particular when the problem contains a large number of true negatives or, in the case of imbalanced datasets.

EXPERIMENTAL RESULTS AND DISCUSSION

To obtain optimal classification results, the authors trained two models (Figure 10); the first model allows to distinguish standard images from those having any pneumonia infection (dataset DS1 [Mooney, 2018] and scenario S1 in Table 1). Training the model based on the DS2 (Bashir, 2020) dataset does not give good scores due to the limited number of cases (DS2 dataset and S2 scenario in Table 1); Figure 11 shows the results. This led the authors to create a new dataset containing images with the Covid-19 infection taken from DS2 and DS3 (Cohen et al., 2020) (Table 1) and cases with pneumonia infections other than Covid-19 taken from DS1. Thus, the resulting classifier could distinguish the characteristics of Covid-19 infections from different types of pneumonia (Figure 13).

In the first scenario (Stage I), the authors extracted features using VGG-16 CNN from all 5216 DS1 x-ray images after feature engineering and selection, as explained in the fourth section.

The researchers did an 80%-20% train/test split and applied SMOT sampling on the training dataset.

Figure 6 shows how the authors trained the model. The model scored more than 97% accuracy on the validation set to classify X-ray images into two classes, namely "No infection (normal)" and "Pneumonia infection."

To detect Covid-19 infections, the researchers trained a new model using the same steps (Stage II). In this scenario (S3), the used dataset contained only infected cases, and the classes for this scenario were "Covid-19 infection" and "Other pneumonia infection." The second model scored more than 98% accuracy (Figures 14 and 15).

Using this two-stage approach by combining S1 and S3 allowed us to obtain very high classification scores vs just one stage, as in S2. This approach of problem simplification into two steps gives each classifier more capability to learn specific characteristics, take more accurate decisions to

Figure 13. Covid-19 classification architecture using two instances of CovStackNet with different training data

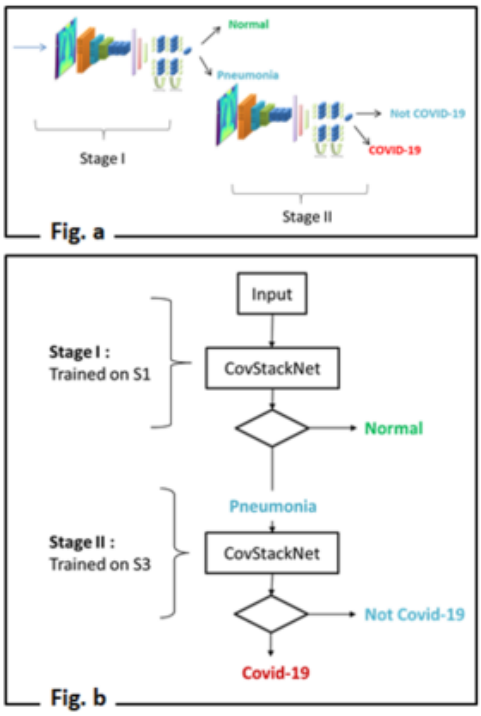


Figure 14. Covid-19 classification performance for all CovStackNet models prediction by level and model number

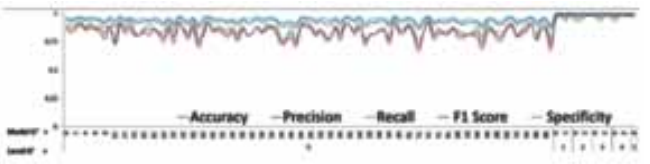
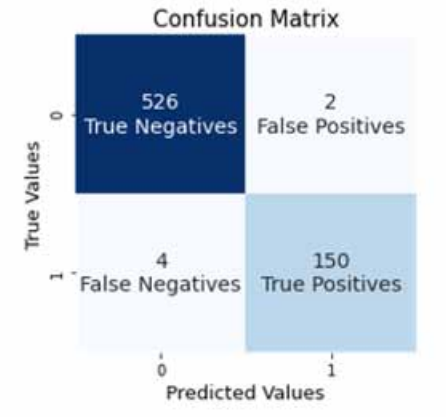


Figure 15. Confusion matrix for CovStackNet level 5 prediction



detect infected cases of any pneumonia, including Covid-19, and then, among those infected cases, apply another classifier trained to detect Covid-19 cases.

In this study, the authors combined two well-known techniques in machine learning to create a super learner able to detect Covid-19 infections with very high performance. The first technique is CNNs, especially five pre-trained architectures to extract x-ray pictures of the features (i.e., VGG-16, ResNet-50, Inception-V3, MobileNet-V2, and DenseNet). The second technique is a six-level StackNet classifier with intermediate prediction restacking. The experimental results showed that all the models converged to an optimal value of performance parameters since the second level (Figure 20). The best results were for VGG16 and ResNet-50 (Figures 17, 18, and 19). Although the VGG16 network performed highly in classifying Covid-19 and non-Covid-19 infections, it did not have the best execution time. In contrast, ResNet-50 could diagnose Covid-19 infection quickly and use fewer features.

High sensitivity is desired to diagnose the maximum positive cases of Covid-19 infections. In this regard, ResNet and VGG16 have the same values, with an execution time performance advantage for ResNet-50. These results prove that the model the authors built in this study and based on deep learning techniques combined with ensemble learning, can accurately aid radiologists in diagnosing infections related to Covid-19. In this study, the researchers also solved the most critical issue for radiologists: differentiating Covid-19 and other atypical and viral pneumonia diseases by using a double training process (Figure 13).

IMPLEMENTATION

To implement their proposed solution in a production environment, the authors developed a Web application using Python. This application serves as a tool for detecting Covid-19 positive and negative

Figure 16. Covid-19 classification probability distribution and ROC curve for the three-stages predictions

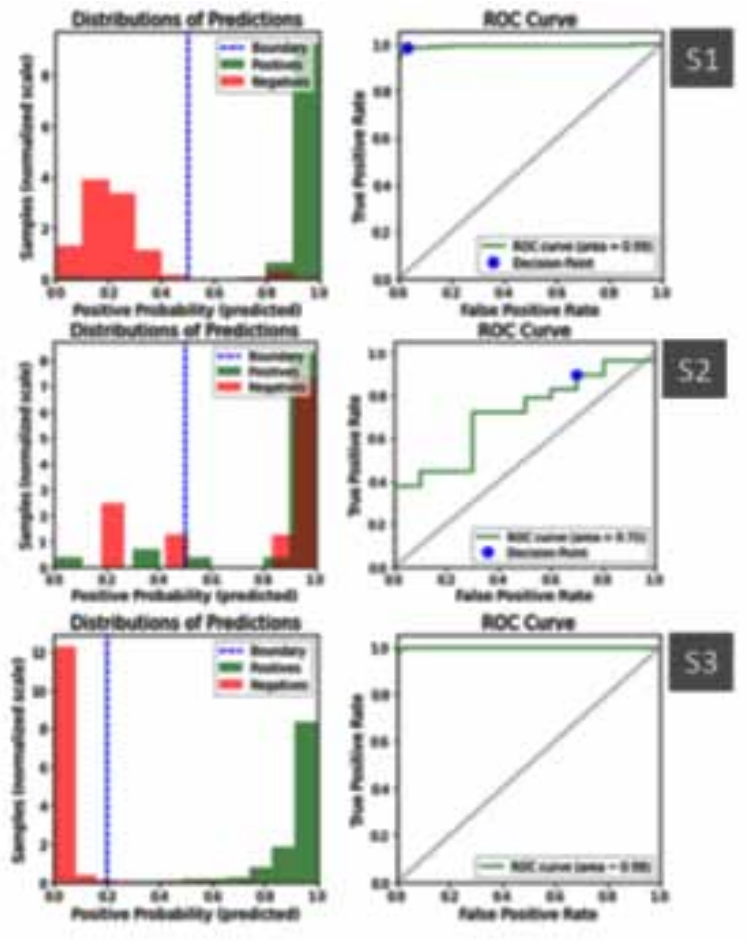
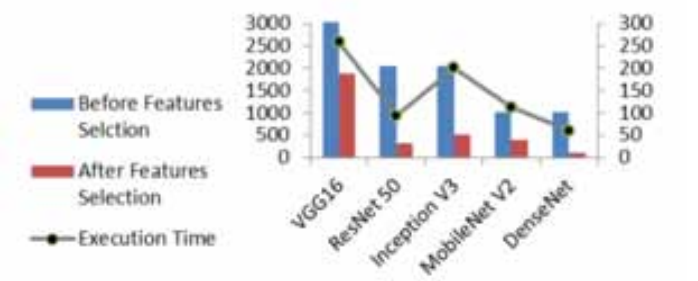


Figure 17. Execution time and features before and after selection by DCNN architecture



cases based on the authors' model. Medical personnel can easily access the application and upload CXR images for analysis (Figures 21 and 22). This Web application offers several advantages in the context of Covid-19 detection. Firstly, it provides a user-friendly interface that simplifies the process for medical personnel. They can easily navigate the application, browse the available CXR images,

Figure 18. Classification performance by CNN

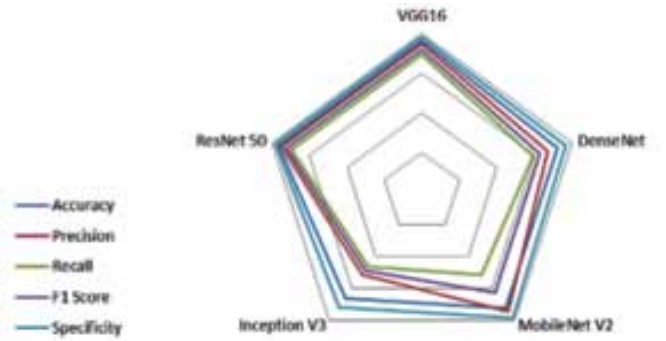
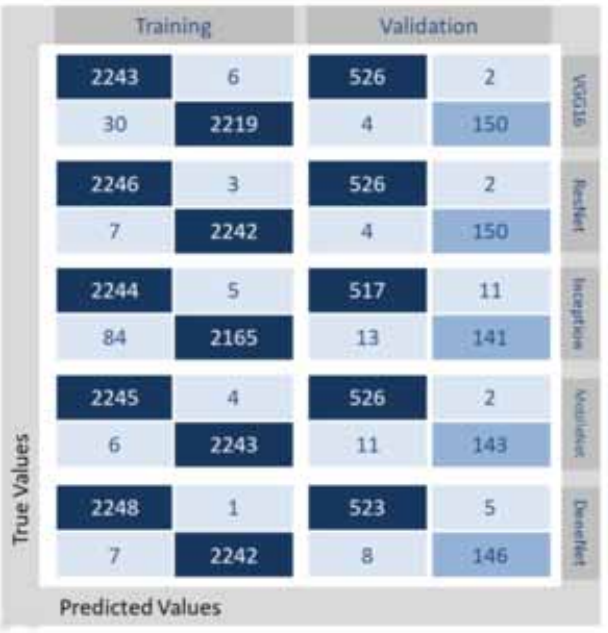


Figure 19. Confusion matrices for training and validation data



and submit them for analysis. This intuitive design reduces the learning curve and ensures healthcare professionals can efficiently utilize the solution. Secondly, the application leverages the power of our developed model for Covid-19 detection. By feeding the CXR images into the application, the model can quickly analyze the data and provide accurate results regarding the presence of Covid-19 in an optimal time (Table 2). Compared to traditional diagnostic methods, this expedited process saves time, allowing for faster decision-making and patient management.

Additionally, the Web application offers the advantage of accessibility. Medical personnel can access the application from any device with Internet connectivity, such as laptops, tablets or smartphones. This flexibility enables healthcare professionals to conveniently perform Covid-19 screenings in various settings, including hospitals, clinics or remote locations.

Figure 20. Classification accuracy (training vs validation) for StackNet algorithms for each DCNN

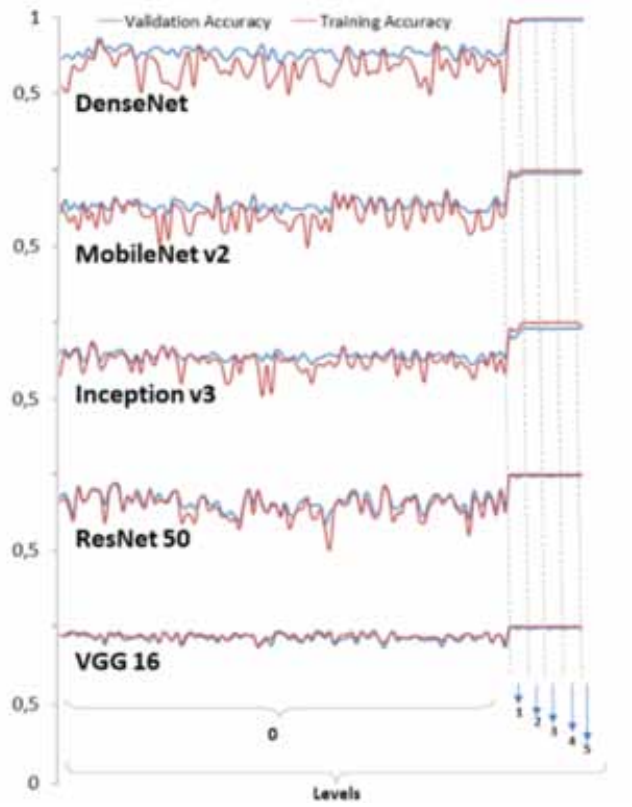


Figure 21. Graphical user interface-based tool for Covid-19 detection: browse and analyze image

Figure 22. Graphical user interface-based tool for COVID-19 detection: result of the test

LIMITATIONS AND CONSIDERATIONS OF THE PROPOSED METHOD

It is essential to acknowledge and discuss the limitations of the authors' proposed method to provide a comprehensive and balanced view of this study. Firstly, the authors' proposed procedure is invasive compared to the widely used PCR technique. This invasiveness may pose challenges regarding patient acceptance and the practicality of widespread implementation. Secondly, this study's data collection process and analysis are not fully automated, which may introduce potential biases or inconsistencies. Additionally, the proposed solution may not be broadly deployable.

CONCLUSION

This paper proposes a new classification model that combines DCNN and ensemble learning to predict pneumonia, especially Covid-19. This is important to detect new infections quickly and save lives. The proposed method exhibits several novel aspects that contribute to its effectiveness in detecting Covid-19 infections from radiological images. Firstly, the authors developed a robust classifier by training a multilevel Stacknet using a diverse set of algorithms. This approach optimized the predictive performance of the model. Additionally, the authors applied feature segmentation into subsets at the first level of the Stacknet, enhancing processing time and classification performance. They also utilized pretrained CNNs for automatic feature extraction from X-ray images, making their model powerful and efficient. Another notable feature is the two-step classification architecture, distinguishing pneumonia cases from standard cases in the first step and accurately detecting Covid-19 cases in the second step. Furthermore, the authors conducted a comparative study, comparing the accuracy and processing time performance of their classifier with different CNN architectures for feature extraction. The researchers compared five well-known pre-trained DCNNs (i.e., VGG-16, ResNet-50, Inception-V3, MobileNet-V2, and DenseNet). Besides, they used many feature selection techniques and grid search cross-validation for hyperparameters optimization; they adopted SMOTE as the sampling technique to handle unbalanced datasets. The experimental results showed that all the models converged to an optimal value of performance parameters since the second level of the authors' StackNet classifier. However, VGG16 and ResNet-50 achieved better accuracy and AUC scores. Results can be improved

Table 3. Comparison with other related work that performed binary classification on x-ray images

Authors	Architecture	Precision
Gupta, A., Gupta, S., & Katarya, R. (2021)	InstaCovNet-19	99.53%
Nandhini, S., & Ashokkumar, K. (2022)	DenseNet121	99.48%
Hussain et al. (2021)	CoroDet	99.1%
Shelke et al. (2020)	DenseNet-161	98.9%
Nayak et al. (2021)	ResNet-34	98.33%
Ozturk et al. (2020)	DarkCovidNet	98.08%
Jain et al. (2020)	ResNet-101	97.78%
The authors of this study	CovStackNet	99.1%

by fine-tuning the hyperparameters of the StackNet model or by choosing a new set of base models. Stacknets are among the best-performing approaches to building robust prediction systems with many base models. It is also possible to make deeper Stacknets for use cases with big datasets that allow for a successful training process through all layers, using k-fold cross-validation.

Based on the above results, the authors recommend CovStackNet based on ResNet50 (99.1% accuracy, 98.6% precision, 97.4% recall, 98.0% F1 score, and the second-best processing time of 93.93 s) for Covid-19 classification of pneumonia cases on CXR. Performance improvement is possible by using DCNN architectures. Comparing the authors' CovStackNet with recently published studies that also performed binary classification, especially of Covid-19 and pneumonia images, the authors' model has the second-highest accuracy (Table 3). (Gupta, A., Gupta, S., & Katarya, R. (2021)) achieved the highest binary classification accuracy using their proposed network called InstaCovNet-19.

COMPETING INTERESTS

The authors of this publication declare there are no competing interests.

FUNDING AGENCY

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors. Funding for this research was covered by the author(s) of the article.

REFERENCES

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., & Ghemawat, S. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint* arXiv:1603.04467.

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., & Zheng, X. (2016, November). Tensorflow: a system for large-scale machine learning. In Osdi, 16, 265-283.

BashirN. (2020). *Radiographic presentation of Covid-19: A systematic review*. SSRN 3557985. 10.2139/ssrn.3557985

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*(2), 123–140. doi:10.1007/BF00058655

Chassagnon, G., Vakalopoulou, M., Paragios, N., & Revel, M. P. (2020). Artificial intelligence applications for thoracic imaging. *European Journal of Radiology*, *123*, 108774. doi:10.1016/j.ejrad.2019.108774 PMID:31841881

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, *16*, 321–357. doi:10.1613/jair.953

Chollet, F. (2015). Keras: Deep learning library for theano and tensorflow. *Keras, 7*(8). https://keras. io/k,

Cohen, J. P., Morrison, P., & Dao, L. (2020). Covid-19 image data collection. *arXiv preprint* arXiv:2003.11597.

Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 248—255). IEEE. doi:10.1109/CVPR.2009.5206848

Freund, Y., Schapire, R., & Abe, N. (1999). A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence, 14*(771-780), 1612.

Gupta, A., Gupta, S., & Katarya, R. (2021). InstaCovNet-19: A deep learning classification model for the detection of COVID-19 patients using Chest X-ray. *Applied Soft Computing*, *99*, 106859. doi:10.1016/j.asoc.2020.106859 PMID:33162872

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4700—4708). IEEE.

Hubel, D. H., & Wiesel, T. N. (1962). Receptive fields, binocular interaction, and functional architecture in the cat's visual cortex. *The Journal of Physiology*, *160*(1), 106–154. doi:10.1113/jphysiol.1962.sp006837 PMID:14449617

Hussain, E., Hasan, M., Rahman, M. A., Lee, I., Tamanna, T., & Parvez, M. Z. (2021). CoroDet: A deep learning based classification for COVID-19 detection using chest X-ray images. *Chaos, Solitons, and Fractals*, *142*, 110495. doi:10.1016/j.chaos.2020.110495 PMID:33250589

Jain, G., Mittal, D., Thakur, D., & Mittal, M. K. (2020). A deep learning approach to detect Covid-19 coronavirus with X-Ray images. *Biocybernetics and Biomedical Engineering*, *40*(4), 1391–1405. doi:10.1016/j.bbe.2020.08.008 PMID:32921862

JupyterLab. (2022). *JupyterLab documentation*. JupyterLab. https://jupyterlab.readthedocs.io/en/stable/

Kedia, P., & Katarya, R. (2021). CoVNet-19: A deep learning model for the detection and analysis of Covid-19 patients. *Applied Soft Computing*, *104*, 107184. doi:10.1016/j.asoc.2021.107184 PMID:33613140

Kim, J., Kim, J., & Kwak, N. (2020). StackNet: Stacking feature maps for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (pp. 242—243). IEEE. doi:10.1109/CVPRW50498.2020.00129

Lomoro, P., Verde, F., Zerboni, F., Simonetti, I., Borghi, C., Fachinetti, C., Natalizi, A., & Martegani, A. (2020). COVID-19 pneumonia manifestations at the admission on chest ultrasound, radiographs, and CT: Single-center study and comprehensive radiologic literature review. *European Journal of Radiology Open*, *7*, 100231. doi:10.1016/j.ejro.2020.100231 PMID:32289051

Michailidis, M. (2017). *StackNet, metamodelling framework*. Kaz Anova. https://github.com/kaz-Anova/StackNet

Michailidis, M., & Soni, A. (2018). *A light Python version of StackNet*. PY Stacknet. https://github.com/h2oai/pystacknet

Miller, M. (2020). 2019 Novel coronavirus COVID-19 (2019-nCoV) data repository: Johns Hopkins University center for systems science and engineering. [ACMLA]. *Bulletin-Association of Canadian Map Libraries and Archives*, (164), 47–51. doi:10.15353/acmla.n164.1730

Monshi, M. M. A., Poon, J., & Chung, V. (2020). Deep learning in generating radiology reports: A survey. *Artificial Intelligence in Medicine*, *106*, 101878. doi:10.1016/j.artmed.2020.101878 PMID:32425358

Mooney, P. (2018). [Chest x-ray images, pneumonia] https://www. kaggle.com/paultimothymooney/chest-xray-pneumonia, tanggal akses.

Nandhini, S., & Ashokkumar, K. (2022). An automatic plant leaf disease identification using DenseNet-121 architecture with a mutation-based henry gas solubility optimization algorithm. *Neural Computing & Applications*, *34*(7), 1–22. doi:10.1007/s00521-021-06714-z

Nayak, S. R., Nayak, D. R., Sinha, U., Arora, V., & Pachori, R. B. (2021). Application of deep learning techniques for detection of COVID-19 cases using chest X-ray images: A comprehensive study. *Biomedical Signal Processing and Control*, *64*, 102365. doi:10.1016/j.bspc.2020.102365 PMID:33230398

Ozturk, T., Talo, M., Yildirim, E. A., Baloglu, U. B., Yildirim, O., & Acharya, U. R. (2020). Automated detection of COVID-19 cases using deep neural networks with X-ray images. *Computers in Biology and Medicine*, *121*, 103792. doi:10.1016/j.compbiomed.2020.103792 PMID:32568675

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Rabbah, J., Ridouani, M., & Hassouni, L. (2020, October). A new classification model based on stacknet and deep learning for fast detection of COVID 19 through X rays images. In *2020 Fourth International Conference On Intelligent Computing in Data Sciences (ICDS)* (pp. 1-8). IEEE. doi:10.1109/ICDS50568.2020.9268777

Rajpurkar, P., Irvin, J., Zhu, K., Yang, B., Mehta, H., Duan, T., & Ng, A. Y. (2017). *Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning.* arXiv preprint arXiv:1711.05225.

Shelke, A., Inamdar, M., Shah, V., Tiwari, A., Hussain, A., Chafekar, T., & Mehendale, N. (2021). Chest X-ray classification using deep learning for automated COVID-19 screening. *SN Computer Science*, *2*(4), 300. doi:10.1007/s42979-021-00695-5 PMID:34075355

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint,* arXiv:1409.1556.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2818-2826). IEEE. doi:10.1109/CVPR.2016.308

Wang, W., Xu, Y., Gao, R., Lu, R., Han, K., Wu, G., & Tan, W. (2020). Detection of SARS-CoV-2 in different types of clinical specimens. *Journal of the American Medical Association*, *323*(18), 1843–1844. doi:10.1001/jama.2020.3786 PMID:32159775

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, *5*(2), 241–259. doi:10.1016/S0893-6080(05)80023-1 PMID:18276425

World Health Organization. (2023). [WHO Coronavirus (COVID-19) Dashboard]. WHO. https://covid19.who.int/

Zhou, S., Wang, Y., Zhu, T., & Xia, L. (2020). CT features of coronavirus disease 2019 (Covid-19) pneumonia in 62 patients in Wuhan, China. *AJR. American Journal of Roentgenology*, *214*(6), 1287–1294. doi:10.2214/AJR.20.22975 PMID:32134681

Jalal Rabbah received his degree of engineer "diplôme d'ingénieur d'état" in telecommunications from National Institute of Post and Telecommunications (Rabat, Morocco) in 2003. He obtained his PhD degree in Applied Mathematics, Computers sciences and Telecommunication at the Hassan I University (Settat, Morocco) in 2016. He Started his professional career as IT engineer at Chauib Doukkali University (El Jadida, Morocco) for two years, and then occupied multiple position at Orange Morocco, first as IT Project Manger at the IT department, Telco Traffic Mediation Expert, and finally as Fraud and Revenue Assurance Expert. And since November 2020 he is Data Science Manger at the Fraud and Revenue Assurance Shared Service (Dakar, Seneg), to manage this Risk management activitie for Orange Middle East and Africa region. He was certified on ITIL in 2007 and PMP in 2011, ISO27001, Lead Auditor, ISO 21500 Project Manager and PECB Certified Trainer between 2017 and 2019, he is also Six Sigma Green Belt Certified from Orange continuous improvement program. He serves as a reviewer for many international journals. He was a recipient of the 2021 Best Paper Award of the International Conference. His research interests is the machine learning performance optimisation for telco use cases, using Ensembles theory. Also his current research interests are data sciences, especially machine learning applications for healthcare and Telecommunication, and Big data.

Mohammed Ridouani received his degree of engineer in telecommunications from National Institute of Post and Telecommunications (Rabat, Morocco) in 2005. He obtained his PhD degree in Applied Mathematics, Computers sciences and Telecommunication at the Hassan I University (Settat, Morocco) in 2016. He has been an Engineer-Teacher at the Computer Sciences department in High School of Technology at the Hassan II University (Casablanca, Morocco) during 2006-2018, and he was appointed as a Professor in 2018. He was certified on CISCO in 2009 and Linux LPIC-1, LPIC-2 and LPIC-3 (senior/expert level) in 2008, 2012 and 2017 respectively. He is a LPI and Cisco instructor and he has served as an international instructor at the Francophone University Agency. He was a recipient of the 2021 Best Paper Award of the International Conference on Smart Systems and Data science 2021 (ICSSD'21). His research interests are in the area of 6G wireless communications, mathematical modeling and IOT. In addition, his current research interests are data sciences and smart city, especially Artificial Intelligence applications for Healthcare, Telecommunication, and Security.

Larbi Hassouni got his engineer degree in 1983 from the "École Centrale de Marseille". He prepared his Phd degree in 1987 at the University of Aix Marseille III in France. Among his research work, there is the development of a list unification software with LeLisp language of INRIA in order to contribute to the realization of the inference engine of an expert system for the digital circuits' diagnossis. He also developed a behavioral symbolic simulator of digital circuits using the C, FRL, and LeLisp languages in order to contribute to the development of a "formal proof" tool for correcting a design of material produced by a Hardware Design Language (HDL). Currently, his research work mainly concerns Data Sciences.