Detection and classification of lung diseases in distributed environment

Thuong-Cang Phan [a,1,*], Anh-Cang Phan [b,2]

[a] Can Tho University, Vietnam
[b] Vinh Long University of Technology Education, Vietnam
[1] ptcang@cit.ctu.edu.vn; [2] cangpa@vlute.edu.vn
* corresponding author

ARTICLE INFO

Article history
Received October 21, 2024
Revised January 28, 2025
Accepted March 21, 2025
Available online May 31, 2025

Keywords
Big data
Spark
Lung diseases classification
Deep learning
Medical imaging

ABSTRACT

A significant increase in the size of the medical data, as well as the complexity of medical diagnosis, poses challenges to processing this data in a reasonable time. The use of big data is expected to have the upper hand in managing the large-scale datasets. This research presents the detection and prediction of lung diseases using big data and deep learning techniques. In this work, we train neural networks based on Faster R-CNN and RetinaNet with different backbones (ResNet, CheXNet, and Inception ResNet V2) for lung disease classification in a distributed and parallel processing environment. Moreover, we also experimented with three new network architectures on the medical image dataset: CTXNet, Big Transfer (BiT), and Swin Transformer, to evaluate their accuracy and training time in a distributed environment. We provide ten scenarios in two types of processing environments to compare and find the most promising scenarios that can be used for the detection of lung diseases on chest X-rays. The results show that the proposed method can accurately detect and classify lung lesions on chest X-rays with an accuracy of up to 96%. Additionally, we use Grad-CAM to highlight lung lesions, thus radiologists can clearly see the lesions' location and size without much effort. The proposed method allows for reducing the costs of time, space, and computing resources. It will be of great significance to reduce workloads, increase the capacity of medical examinations, and improve health facilities.

1. Introduction

Recent studies have demonstrated the effectiveness of machine learning techniques in supporting medical diagnosis. Especially in medical imaging, deep learning algorithms can perform on par with human physicians. There are some researches using chest X-rays to detect and classify lung diseases. Emtiaz Hussain et al. [1] built CoroDet, a 22-layer CNN model, to detect Covid-19 on chest X-rays and CT scans. The model was tested for 2, 3, and 4 classes (infected, normal, viral pneumonia, and bacterial pneumonia) and received the accuracy of 99.1%, 94.2%, and 91.2%, respectively. The system performance was shown to be better than existing state-of-the-art methods in terms of accuracy. A limitation mentioned in their future goal was the hardware issue to work on larger image sets. Ali Narin et al. [2] used five pre-trained network models to detect coronavirus pneumonia-infected patients using chest radiographs. The authors implemented three binary classifications with 4 classes (similar to 4 classes of [1]) using five-fold cross-validation. The results showed that the pre-trained ResNet-50 provides the highest classification performance of 98.4% on average. Rachna Jain et al. [3] presented

CNN models to detect pneumonia using X-rays. The authors trained six models, of which four were trained and tested on the ImageNet dataset (VGG16, VGG19, ResNet50, and Inception-V3). The experimental results showed an accuracy of up to 92.31%. Their proposed model had a high Recall of 98%, which is important in evaluating the number of false negatives in medical imaging. The authors presented their future concerns regarding the efficiency and accuracy of models in real-time applications using larger datasets. Asnaoui and Chawki [4] conducted a comparative study of the use of some recent deep learning models to detect and classify coronavirus pneumonia. The authors provided experiments on the chest X-rays/CT dataset of 6,087 images. They found that Inception ResNet V2 and DenseNet201 yielded better results than other models, with an accuracy of 92.18% and 88.09%, respectively. Laboni Sarker et al. [5] proposed a deep learning based approach (CheXNet) to detect COVID-19 patients effectively. The authors trained and tested their model on a 13,800 chest radiography images dataset across 13,725 patients. They performed both two-class and three-class classifications and achieved an accuracy of 96.49% and 93.71%, respectively. The experimental dataset was highly skewed because of the limitation in open-source data for COVID-19 radiology images. To deal with this issue, the authors augmented only the COVID-19 images in the training dataset. Thus, their model can be tested further with more data if available. Liu et al. [6] proposed a new vision Transformer architecture called Swin Transformer that utilizes shifted windows to compute hierarchical feature maps efficiently. The authors introduced a shifted window partitioning approach, where self-attention is computed within local windows, and the windows are shifted between consecutive layers to enable cross-window connections. This allows the model to capture local and global dependencies while maintaining linear computational complexity concerning the input image size. Kolesnikov et al. [7] introduced Big Transfer (BiT), a scalable approach to representation learning that utilizes large-scale supervised pre-training followed by task-specific fine-tuning. The authors explored the interplay between model capacity, dataset size, and computational budget, and demonstrated that transfer performance can be significantly improved by scaling up these factors in tandem. Nahida Habib et al. [8] proposed an ensemble method to diagnose pneumonia on chest X-rays. Two network models of VGG19 and CheXNet are used to extract features, and then these features are ensembled for classification. The ensembled feature vector is classified using Random Forest, Adaptive Boosting, and K-Nearest Neighbors. Random Forest provided a better performance of 98.93%. The system could be tested with two or more lung diseases for evaluating the effectiveness of the proposed method. Hasan et al. [9] proposed the CTXNet model, based on the Compact Convolutional Transformer (CCT) architecture that combines CNNs and Vision Transformers, achieving remarkable accuracies of 99.77% and 95.37% on CT and X-ray images, respectively, for classifying lung diseases. CTXNet was demonstrated with faster training times, requiring only 10 to 12 seconds per epoch for CT scans and 40 to 42 seconds per epoch for X-rays, while traditional models take 61 to 90 seconds and 170 to 175 seconds per epoch, respectively.

Ei Khaing and Thu Zar Aung [10] proposed a hybrid CNN-LSTM architecture to classify lung diseases using two publicly available Kaggle datasets. A comprehensive dataset was built, including 3,616 COVID-19, 350 lung opacity, 500 tuberculosis, 10,192 normal, and 1,345 viral pneumonia chest X-ray images. Ruaa N. Sadoon et al. [11] presented a method to classify lung diseases based on chest X-rays using transfer learning techniques and deep learning models. Their method achieved accuracies of 95.49%, 94.89%, and 93.69%, respectively, for DenseNet201, MobileNetV3, and VGG19. The proposed models were tested on real-world images, demonstrating their predictive capabilities. Kavitha S et al. [12] develop a deep learning model for classifying lung diseases using the NIH Chest X-ray dataset, achieving the highest accuracy at 83.57% with ResNet50, followed by a custom CNN at 78.25%. This work provided an insight into the performance of each illness class's classification in the experimental dataset. Rekha H et al. [13] proposed a method for the automated classification of lung diseases using VGG16, VGG19, and DenseNet201, with achieved accuracies of 95%, 96%, and 97%, respectively. The DenseNet201 was shown to provide superior performance among the three architectures for all three lung disease classifications. Bhookya [14] proposed a convolutional neural network model to classify three types of lung diseases on an X-ray dataset, including COVID-19, pneumonia, and tuberculosis, achieving a 94% accuracy in lung disease classification. Patel et al. [15] present a framework using EfficientNet-B4 through a transfer learning technique for classifying lung diseases in chest X-rays,

achieving 96% accuracy. The authors integrated GradCAM to generate insightful heatmaps highlighting critical regions within images. In practical implementation, the inclusion of multi-modal data is needed to improve the accuracy of disease classification. Azmat Ali et al. [16] presented the CX-RaysNet framework for effectively classifying lung diseases in digital chest X-ray images, achieving a multi-class classification accuracy of 97.25%. The core underlying of the framework is the integration of both convolutional and groups of convolutional layers, along with small filter sizes and dropout regularization. Prathibha TP and Pulna M Arabi [17] presented a novel method that is proposed to identify and classify high-resolution CT images of cancerous lung using machine learning and a K-Nearest Neighbor classifier with 94% accuracy on 996 images. A future direction of the study is to work on a larger dataset and include parenchymal pattern identification in lung images to determine the type and the extent of lung diseases. Padmanabha Reddy et al. [18] propose a model based on a combined architecture of VGG-19 and CNN to classify lung diseases based on a lung X-ray dataset, with an accuracy of 91.57% on the test dataset. Poonam Rana et al. [19] developed an automated classification scheme for lung cancer presented in histopathological images using a Dense AlexNet framework with an accuracy of 93% in the training phase. The experiments were conducted on a dataset of 15,000 histopathological images with three classes: benign tissue, adenocarcinoma, and squamous cell carcinoma. Gupta et al. [20] proposed a model that employs two convolutional neural networks (Inception-V3 and Xception) for tuberculosis classification using the ImageNet dataset, achieving an accuracy of about 98% and 95% on the Shenzhen and Montgomery datasets. Xiaoyang Fu et al. [21] proposed an explainable transformer with a hybrid network structure (LungMaxViT) combining a CNN initial stage block with a Sqeeze-and-Excitation block to improve feature recognition for predicting Chest X-ray images in multi-class classification of lung diseases and achieving an accuracy of 96.8%. The proposed hybrid model showed its superiority in terms of accuracy and provided explainable identification results in terms of heat maps.

In addition to many achievements of deep learning across multiple domains, we have seen that due to computational complexities of the machine learning models and large-scale datasets, the performance of deep learning methods is reduced with single-computation approaches. Most of the research focuses on improving the accuracy of the classification models without focusing on the processing time. Meanwhile, processing time is also an important factor in the context of big data and applications in real-time systems with a reasonable response time. The challenge of the traditional deep learning models is that they consume a lot of time and require high-performance processing units. Therefore, the use of big data along with deep learning techniques is one of the most interesting areas of research, which can overcome these problems to improve health and medical science. This work proposes a parallel and distributed deep learning approach for lung disease detection and classification in a big data context. The main contributions of this paper are summarized as follows:

- Propose a distributed and parallel deep learning approach using Spark for classifying 14 types of lung diseases on chest X-ray images
- Provide ten scenarios to train CTXNet, Big Transfer (BiT), Swin Transformer, and two neural networks of the Faster RCNN and RetinaNet with three different backbones of the ResNet-50, CheXNet, and Inception ResNet V2 to detect and classify lung diseases. The deep learning models are run across multiple computing nodes
- Provide a comparison of the accuracy, Loss value, and processing time to find the most suitable scenario that can be used in reality.

2. Background

2.1. Big Data Processing

Big data and machine learning are now rapidly expanding in all science and engineering domains and changing all aspects of modern life in some way. A large amount of data has been continually generated at unprecedented and ever-increasing scales. Consequently, it brings tremendous challenges for us to address problems with higher memory and computation of large-scale datasets and machine learning

models. Thus, using methods that can reduce storage and computation problems with single computation approaches is important. To overcome these problems, a parallel and distributed approach can make the training time and computation drastically faster.

Apache Spark is an open-source MapReduce model framework highlighting speed, ease of use, and fault tolerance. Spark offers in-memory operations, robust, distributed, fault-tolerant data objects (namely RDD), and has become a more popular framework for machine learning and processing big data. Thus, it is a worthy choice for many applications with large data volumes and needing dependable performance. In this work, the input dataset from Chest X-ray images is stored in HDFS to support feature extraction and classification in the Spark environment. Spark is a better choice than Hadoop MapReduce since it processes files in memory instead of on disks. Thus, it is used in this study to enable fast and efficient distributed processing in a big data context.

2.2. Lesion detection and feature extraction using deep learning

2.2.1. Residual Neural Network (ResNet)

ResNet [22] is the most commonly used architecture since it has fewer parameters. The convolutional layers of the network have a 3x3 filter, and down-sampling is performed directly by the convolutional layers of stride 2. The network's last layer is a fully connected layer with two channels using ReLU activation and softmax activation functions. Shortcut connections are used in ResNet to overcome the problems of reducing accuracy and vanishing gradients that occur in deep neural networks.

2.2.2. CTXNet

The CTXNet [9] model includes a Convolutional Tokenization block that divides the input image into patches and converts them into a sequence of embeddings. This sequence of embeddings is then fed into a transformer encoder block with a self-attention mechanism and a feed-forward network to extract features. The output of the encoder block is processed by a sequence pooling layer to aggregate information from all patches. Finally, the output is passed through a fully connected layer with a softmax activation function to predict the probabilities of the disease classes. With this integrated and optimized architecture, CTXNet can effectively learn the discriminative features of lung diseases on both X-ray and CT images with high accuracy and short training time.

2.2.3. Swin Transformers

Swin Transformer [6] is a novel architecture recently proposed for computer vision tasks. Instead of computing self-attention globally like standard Transformer models, Swin Transformer performs it within local windows of size 7×7 and shifts the window positions across layers. This shifted window mechanism significantly reduces the computational complexity from quadratic to linear with respect to the input image size while still ensuring connections among image regions.

2.2.4. BigTransfer (BiT)

Big Transfer (BiT) [7] is a general and flexible visual representation learning method that utilizes ResNet-V2 architecture with varying depths and widths, where Group Normalization and Weight Standardization replace the Batch Normalization layers to improve training efficiency on large batches. The largest model, BiT-L, with a ResNet-152x4 architecture, is pre-trained on the JFT-300M dataset and achieves state-of-the-art results on many benchmarks, such as 87.54% top-1 accuracy on ImageNet ILSVRC-2012, 99.37% on CIFAR-10, and 93.51% on CIFAR-100.

2.2.5. CheXNet

CheXNet [23] is a 121-layer DenseNet trained on the chest X-rays dataset, which contains 112,120 frontal-view X-rays of 30,805 patients. The weights of CheXNet are initialized with a model pretrained on the ImageNet dataset. DenseNet [24] controls the amount of information to be added, improves information flow with direct connections, and makes optimization of very deep networks tractable. DenseNet has less than half the parameters of ResNet-50 but has higher accuracy when training on the

ImageNet dataset. In this study, the final layer of CheXNet is connected to a fully connected network with a binary output layer for classification.

2.2.6. Inception ResNet V2

Inception ResNet V2 [25] was developed from the Inception architectures [26] to take advantage of residual networks and improve the accuracy and convergence speed of the original model. Inception ResNet V2 is more exquisitely designed based on Inception V4. It utilizes residual connections to accelerate the training and improve performance. A complete Inception network consists of multiple Inception modules. The idea of the Inception module is that instead of using a Conv layer with a fixed kernel size, Inception uses multiple Conv layers simultaneously with different kernel sizes and concatenates the outputs.

2.3. Neural networks for classification

2.3.1. Faster R-CNN

Faster R-CNN [27] is one of the modern and effective methods for object detection and classification. It has gone through many versions, such as R-CNN, Fast R-CNN, and Faster R-CNN. Faster R-CNN is proposed to train a more efficient model, replacing the selective search algorithm, which is inherently slow. The feature is extracted from the input image and applied to a region-of-interest (ROI) to obtain the feature vector. The classification results are output through a softmax and a regression layer to predict the bounding boxes.

2.3.2. RetinaNet

RetinaNet [28] is considered to be quite good at detecting small objects. Cross-entropy is often used as a loss function in classification problems, which is weak in treating classes equally. The model will tend to be biased in favor of the main classes and miss the minority classes. This becomes worse when the minority classes are important classes, such as lung lesions. RetinaNet uses the balanced cross-entropy function, which assigns higher weights to the minority classes to penalize the model when it incorrectly predicts these classes strongly.

2.4. Evaluation metrics

2.4.1. Accuracy

Accuracy is calculated based on equation 1, where TP is the true positive; TN is the true negative; FP is the false positive; and FN is the false negative.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{1}$$

2.4.2. Loss Values

The Loss function of Faster R-CNN is determined by classification loss and localization loss as equations 3 and 4.

$$Loss(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i,p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i L_{reg}(t_i,t_i^*) \tag{2}$$

$$smooth_{L1} = \begin{cases} |x| & if\ |x| > \alpha \\ \frac{1}{|\alpha|}x^2 & otherwise \end{cases} \tag{3}$$

where i is the index of the anchor in mini-batch; p_i is the predicted probability of anchor i being an object; the ground-truth label value p_i^* is 1 if the anchor is positive, and 0 otherwise; ti is a 4-dimensional vector represents the coordinate values of the predicted bounding box; t_i^* is a 4- dimensional vector represents the coordinate values of the ground-truth box corresponding to the positive anchor; L_{cls} is the Multi-class cross-entropy loss of N classes; and L_{reg} is calculated based on equation 4 for

bounding box regression loss. RetinaNet uses the Focal Loss Function proposed on the basis of the Cross Entropy loss function by adding two parameters α and γ. The Focal Loss Function is presented in equation 5.

- α is used to represent the percentage of generated boxes containing background and foreground information to help balance the background and foreground when generating bounding boxes.
- γ represents the "concentration" on indistinguishable regions. The larger γ is, the smaller the loss value in the distinguishable region will be. The loss value of the indistinguishable region will contribute to the total loss value of the model.
- s_i is the model's estimated probability in range [0,1]

$$L = -\alpha_i(1 - s_i)^\gamma \log(s_i) \tag{4}$$

3. Method

We deploy a distributed and parallel processing model for feature extraction and lung lesion classification to improve processing time for large datasets. The general model consists of two phases, the training and testing phases, as illustrated in Fig. 1. The details of the proposed method are described as follows.

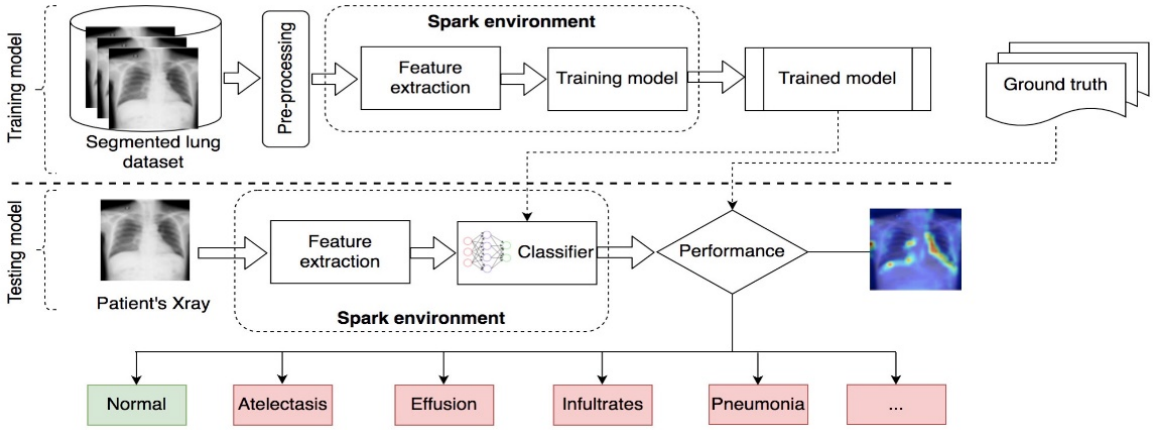


Fig. 1. General model of the proposed method

3.1. Training phase

3.1.1. Pre-processing

Various image enhancement techniques are applied to the input X-ray images, such as Gabor filter, Local binary Pattern, Histogram Equalization, Adaptive Histogram Equalization (AHE), etc. In this stage, noise reduction is accomplished by using a filter like the median filter. The AHE [29] technique is used to enhance the contrast of the image and performs well in CNN-based feature extraction. The images are resized to 256x256 to train as per the pretrained models' requirements. Then, we normalize the gray-scale images to the range [0,1] to match the input type of the models. We also apply two techniques of random cropping and flipping images to have data augmentation for the training dataset.

3.1.2. Distributed and parallel data processing with Spark for extraction and training models

Feature extraction and classification in large-scale datasets are time-consuming tasks for lung lesion detection and classification systems. To overcome this problem, we propose the extraction and classification of lung diseases in the MapReduce distributed and parallel processing model with the flexibility to persist data records, either in memory, on disk, or both. The MapReduce processing model for feature extraction and classification consists of map and reduce functions as presented in Fig. 2. A large amount of input data is split into different partitions on a distributed file system. The partition processing is distributed efficiently for execution on multiple computing nodes. A map task is mapping

an input image into a feature vector and a reduce task combines all feature vectors of the same key to produce a classification result. The output of this process is a trained model for lung lesion detection and classification.

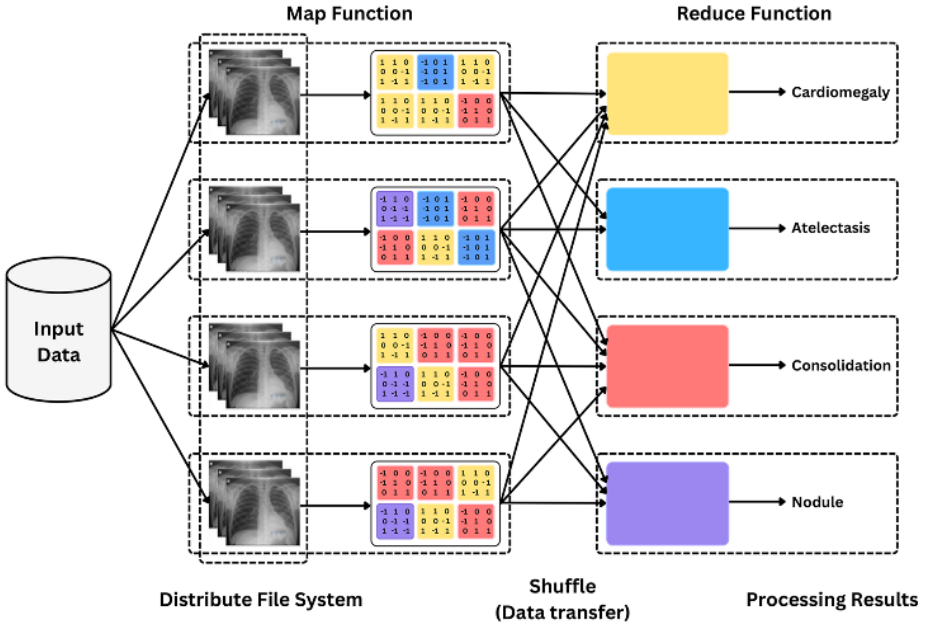


Fig. 2. MapReduce processing model

In this study, we deploy a distributed deep learning model on Spark for lung disease classification in a big data context. It supports iterative processes and is suitable for machine learning and optimization algorithms. This not only saves processing time but also reduces the training and identification time for deep learning models. Fig. 3 describes the distributed and parallel model to train and identify lung lesions and diseases. It shows a Spark cluster consisting of a manager (master) and a number of workers (slaves). The master node running the Spark job is responsible for scheduling, assigning, distributing, and monitoring tasks to worker nodes, which run the actual Spark tasks.

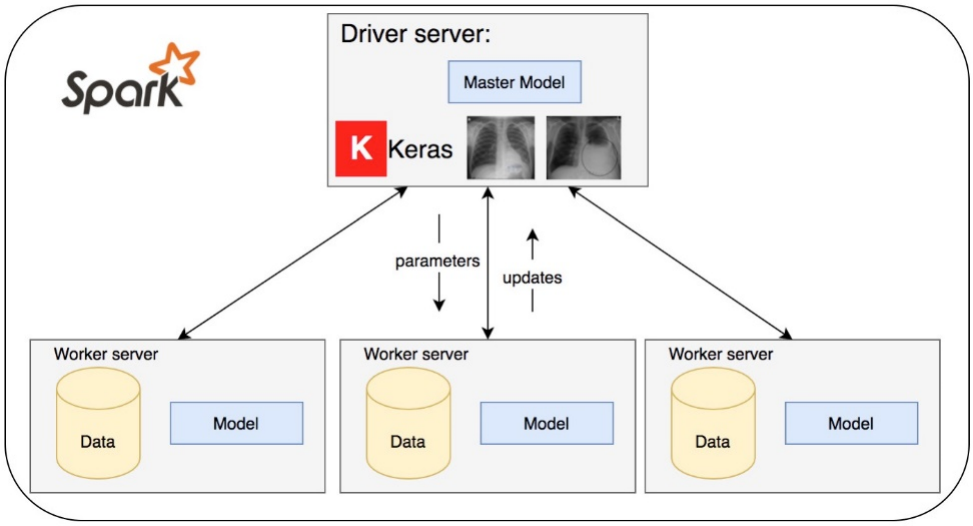


Fig. 3. Distributed and parallel model on a Spark cluster for extraction and training

The master node splits the input dataset into sub-datasets and then feeds them to the workers to handle parallel and distributed feature extraction. The workers execute tasks and forward the status of the tasks to the master. As a result, the features of lung lesions are extracted from chest x-ray images and temporarily saved in memory serving the further process of model training. In parallel training, the

master is responsible for computing average weights to provide a global average parameter (W) of network parameters, meanwhile, the remaining workers are responsible for training. Each worker obtains the local weights Wi corresponding to its local weights of the network to send updates to the master node. The same weights W are distributed to all workers when the averaging is executed. After training is complete, the training model with a global average parameter (W) is stored in HDFS on each worker to identify the lung lesions and diseases.

Due to the limitation of the experimental dataset and the advantages of deep neural networks, we apply the transfer learning technique to re-use the parameters. This helps to learn new features faster, shortens the training time, and does not require large datasets. The input images are extracted features to feed into the pre-trained network models on the workers for training. The model parameters are updated from different workers to the driver program on the master. The training process is repeated with a number of learning steps until the Loss value is not improved (not reduced) and the classification accuracy does not increase. We stop the training phase and implement the testing phase.

3.2. Testing phase

Lung disease features are extracted from the test images and then passed through the trained models to identify lung diseases. In this phase, we use a parallel and distributed model for extraction and classification. The master node is responsible for splitting the test image dataset into batches and inputting them to the trained models with the global average parameter W on workers to extract features and classify. The output of this process includes bounding boxes of the lung lesions and the type labels of the lung diseases.

To accurately identify the area of severe lung injury, we use GradCAM (Gradient-weighted Class Activation Mapping) [30] heat map (Fig. 4). We use the gradient information of the final conv layer to generate a rough map of the important regions in the image. Grad-CAM is a strict generalization of the Activation Mapping layers. It does not require retraining and is widely applicable to any CNN based architecture. To generate the heat maps, in this study, the Convolutional Block Attention Module (CBAM) and ResNet integrated network [31] are used to refine the output features of each residual block in ResNet.

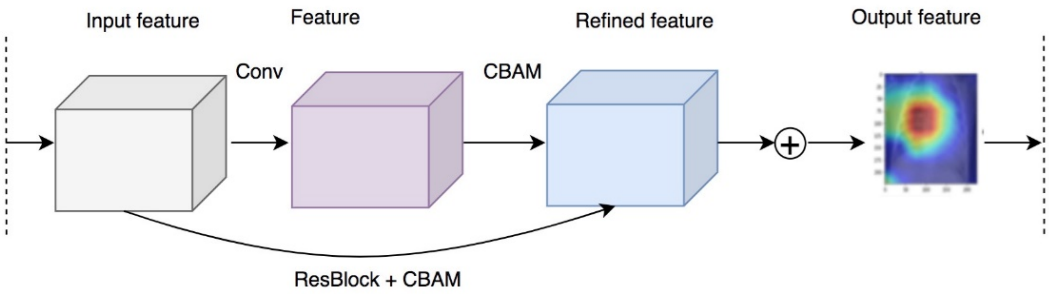


Fig. 4. Creating GradCam for output images

4. Results and Discussion

4.1. Experiments

4.1.1. Dataset and installation environment

The experiments are conducted on the public Kaggle NIH Chest X-ray dataset [32]. This dataset includes 112,120 disease-labeled X-ray images from 30,805 patients. The labels are expected to be over 90% accurate, suitable for weakly supervised learning. We divide 89,600 images used as the training set and 22,400 images as the testing set, corresponding to the ratio of 80:20. There are 15 classes, including 14 types of diseases and a class of "No findings". The images can be classified as "No findings" or a disease as listed in Fig. 5. The input images are 600×600px.

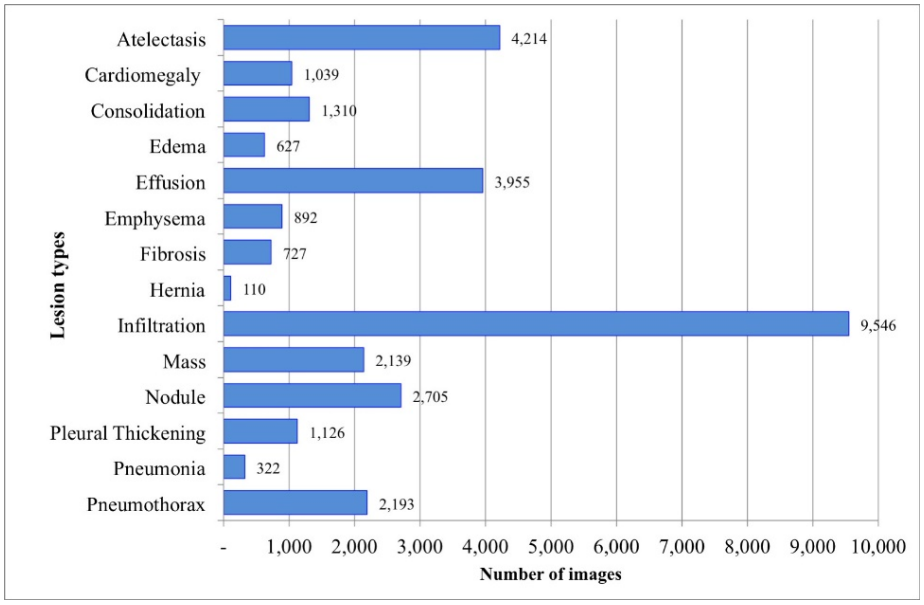


Fig. 5. Input dataset description

The experiments are conducted in two types of environments: a single computing environment and a parallel computing environment.

- Env1: Single computing environment: The computer's configuration is 16GB RAM and Nvidia Tesla P100.
- Env2: Parallel computing environment: We store the training model and data on multiple machines and cores. The processing is divided on an Apache Spark cluster, including a master (2 cores and 4GB RAM) and 8 workers (each has 2 cores and 8GB RAM). The library used to train the models is BigDL 0.12

4.1.2. Scenarios and Parameters

We propose ten scenarios with the training network models and parameters as described in Table 1. All scenarios have the same learning rate of 0.0001, mAP@IoU of 0.5, and 20 training steps. The number of classes is 15, corresponding to 14 types of lung lesions and "No findings".

Table 1. Proposed scenarios and training parameters

Scenario	Environment	Classification	Feature extraction
1.	Env1	Faster R-CNN	Inception ResNet-V2
2.	Env1	Faster R-CNN	CheXNet
3.	Env1	Faster R-CNN	ResNet-50
4.	Env1	RetinaNet	ResNet-50
5.	Env1	CTXNet	Transformer encoder
6.	Env1	Big Transfer (BiT)	ResNet + Group Normalization
7.	Env1	Swin Transformers	Self-attention
8.	Env2	Faster R-CNN	Inception ResNet-V2
9.	Env2	Faster R-CNN	CheXNet
10.	Env2	Faster R-CNN	ResNet-50

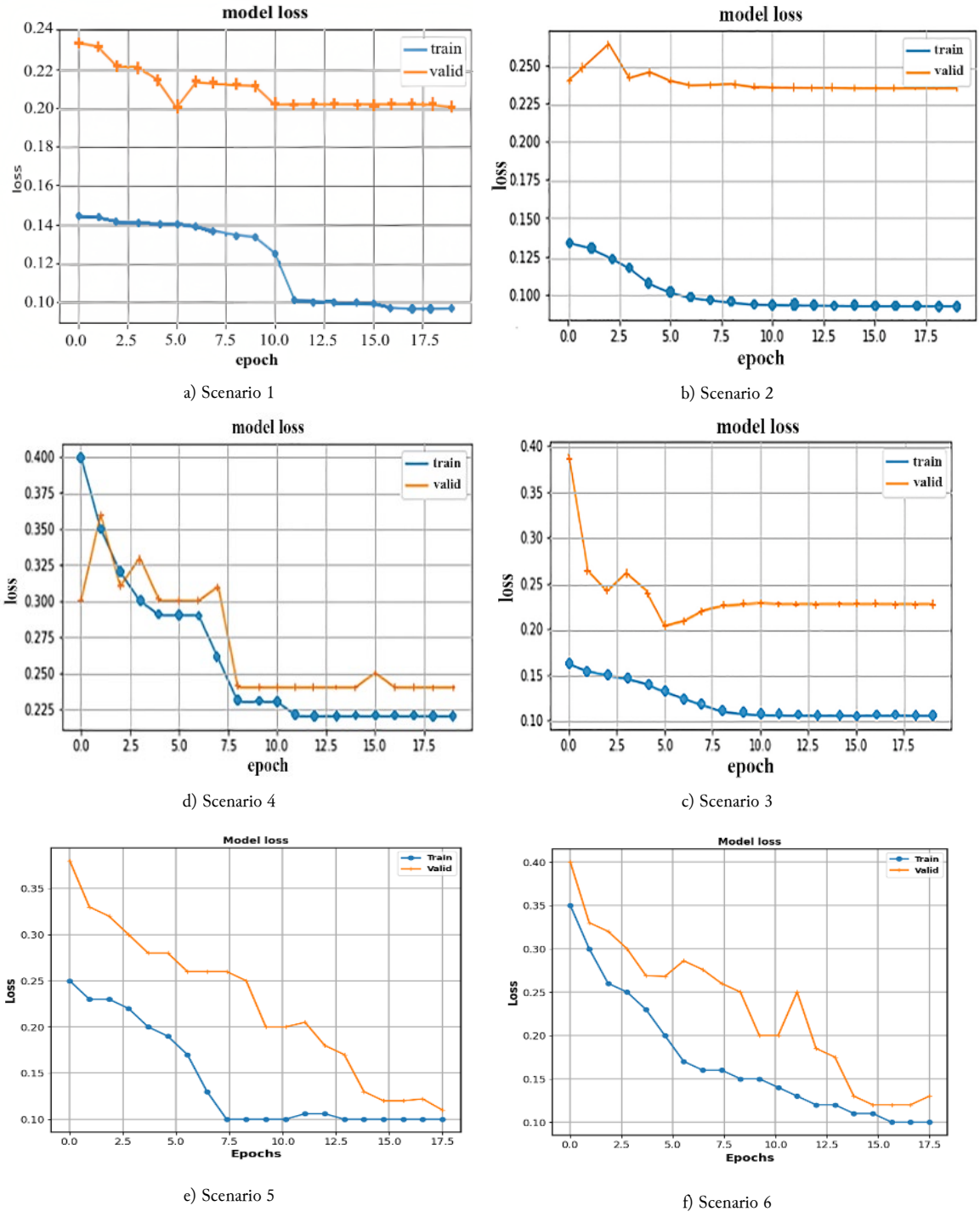
4.2. Results

4.2.1. Training results

In the training phase, we find the optimal weights of the scenarios by calculating the lowest Loss values to make a decision to stop training.

4.2.1.1. Loss value

Fig. 6 and Fig. 7 show the loss values of ten scenarios in a single computing approach and a distributed processing environment. The loss value tends to decrease rapidly at the beginning of the training process. It significantly and evenly decreases in the following training steps, showing that the features are learned effectively after 20 epochs. We stop the training process when the loss value no longer decreases after 20 epochs. The average loss values of scenarios 1, 2, 3, and 5 are 0.097, 0.096, 0.112, and 0.12, respectively.



a) Scenario 1 b) Scenario 2

d) Scenario 4 c) Scenario 3

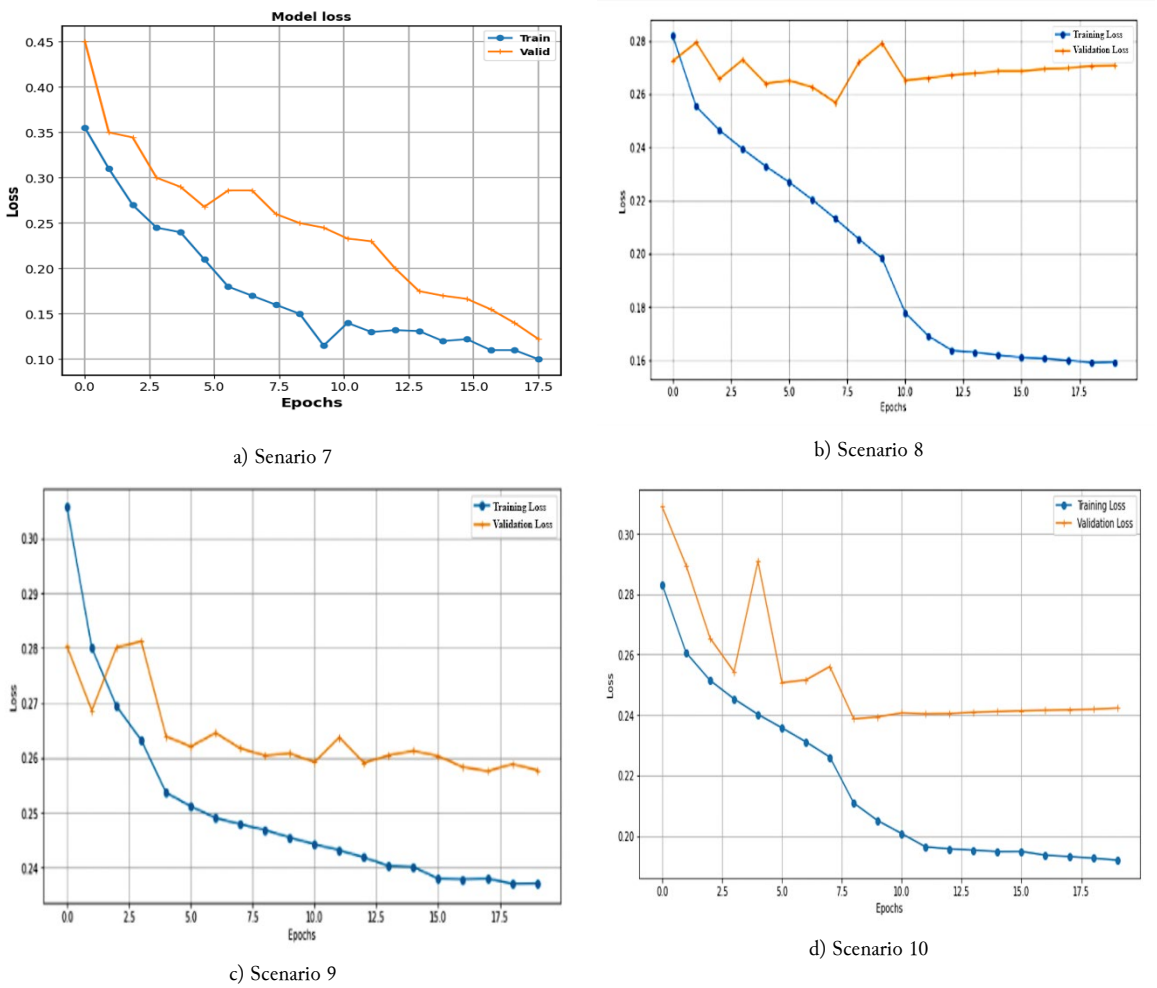
e) Scenario 5 f) Scenario 6

Fig. 6. Loss values of scenarios 1, 2, 3, 4, 5, and 6

The average loss value in scenario 4 is up to 0.24, which is quite high compared to scenarios 1-3. As can be seen, scenario 2 with Faster R-CNN CheXNet has the lowest loss value, and scenario 4 with

RetinaNet ResNet-50 has the highest loss value among the first four scenarios. Among the three models with Transformer architecture in scenarios 5, 6, and 7, the loss values are 0.095, 0.097, and 0.096, respectively. The results show that all three models achieve good performance with very low and nearly equivalent loss values, demonstrating the power of the Transformer architecture in learning features on medical images.

Fig. 7 shows the loss values of scenarios 8, 9, and 10 with 20 epochs of training for a parallel computing approach. The loss values are taken when training the network models on a Spark cluster with workers. As the number of workers increases, the loss value also increases. The loss values of scenarios 8, 9, and 10 when running on a cluster of four workers are 0.176, 0.185, and 0.177, respectively, and on a cluster of 8 workers are 0.191, 0.228, and 0.212, respectively. Consequently, scenario 8 has the lowest loss value with an average loss of 0.16 compared to scenarios 9 and 10.



a) Senario 7

b) Scenario 8

c) Scenario 9

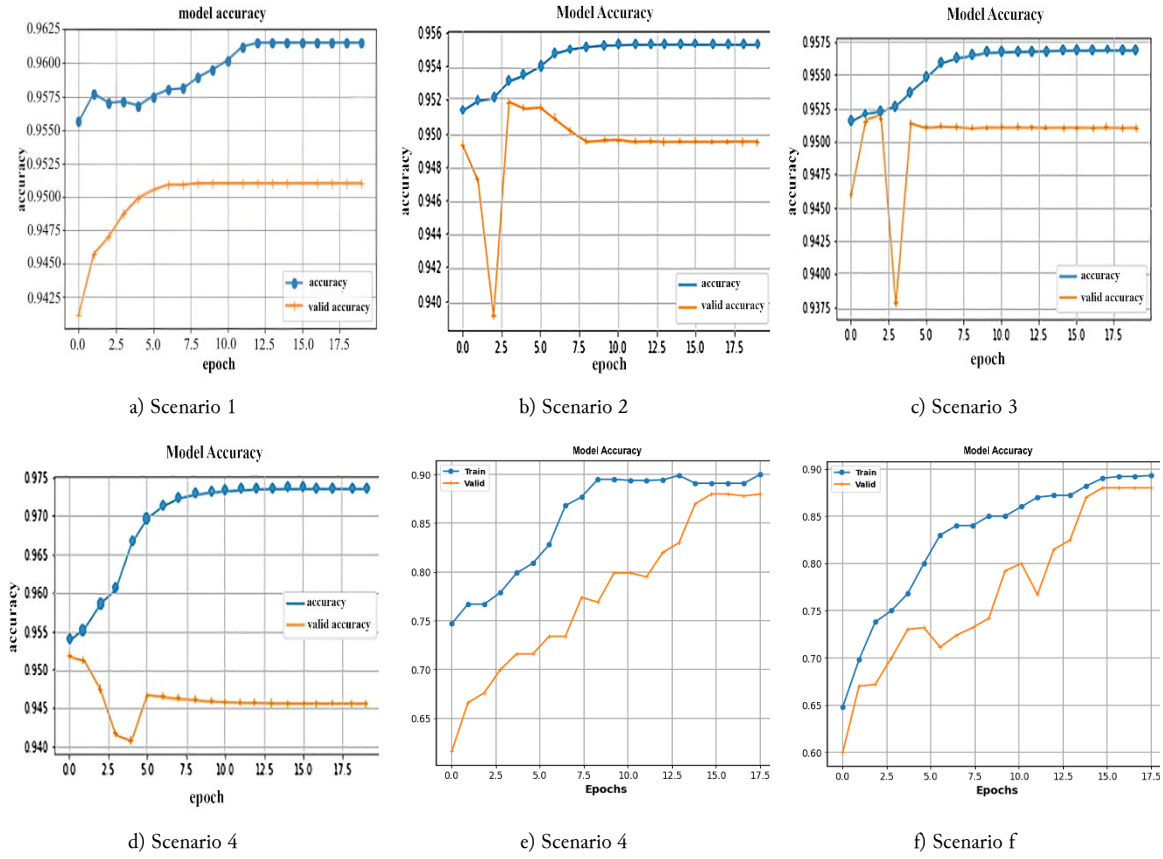
d) Scenario 10

Fig. 7. Loss value of scenarios 7 on a single computing environment and scenarios 8, 9, and 10 on a Spark cluster of 2 workers

Increasing the number of workers for parallel computation causes the loss values of scenarios 8-10 to increase compared to scenarios 1-7 in single computation, which means that the false predictions increase. Using the same network models, Faster R-CNN, Inception ResNet V2, but running in different environments, scenario 6 shows the ability of the parallel computation compared to scenario 1 with single computation. In a big data context, the input dataset is divided and fed into the workers to train the neural networks with a large activation function consuming a lot of memory. It is inappropriate for deep learning to train on a single machine with low memory capacity causing congestion, which increases training time.

4.2.1.2. Accuracy

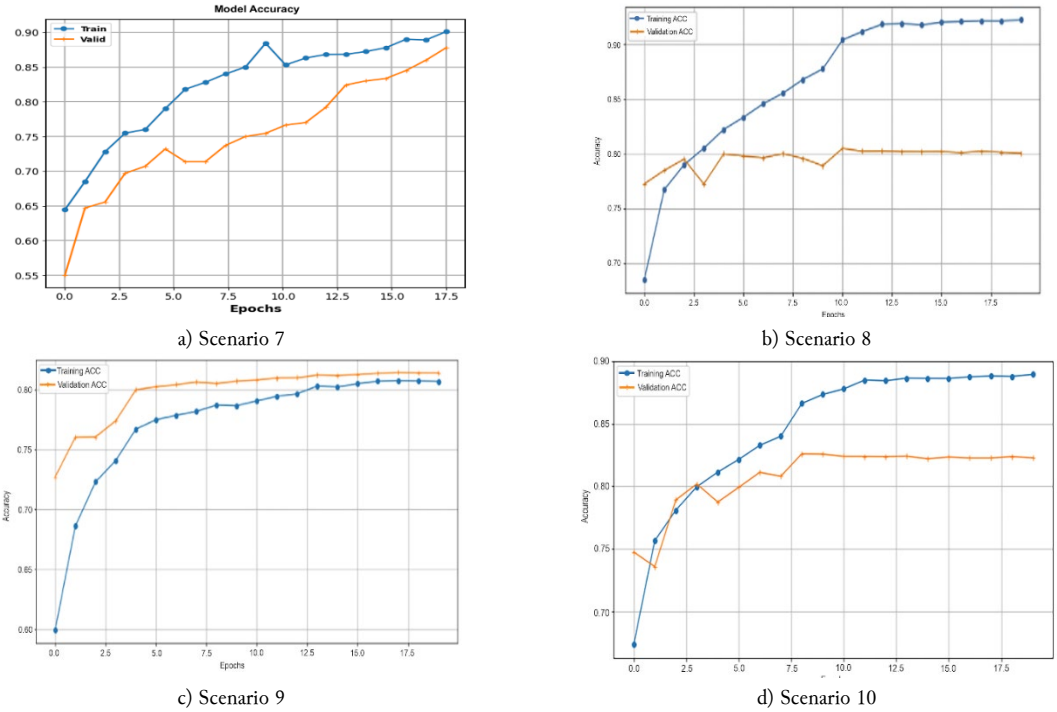
Fig. 8 and Fig. 9 show the accuracy of Faster R-CNN and RetinaNet in the first seven scenarios. As can be seen, the seven models give quite similar and stable results at above 94%.



a) Scenario 1 b) Scenario 2 c) Scenario 3

d) Scenario 4 e) Scenario 4 f) Scenario f

Fig. 8. Accuracy of scenarios 1, 2, 3, 4, 5, and 6



a) Scenario 7 b) Scenario 8

c) Scenario 9 d) Scenario 10

Fig. 9. Accuracy of scenarios 7, 8, 9, and 10

Fig. 9 shows the accuracy of scenarios 8, 9, and 10 on a Spark cluster of 2 workers. Table 2 shows the average accuracy of 10 scenarios. The network models in scenarios 1, 6, and 7 give better results compared to the models in scenarios 2, 3, 4, and 5. Although RetinaNet in scenario 4 does not improve accuracy, it is suitable for shortening training time and detecting smaller objects. In a parallel processing environment, scenario 8 gives the highest accuracy above 90%. In general, increasing the number of workers causes a decrease in accuracy.

Table 2. Average accuracy of 10 scenarios

Scenario	Environment	No. Workers	Accuracy (%)
1.	Env1	01	96.05
2.	Env1	01	95.2
3.	Env1	01	95.5
4.	Env1	01	94.7
5.	Env1	01	95.77
6.	Env1	01	96.03
7.	Env1	01	96.05
8.	Env2	02	92.7
		04	91.3
		08	90
9.	Env2	02	89.5
		04	83
		08	81.3
10.	Env2	02	89.5
		04	87.8
		08	87.5

4.2.1.3. Training time

Fig. 10 shows the model's training time in 10 scenarios. Faster R-CNN Inception ResNet V2 in scenario 1 provides the shortest training time compared to the remaining models in scenarios 2, 3, and 4. The longest training time is 7.1 hours of Faster R-CNN ResNet-50. RetinaNet in scenario 4 gives faster training time compared to Faster R-CNN when combined with ResNet-50. In the parallel processing environment, the training time is more than 2 times faster than the training time in a single computing environment. The training time does not decrease significantly when we increase the number of workers, since the master has to aggregate data on many workers, increasing the waiting time. Thus, choosing the appropriate number of workers depends on the dataset and computer configuration.

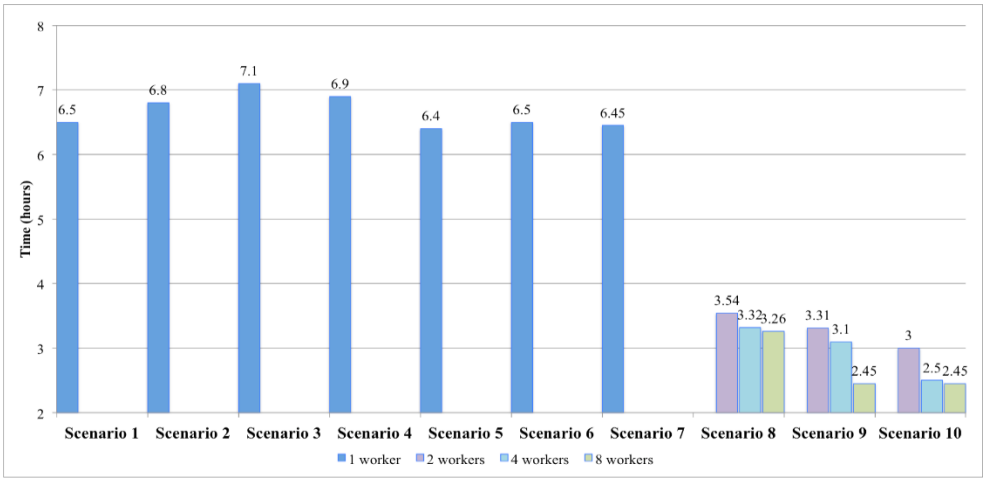


Fig. 10. Training time of the 10 scenarios

The training results indicate that the experimental models trained in a distributed environment have significantly accelerated the processing and training phases, greatly reducing the required time compared to a local setup. However, this approach introduces a certain degree of accuracy variation between the two environments, as shown in Table 2. Nevertheless, the discrepancy remains within an acceptable

range, ensuring the overall effectiveness of the model. This aligns with real-world challenges, where the increasing volume of medical data necessitates systems capable of rapid analysis and efficient large-scale data processing while maintaining flexibility for future scalability and expansion. Additionally, we will continue to optimize the experimental models in the distributed environment to minimize misclassification cases while enhancing overall performance in terms of accuracy and training speed in our future work.

4.2.2. Testing results

4.2.2.1. Average accuracy and prediction time

We evaluate scenarios 1-10 by calculating the accuracy and run-time of the detection and classification for the testing dataset, as illustrated in Fig. 11 and Fig. 12. Fig. 11 represents the average prediction accuracy of the scenarios. The accuracy of 10 scenarios ranges from 81.4% to 96.7%. Scenarios 1 and 7 give the results of classification with the highest accuracy, up to 96.7%, compared to the remaining scenarios. We can see that the classification accuracy of scenarios 8-10 decreases when the number of workers increases because the training dataset is subdivided into batches to train on workers; thus, the network models are trained and fine-tuned on the workers with sub-training datasets. As a result, the global weights of the training model are the average of the local weights on workers.

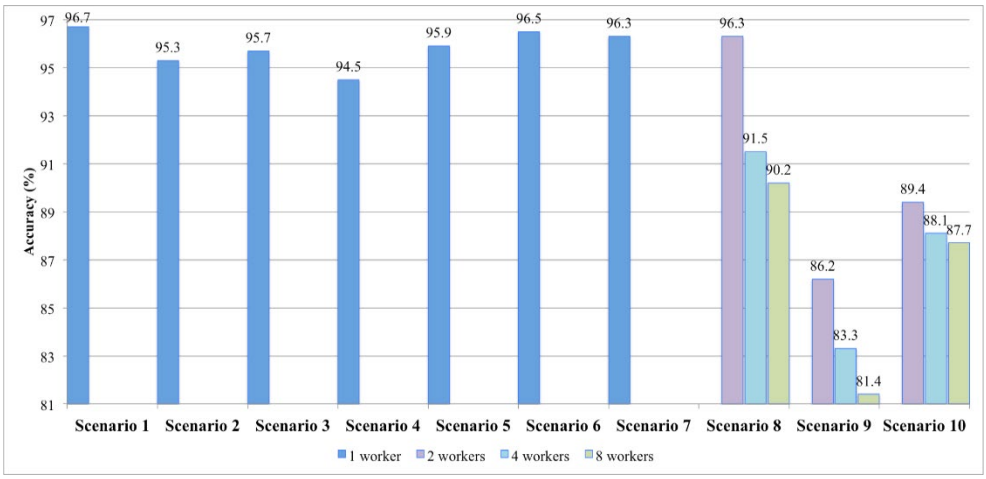


Fig. 11. Average prediction accuracy of 10 scenarios for the testing phase

Nevertheless, the decrease in accuracy is negligible, while the processing time for prediction decreases rapidly, as shown in Fig. 12.

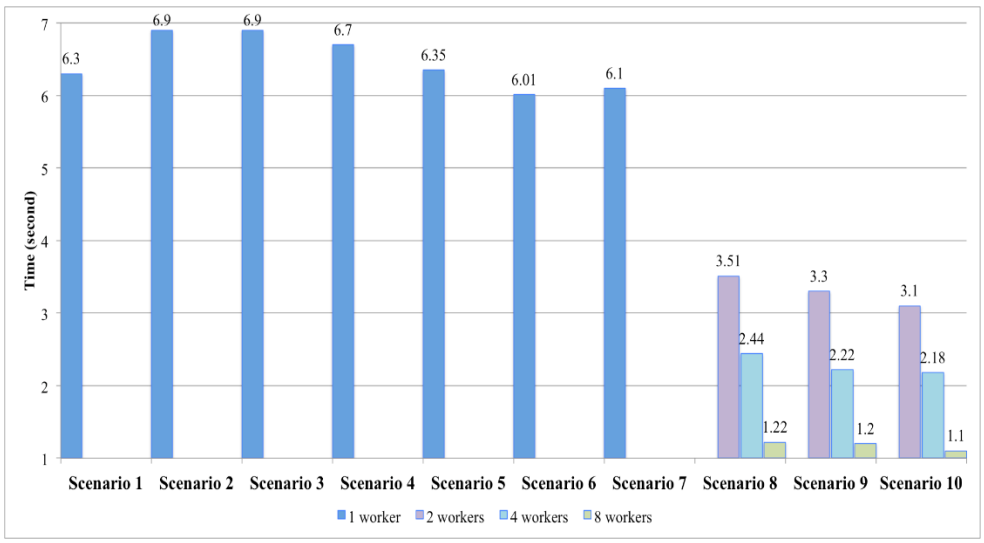


Fig. 12. Execution time of 10 scenarios for the testing phase

The execution time of scenarios 8-10 in Spark is shortened by approximately 50% compared with the execution time of scenarios 1-7. This is the price to trade off when the processing speed is fast, the accuracy decreases, and vice versa. However, it is suitable in today's increasingly large data context for early diagnosis and timely treatment because a key requirement for lung disease detection is that the solutions work in real-time or near real-time.

4.2.2.2. GradCam heat map

We use the GradCam heat map tool to determine the most severe area of the lesion. This method uses gradients to calculate the significance of the spatial positions in the convolutional layers. Grad-CAM results clearly show central regions. The location of the most severe lung lesion is shown with colors. The colors on the GradCam heat map represent the severity of the injury in order from left to right. Some illustration results of lung lesion regions are shown in Fig. 13.

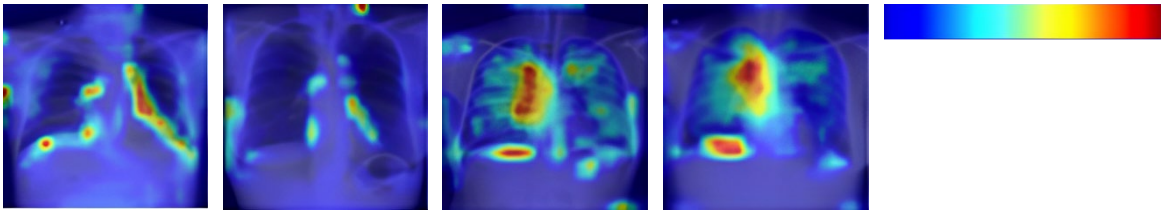


Fig. 13. GradCam heat map on X-rays

4.2.3. Comparison and Discussion

Fig. 14 presents a comparison of the proposed method and recent related works in terms of accuracy. While existing research has achieved accuracy rates of up to 99%, the proposed approach attains approximately 96% accuracy during the training phase. However, our results indicate that training in a distributed environment significantly reduces processing time, making model training more efficient. This comes at the cost of a slight reduction in accuracy compared to a local environment. In future work, we aim to optimize the distributed training process further to mitigate this accuracy gap while maintaining computational efficiency.

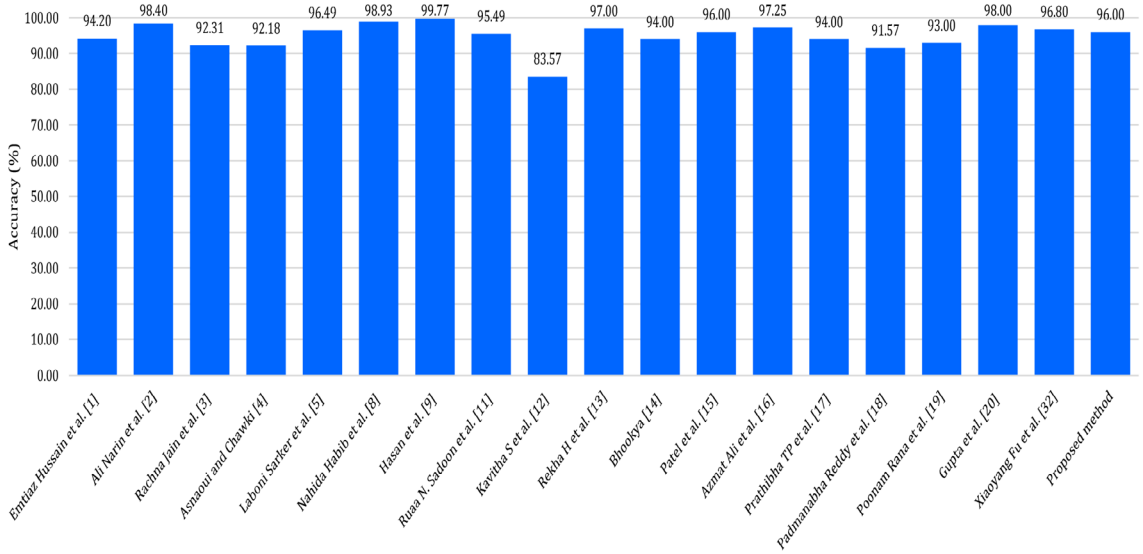


Fig. 14. Comparing the accuracy with recent related works

In this work, we also conducted lung cancer classification using state-of-the-art classification architectures, including CTXNet, BiT, and Swin Transformer, corresponding to scenarios 5, 6, and 7. These experimental models achieved high accuracy during training while requiring less training time compared to other scenarios in the local environment. In future work, we will proceed with

implementing and deploying scenarios 5, 6, and 7 in a distributed environment to evaluate their performance in terms of accuracy and training time compared to the local environment. In addition, small lesions that are difficult to detect, such as solitary pulmonary nodules, are easily missed, but RetinaNet gives outstanding detection results. This model can detect small lesions thanks to the Focal Loss function.

5. Conclusion

This research focuses on developing a prediction model using deep learning methods and Apache Spark architecture for diagnosing lung diseases. We focus on designing ten scenarios to train two Faster R-CNN and RetinaNet neural networks with three backbones of the ResNet-50, CheXNet, and Inception ResNet-V2 in two environments of single and parallel computation. Moreover, we also experimented with three new network architectures on the medical image dataset: CTXNet, Big Transfer (BiT), and Swin Transformer. The proposed method with the first seven scenarios extracts and classifies features of lung diseases using deep learning with a single computing approach. The three remaining scenarios of the proposed method are deployed by a parallel and distributed approach on Spark, with the number of worker nodes increasing from 2 to 8. The experimental results show that the Faster R-CNN Inception ResNet-V2 model gives the highest result, 96.05% in a single computing environment and 92.7% in a parallel computing environment. Additionally, the Swin Transformers model gives a similar accuracy of 96.05% on a single computer. We compare the accuracy, Loss value, and computational time for the proposed scenarios to find the most promising scenarios that can be used to detect and classify lung diseases on chest X-rays. Moreover, we use the Grad-CAM tool to highlight the lung lesions, thus radiologists can see the location and size of the lesions without much effort. The experimental results show that using big data combined with deep learning for training and prediction is more effective than traditional deep learning, as both deep learning and big data are rapidly growing fields. In further research, we will make a comparative study by applying some deep learning methods along with Spark architecture on other well-developed datasets to find the most promising models that can be used for early diagnosis of lung disease on chest X-rays.

Declarations

Author contribution. . All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.
Funding statement. None of the authors has received any funding or grants from any institution or funding body for the research.
Conflict of interest. The authors declare no conflict of interest.
Additional information. No additional information is available for this paper.

References

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

[2] A. Narin, C. Kaya, and Z. Pamuk, "Automatic detection of coronavirus disease (COVID-19) using X-ray images and deep convolutional neural networks," *Pattern Anal. Appl.*, vol. 24, no. 3, pp. 1207–1220, Aug. 2021, doi: 10.1007/s10044-021-00984-y.

[3] R. Jain, P. Nagrath, G. Kataria, V. Sirish Kaushik, and D. Jude Hemanth, "Pneumonia detection in chest X-ray images using convolutional neural networks and transfer learning," *Measurement*, vol. 165, p. 108046, Dec. 2020, doi: 10.1016/j.measurement.2020.108046.

[4] K. El Asnaoui and Y. Chawki, "Using X-ray images and deep learning for automated detection of coronavirus disease," *J. Biomol. Struct. Dyn.*, vol. 39, no. 10, pp. 3615–3626, Jul. 2021, doi: 10.1080/07391102.2020.1767212.

[5] L. Sarker, M. M. Islam, T. Hannan, and Z. Ahmed, "COVID-DenseNet: A Deep Learning Architecture to Detect COVID-19 from Chest Radiology Images," *Preprints*, p. 5, May 2020, doi: 10.20944/preprints202005.0151.v1.

[6] Z. Liu *et al.*, "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2021, pp. 9992–10002, doi: 10.1109/ICCV48922.2021.00986.

[7] A. Kolesnikov *et al.*, "Big Transfer (BiT): General Visual Representation Learning," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12350 LNCS, Springer, Cham, 2020, pp. 491–507, doi: 10.1007/978-3-030-58558-7_29.

[8] N. Habib, M. M. Hasan, M. M. Reza, and M. M. Rahman, "Ensemble of CheXNet and VGG-19 Feature Extractor with Random Forest Classifier for Pediatric Pneumonia Detection," *SN Comput. Sci.*, vol. 1, no. 6, p. 359, Nov. 2020, doi: 10.1007/s42979-020-00373-y.

[9] M. Z. Hasan *et al.*, "Fast and Efficient Lung Abnormality Identification With Explainable AI: A Comprehensive Framework for Chest CT Scan and X-Ray Images," *IEEE Access*, vol. 12, pp. 31117–31135, 2024, doi: 10.1109/ACCESS.2024.3369900.

[10] E. E. Khaing and T. Z. Aung, "Lung Disease Classification from Chest X-ray Images Using Convolutional Neural Network and Long Short-Term Memory Model," in *2024 3rd International Conference on Artificial Intelligence For Internet of Things (AIIoT)*, May 2024, pp. 1–6, doi: 10.1109/AIIoT58432.2024.10574623.

[11] R. N. Sadoon and A. M. Chaid, "Deep Learning Technique for The Classification of Lung Diseases from X-ray Images," *J. Al-Qadisiyah Comput. Sci. Math.*, vol. 16, no. 2, pp. 70 83 70–83, Jun. 2024, doi: 10.29304/jqcsm.2024.16.21544.

[12] K. S, R. S. Shudapreyaa, P. Prakash, V. S, V. V, and Y. S, "Classification of Lung Diseases Using Transfer Learning with Chest X-Ray Images," in *2024 Second International Conference on Emerging Trends in Information Technology and Engineering (ICETITE)*, Feb. 2024, pp. 1–6, doi: 10.1109/ic-ETITE58242.2024.10493367.

[13] R. H, T. Kumaravel, P. Natesan, B. B M, S. Sangeetha, and S. Dharanesh, "Comparative Study of Deep Learning Techniques for Automated Classification of Lung Diseases," in *2023 4th International Conference on Smart Electronics and Communication (ICOSEC)*, Sep. 2023, pp. 1324–1328, doi: 10.1109/ICOSEC58147.2023.10276053.

[14] J. Bhookya, "Examine Lung Disorders and Disease Classification Using Advanced CNN Approach," in *2023 3rd Asian Conference on Innovation in Technology (ASIANCON)*, Aug. 2023, pp. 1–6, doi: 10.1109/ASIANCON58793.2023.10270338.

[15] A. N. Patel *et al.*, "An explainable transfer learning framework for multi-classification of lung diseases in chest X-rays," *Alexandria Eng. J.*, vol. 98, pp. 328–343, Jul. 2024, doi: 10.1016/j.aej.2024.04.072.

[16] A. Ali, Y. Wang, and X. Shi, "Detection of multi-class lung diseases based on customized neural network," *Comput. Intell.*, vol. 40, no. 2, p. e12649, Apr. 2024, doi: 10.1111/coin.12649.

[17] P. T. P and P. M. Arabi, "Computer Aided Classification of Lung Cancer, Ground Glass Lung and Pulmonary Fibrosis Using Machine Learning and KNN Classifier," *Int. J. Adv. Comput. Sci. Appl.*, vol. 15, no. 7, pp. 1145–1151, Jun. 2024, doi: 10.14569/IJACSA.2024.01507111.

[18] Y. C. A. P. Reddy, S. L. Vemuri, T. P. Mahan, V. S. Teja, and V. Anil, "A Semi-Supervised Deep Learning Approach for Detection and Classification of Lung Diseases," in *2023 IEEE Silchar Subsection Conference (SILCON)*, Nov. 2023, pp. 1–6, doi: 10.1109/SILCON59133.2023.10404180.

[19] P. Rana, V. Sharma, and P. Kumar Gupta, "Lung Disease Classification using Dense Alex Net Framework with Contrast Normalisation and Five-Fold Geometric Transformation," *Int. J. Recent Innov. Trends Comput. Commun.*, vol. 11, no. 2, pp. 94–105, Mar. 2023, doi: 10.17762/ijritcc.v11i2.6133.

[20] A. Gupta and A. Kumar, "Deep-Learning Based Hybrid Model For The Classification of Lung Diseases," in *2022 4th International Conference on Artificial Intelligence and Speech Technology (AIST)*, Dec. 2022, pp. 1–4, doi: 10.1109/AIST55798.2022.10065198.

[21] X. Fu, R. Lin, W. Du, A. Tavares, and Y. Liang, "Explainable hybrid transformer for multi-classification of lung disease using chest X-rays," *Sci. Rep.*, vol. 15, no. 1, p. 6650, Feb. 2025, doi: 10.1038/s41598-025-90607-x.

[22] B. Koonce, "ResNet 50," in *Convolutional Neural Networks with Swift for Tensorflow*, Berkeley, CA: Apress, 2021, pp. 63–72, doi: 10.1007/978-1-4842-6168-2_6.

[23] A. S. Al-Waisy *et al.*, "RETRACTED ARTICLE: COVID-CheXNet: hybrid deep learning framework for identifying COVID-19 virus in chest X-rays images," *Soft Comput.*, vol. 27, no. 5, pp. 2657–2672, Mar. 2023, doi: 10.1007/s00500-020-05424-3.

[24] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, vol. 2017-Janua, pp. 2261–2269, doi: 10.1109/CVPR.2017.243.

[25] C. Peng, Y. Liu, X. Yuan, and Q. Chen, "Research of image recognition method based on enhanced inception-ResNet-V2," *Multimed. Tools Appl.*, vol. 81, no. 24, pp. 34345–34365, Oct. 2022, doi: 10.1007/s11042-022-12387-0.

[26] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, vol. 2016-Decem, pp. 2818–2826, doi: 10.1109/CVPR.2016.308.

[27] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017, doi: 10.1109/TPAMI.2016.2577031.

[28] L. Tan, T. Huangfu, L. Wu, and W. Chen, "Comparison of RetinaNet, SSD, and YOLO v3 for real-time pill identification," *BMC Med. Inform. Decis. Mak.*, vol. 21, no. 1, p. 324, Dec. 2021, doi: 10.1186/s12911-021-01691-8.

[29] S. M. Pizer *et al.*, "Adaptive histogram equalization and its variations," *Comput. Vision, Graph. Image Process.*, vol. 39, no. 3, pp. 355–368, Sep. 1987, doi: 10.1016/S0734-189X(87)80186-X.

[30] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 336–359, Feb. 2020, doi: 10.1007/s11263-019-01228-7.

[31] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional Block Attention Module," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11211 LNCS, Springer Verlag, 2018, pp. 3–19, doi: 10.1007/978-3-030-01234-2_1.

[32] "NIH Chest X-rays," *Kaggle*, 2018. [Online]. Available at: https://www.kaggle.com/datasets/nih-chest-xrays/data.