

Article

A Novel Deep Learning Framework Enhanced by Hybrid Optimization Using Dung Beetle and Fick's Law for Superior Pneumonia Detection

Abdulazeez M. Sabaawi ^{1,*}  and Hakan Koyuncu ² ¹ Information Technologies Department, Altinbas University, Istanbul 34217, Turkey² Computer Engineering Department, Altinbas University, Istanbul 34217, Turkey; hakan.koyuncu@altinbas.edu.tr

* Correspondence: 213721015@ogr.altinbas.edu.tr

Abstract: Pneumonia is an inflammation of lung tissue caused by various infectious microorganisms and noninfectious factors. It affects people of all ages, but vulnerable age groups are more susceptible. Imaging techniques, such as chest X-rays (CXR), are crucial in early detection and prompt action. CXRs for this condition are characterized by radiopaque appearances or sometimes a consolidation in the affected part of the lung caused by inflammatory secretions that replace the air in the infected alveoli. Accurate early detection of pneumonia is essential to avoid its potentially fatal consequences, particularly in children and the elderly. This paper proposes an enhanced framework based on convolutional neural network (CNN) architecture, specifically utilizing a transfer-learning-based architecture (MobileNet V1), which has outperformed recent models. The proposed framework is improved using a hybrid method combining the operation of two optimization algorithms: the dung beetle optimizer (DBO), which enhances exploration by mimicking dung beetles' navigational strategies, and Fick's law algorithm (FLA), which improves exploitation by guiding solutions toward optimal areas. This hybrid optimization effectively balances exploration and exploitation, significantly enhancing model performance. The model was trained on 7750 chest X-ray images. The framework can distinguish between healthy and pneumonia, achieving an accuracy of $98.19 \pm 0.94\%$ and a sensitivity of $98 \pm 0.99\%$. The results are promising, indicating that this new framework could be used for the early detection of pneumonia with a low cost and high accuracy, especially in remote areas that lack expertise in radiology, thus reducing the mortality rate caused by pneumonia.

Keywords: deep learning; X-ray image classification; machine learning in healthcare; convolutional neural networks; optimization algorithms



Citation: Sabaawi, A.M.; Koyuncu, H. A Novel Deep Learning Framework Enhanced by Hybrid Optimization Using Dung Beetle and Fick's Law for Superior Pneumonia Detection. *Electronics* **2024**, *13*, 4042. <https://doi.org/10.3390/electronics13204042>

Academic Editors: Yuji Iwahori,
Haibin Wu and Aili Wang

Received: 5 September 2024

Revised: 28 September 2024

Accepted: 30 September 2024

Published: 14 October 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Bacterial, viral, and fungal pathogens can exacerbate pulmonary disorders linked to pneumonia [1]. Specifically, in susceptible populations such as the elderly and individuals with preexisting medical conditions, it is a grave condition that can result in severe illness or even mortality [2]. Each year, millions of people contract pneumonia, a potentially lethal respiratory illness. If left untreated, it can have significant detrimental consequences and is typically challenging to identify [3]. Developing novel strategies to promptly and accurately detect pneumonia is crucial for minimizing patient mortality [4]. Computer-aided diagnostic solutions are urgently required in countries with inadequate medical infrastructure, especially in emerging nations. These methods, utilizing artificial intelligence, aid radiologists in detecting acute infection by analyzing chest X-ray pictures [5]. Deep learning techniques, a type of machine learning algorithm, are highly proficient in image recognition tasks. They are specifically employed to extract distinctive attributes from chest X-ray images. This is achieved by training a neural network with a substantial number of photos, enabling it to identify patterns and characteristics in the images. This technique

can be used to categorize photos of pneumonia into different groups, such as healthy or diseased [6].

Machine-learning-based artificial intelligence (AI) research has been making considerable progress in medicine in the past few years. These techniques are now increasingly used for the diagnosis of diseases, creating predictive models to aid clinical decision making and identifying disease-associated factors [7,8]. During this global crisis, several models have been distributed in COVID-19 developments, and one of the most outstanding approaches is metaheuristic algorithms (MAs) [9,10]. One of the most important parts of using models to predict pneumonia outcomes is the choice of the features that can most likely influence the disease diagnosis. There are three main types of feature selection methods: filter method [11,12], wrapper method [13], and embedded-based [14]. KNN is generally really good and does not require knowledge of the data or assumptions to be made for any underlying data; its single parameter can be tuned. The proposed system also enjoys a lower time complexity when compared to a support vector machine (SVM), making this an attractive solution for many MA and feature selection applications. Mohamed et al. [15] proposed a feature selector inspired by the evolution behaviors among predators and parasites, as well as the host, in nature to solve the problem of big data dimensionality and cooperate with KNN for finding the most relevant feature combination. Many deterministic and evolutionary optimization methods have been developed and used to solve single-objective and multiple-objective optimization problems [16,17]. Metaheuristic algorithms (MAs) have demonstrated the capability to solve different optimization problems [18,19]. Traditional approaches that use gradients for optimization are treated as a black box and are more versatile, which means that sometimes MAs are also more efficient than gradient-based approaches [20,21]. Some of the most prominent algorithms among MAs are the genetic algorithm (GA) [22], differential evolution (DE) [23], simulated annealing (SA) [24], and particle swarm optimization (PSO) [25]. In this work, an efficient model that classifies pneumonia into two groups, healthy and infected, is developed and tested using chest X-ray images. Toward maintaining a high accuracy and minimizing false-negative cases, we present a deep learning technique for automated pneumonia classification. In addition, we apply a hybrid optimization that combines two metaheuristic algorithms, FLA and DBO, that have achieved promising results in this area.

The impetus for this research stems from the pressing need to improve pneumonia diagnosis, a leading cause of mortality and morbidity worldwide, especially in vulnerable populations such as the elderly and immunocompromised individuals. Current diagnostic methods, while effective, have limitations, including variability in diagnostic criteria and dependence on high-quality imaging and expert interpretation. These challenges often result in delayed or inaccurate diagnoses, particularly in resource-limited settings. Our study aims to address these gaps by harnessing the power of deep learning and innovative optimization techniques to enhance the accuracy and efficiency of pneumonia detection from chest X-rays. By developing more robust and reliable diagnostic tools, this research seeks to facilitate quicker clinical decision making, reduce the strain on healthcare systems, and, ultimately, improve patient outcomes by enabling timely and precise treatment interventions.

The subsequent sections of this paper are organized in the following manner. Section 2 of this document discusses different approaches employed for the detection and categorization of pneumonia. Sections 3 and 4 outline the methodologies employed to construct the proposed model, encompassing preprocessing techniques and feature extraction using convolutional neural networks (CNNs). Section 5 presents a comprehensive assessment of the models and a detailed analysis of the obtained results. Section 6 serves as the final part of this paper, summarizing the most significant findings.

2. Literature Review

Pneumonia identification from CXR images has been one of the foremost areas of interest, with many deep learning and machine learning techniques promising to advance

diagnostic performance. Nevertheless, each system expresses its own limitations, particularly concerning the challenge of performance consistency for long and diverse data or leveraging optimization algorithms effectively. This section reviews major past works and indicates how our study responds to prevailing shortcomings.

Yi et al. [26] performed pneumonia classification by proposing a model using DCNN. Their mentioned accuracy was 96.09%, the precision rate was 98.61%, and the recall was 93.33%. However, their approach did not involve robust optimization techniques to improve generalization capability and was specifically not designed for large datasets. Kumar et al. [27] proposed a few transfer learning-based models, such as InceptionV3 and ResNet50, with a very impressive accuracy of 96.82% being achieved. Although effective, this method had some limitations in domain feature extraction and relied on pretrained networks; hence, it may not generalize well to new datasets. Mannepalli and Namdeo. [28] developed a hybrid multiscale CNN which was enhanced by embedding self-attention mechanisms that attained an accuracy of 97%. Although their results looked promising, the model did not use advanced optimization techniques to improve feature selection further.

In 2021, Manickam et al. [29] adopted a U-Net architecture with transfer learning for pneumonia diagnosis. The proposed approach yielded an accuracy of 93.06% and a precision of 88.97%. Although the segmentation performance of U-Net sounds impressive, the overall performance of the model was limited due to a relatively lower recall of 96.78% and the unavailability of advanced optimization techniques. Ieracitano et al. [30] proposed a fuzzy-enhanced deep learning method, CovNNet, which managed uncertainties in CXR images but was less effective than other methods and achieved only 80.9% accuracy.

Alqudah et al. [31] classified pneumonia by combining CNN with a support vector machine and obtained 94% accuracy. Yet, even though this hybrid approach provided improvements compared to traditional CNNs, the model still represented high computational cost and performance that was unfavorable due to the need for extracting features from high-dimensional data. Szepesi and Szilágyi [32] further investigated CNN models with a new type of dropout layer placement that achieved 97.21% in accuracy, 97.40% in precision, and 97.34% in recall. This superior architecture has proven to be a worthy contribution to architectural novelty. Punitha and colleagues [33] developed a computer-aided diagnosis (CAD) system utilizing an artificial neural network (ANN) in conjunction with an artificial bee colony (ABC) algorithm, resulting in an accuracy rate of 92.37%. Despite its efficacy in specific contexts, the model's dependence on ABC for the extraction of features resulted in reduced precision and recall rates in comparison to leading methodologies.

Furthermore, several studies utilized deep learning techniques with chest X-ray (CXR) pictures to identify pneumonia. As an example, Ho and Gwak [34] created an ensemble model that integrates handcrafted features, radiomic data, and deep learning characteristics, resulting in an accuracy of 94.1%, precision of 94.5%, and recall rate of 94.1%. Their study highlighted the advantages of merging different features; however, they did not utilize hybrid optimization methods to improve performance further. In addition, Trivedi and Gupta [35] presented a streamlined deep learning framework known as MobileNet, which attained an accuracy of 94.23%. This model's strength lay in its efficiency for mobile applications, while performance lagged compared to more complex architectures. In response to the limitations found in previous studies, this work proposes a hybrid optimization approach geared toward improving the process of feature selection and model performance in pneumonia detection. Next, the methodology section describes in detail how our model leverages the power of deep learning frameworks and optimization techniques in an attempt to achieve better results.

The results highlight the possible utility of this framework in pneumonia identification and assessment of severity, as shown in Table 1. All of them point out novel solutions and algorithms that can enhance the capability of computer-aided diagnosis (CAD) as well as the accurateness and efficiency of CAD in radiological imaging, with considerable improvement in the performance across various imaging methods.

Table 1. Summary of different studies on pneumonia detection using various AI methods.

Ref.	Method	Accuracy	Precision	Recall	F1-Score
[26]	DCNN	96.09%	98.61%	93.33%	-
[27]	Trained transfer learning models	96.82%	95.97%	97.54%	96.74%
[28]	U-Net architecture and transfer learning	93.06%	88.97%	96.78%	92.71%
[29]	Fuzzy-enhanced deep learning method	80.9%	85.2%	82.5%	-
[30]	CNN with SVM	94%	96.68%	93.3%	-
[31]	Novel dropout layer placement in the convolutional part	97.21%	97.40%	97.34%	97.37%
[32]	Hybrid multiscale convolutional network adaptive manta ray foraging optimization	97%	95%	-	96%
[33]	ANN with ABC for region-based fractal analysis	92.37%	-	-	-
[34]	Combines handcrafted, radiomic, deep features	94.1%	94.5%	94.1%	94.0%
[35]	DL-based on “MobileNet”	94.23%	97%	98%	97%

From the comparison between models in the literature reviewed above, we can observe that former methods tend to exploit either via transfer learning model or regular convolutional networks, and they rarely utilize a combination of all optimization approaches. On the other hand, DBO and FLA are both used in our work, and this combination enables a more stable tradeoff between exploration and exploitation, which gives rise to better performance. It performed with an accuracy of 98.19%, which was higher than the current maximum accuracy (97.21% in Table 1). Furthermore, our model’s precision and recall are higher than those of others, making it a reliable tool for predicting pneumonia

3. Proposed Methodology

Our proposed method for pneumonia detection uses chest X-ray images, and first aggregates data consisting of several radiological images labeled as “NORMAL” and “PNEUMONIA”. Each image is preprocessed to a standard size and contrast, enhancing the model’s capability to extract relevant features. After the division of images, CNNs extract deep, intricate features. The process is carried out in a multistep process where data representation is passed through multiple filters and pooling layers to enhance model predictiveness and better capture differentiated patterns of respiratory conditions as anomalies. Toward further refining the model’s predictions, feature selection methodologies are implemented and enhanced through hybrid optimization by the meta-heuristic algorithms, identifying the most significant features for classification with high precision. This hybrid approach will improve the model’s performance by maintaining a balance in exploring and exploiting the search space during feature selection to retain only the most relevant features. In addition, we make use of data augmentation to increase variety in our training data and avoid overfitting to make the finally deployed model more robust. Some metrics used to ensure these algorithms are clinically viable are accuracy, sensitivity, and specificity. Testing and refining of the model are performed constantly to improve the predictions. As shown in Figure 1, this final model is tested with images that it has never seen to engage in more realistic scenarios. The proposed methodology here may provide an efficient automatic tool for the early detection of pneumonia by effectively integrating traditional machine learning approaches and deep learning architectures to support medical practitioners in reaching faster and more accurate diagnostic conclusions.

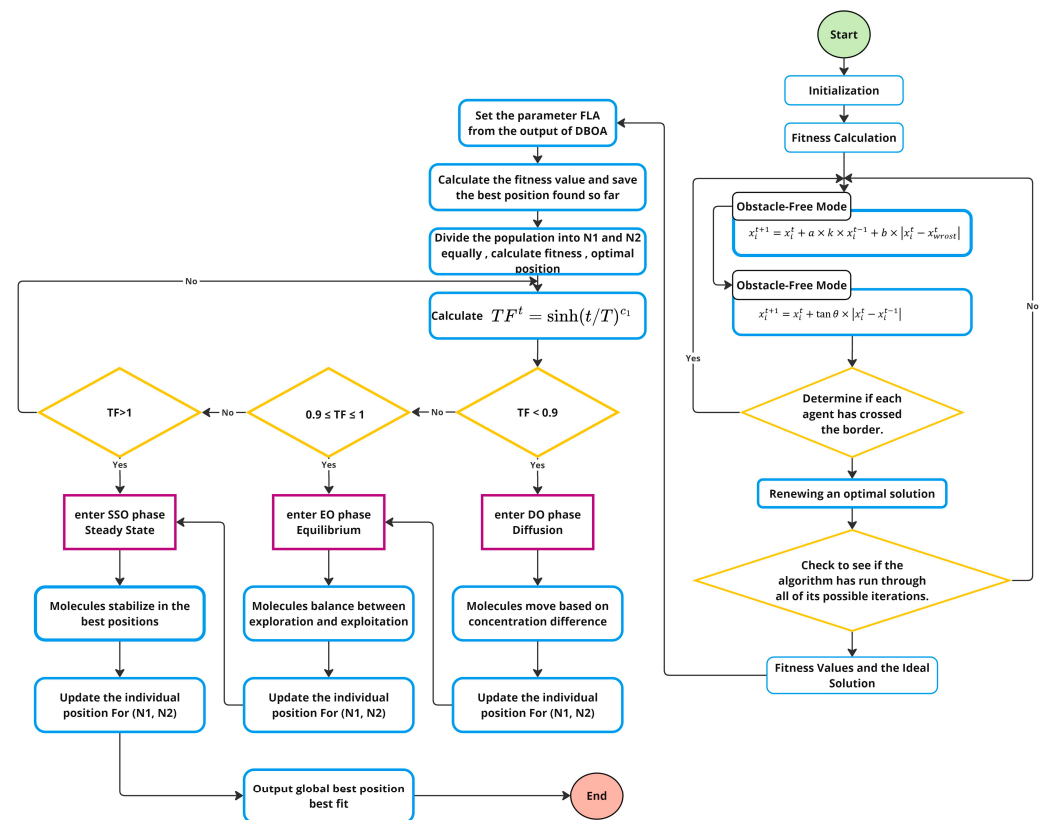


Figure 1. Flowchart showing the stages of the hybrid optimization process.

The robustness of our methodology is derived from incorporating a hybrid optimization technique that optimizes the model's hyperparameters, greatly improving its performance. Merging the dung beetle optimization algorithm (DBOA) with Fick's law algorithm (FLA) establishes a well-rounded exploration and exploitation framework, facilitating the model's convergence towards optimal solutions. The subsequent section elaborates on how this hybrid approach enhances feature selection and elevates overall predictive accuracy.

3.1. Data Overview

The dataset comprises 7750 images in total, and chest X-ray images are divided into two main classes: PNEUMONIA and NORMAL. This provides the basis for training our deep learning model and testing its performance. The dataset is equally weighted between the two classes, which should prevent the model from biasing learning to correctly identify and distinguish healthy findings from findings that are typically associated with pneumonia. Pulmonary structures have various clues in the X-ray images, as shown in Figure 2. PNEUMONIA shows an increased opacity in the lungs, which may indicate the presence of fluid (consolidation) or features suggestive of other infection-related changes in the lungs. In comparison, for NORMAL, we can see clear lung fields, and no significant abnormalities are present. Therefore, the diversity and the quality of these images are very important for the feature extraction and classification processes, which are used to build a reliable diagnostic tool for early detection of pneumonia changes (such as the ones often happening in COVID-19).

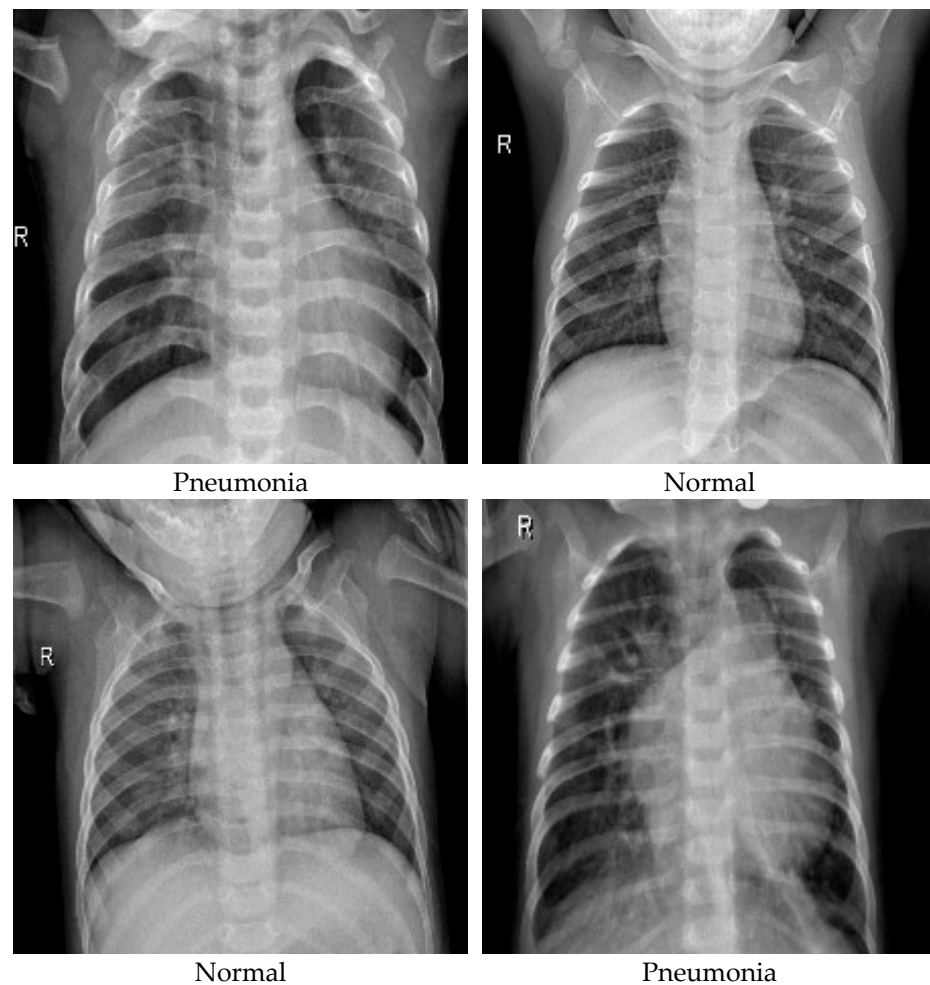


Figure 2. A random sample of chest X-ray images illustrates the two categories in the model.

3.2. EDA and Preprocessing

During the exploratory data analysis (EDA) and preprocessing stage of our chest X-ray and CT image dataset, we conduct an exhaustive analysis of the image data to evaluate its properties and ready it for the next modeling step. We will first load a balanced dataset with a total of 7750 images, equally distributed into PNEUMONIA and NORMAL labels. It contains images that are fetched from a specified folder whose structure comes from the folders of their labels. These images are preprocessed in several steps to convert them into a suitable form for deep learning models. We resize each image to 224×224 pixels and then convert it to RGB from BGR to fit the desired input of our model architecture. Images are also normalized to pixel values ranging from 0 to 1, which helps with the convergence of neural networks during training. Data preparation for training, validation, and testing can be performed by dividing our dataset into training, testing, and validation sets, where 90% of the data are used for training and 10% for testing. This dataset is divided into training and validation sets (80/20 split). A model is trained from the training set. Due to this stratified splitting, all three sets contain all classes in the same proportion, making sample distribution balanced across the sets essential to avoid biased model training. Visual exploration of the data: We plot the data to see the distribution and characteristics of the dataset. We plot bar charts to check the distribution of function classes in both the training and test sets and confirm that class imbalance might not impact our training and evaluation. Also, it displays example images of the dataset to gain an idea of the variations within the classes as well as in between. These visualizations serve to verify the existence of informative patterns among the normal and pneumonia X-rays. Statistical analysis—Again, we test the model on its performance matrix, such as using accuracy, precision, recall, and

F1-score. In addition, we calculate the sensitivity and specificity of each class to analyze how well the model identifies each disease status. Being pretty self-explanatory, these metrics are essential for gauging whether or not the model is performing well and how it will perform in different conditions of a dataset. This meticulous EDA and preprocessing step lays down the first stone for building a performant model that can label X-ray images correctly, either PNEUMONIA or NORMAL, for the diagnosis of respiratory diseases associated with the lung. These metrics are essential for evaluating the model's performance and its ability to function under various conditions within the dataset. The meticulous exploratory data analysis (EDA) and preprocessing steps establish the foundational work for constructing a performant model capable of accurately labeling X-ray images as either PNEUMONIA or NORMAL, facilitating the diagnosis of respiratory diseases associated with the lungs.

The preprocessing of chest X-ray images plays a pivotal role in enhancing the performance of our deep learning models. Each image undergoes several crucial preprocessing steps—resizing, normalization, and augmentation—that directly influence model training and effectiveness. Initially, images are resized to a uniform dimension (e.g., 224×224 pixels), ensuring that the neural network receives inputs of consistent size, which is vital for maintaining the integrity of spatial features across different images. Following resizing, normalization is applied to adjust the pixel values so that they fall within a specific range (typically 0 to 1). This step is critical for stabilizing the learning process as it ensures that the model is not biased toward images with higher pixel intensity variations, thus facilitating faster and more stable convergence during training. Augmentation techniques such as random rotations, shifts, and flips are employed to artificially expand the training dataset. This not only helps prevent the model from overfitting but also enhances its ability to generalize from the training data to new, unseen images by exposing the model to a wider variety of scenarios and perspectives found in medical images. Together, these preprocessing techniques contribute significantly to the robustness and accuracy of the model, enabling it to perform with high reliability in clinical settings.

3.3. Dung Beetle Optimizer

The dung beetle optimizer (DBO) is an exploration-oriented algorithm that navigates the search process by means of guidance based on dung beetles' navigation strategy. It, thus, enables the model to overcome local optima in pursuit of better solutions. In contrast, Fick's law algorithm (FLA) improves exploration by simulating diffusion to guide solution spreading. With the combination of DBO and FLA, we combine two types of evolutionary algorithms in a complementary manner to balance exploration and exploitation, which helps ensure better model optimization, which results in higher pneumonia detection accuracy. A flowchart representation of the relationship between both optimizers and within the training process is shown in Figure 1.

In this part, we present an original swarm intelligence (SI)-based optimization algorithm called the dung beetle optimizer (DBO). The concept defined for this technique derives from the actions of dung beetles, and the mathematical simulation of this process is explained as follows: Depending on the type and stage of decomposition, dung beetles like **Copris ochus Motschulsky**, **Onthophagus gibbulus**, and **Caccobius jessoensis Harold** are typical decomposers in ecosystems all over the world. These beetles are known to be a rather curious lot as far as dung is concerned; they shape the dung into balls and roll it. This behavior is as interesting as it is essential for their survival and reproduction. These are rowdy insects that use the sun in order to maintain a straight course while rolling dung balls. This suggested navigational skill is crucial in the simulation model for the DBO algorithm and its implementation. The transport algorithm emulating rolling in the relocation of a dung beetle in the defined search space is identified mathematically. The position update equation provided for the dung beetle algorithm can be clarified with a

correct description of each parameter in the equation. The following is the corrected and detailed explanation:

$$x_i(t+1) = x_i(t) + \alpha \times k \times x_i(t-1) + b \times \Delta x, \quad (1)$$

where

- $x_i(t+1)$ represents the position of the i -th dung beetle at the next iteration $t+1$.
- $x_i(t)$ denotes the position of the i -th dung beetle at the iteration t .
- $x_i(t-1)$ indicates the position of the i -th dung beetle at the previous iteration $t-1$.
- k is the deflection coefficient, influencing how past movements affect current direction.
- b is a constant ($0 < b < 1$) that scales the influence of the distance from the worst position.
- α is a coefficient set to either 1 or -1 , which adjusts the direction based on external factors like wind or uneven terrain.
- Δx is the magnitude of the distance from the worst position in the search space, calculated as $|\Delta x = x_i(t) - X_w|$, where X_w refers to the worst position observed.

This model allows the algorithm to explore the search space efficiently, avoiding local optima by incorporating past movement and the relative quality of current positions.

3.3.1. Updating the Rolling Direction

To mimic the dung beetle's ability to adjust its rolling direction based on environmental cues, we use the tangent function, which only considers values within the interval $[0, \pi]$ where θ is the deflection angle, constrained within $[0, \pi]$:

$$x_i(t+1) = x_i(t) + \tan(\theta) \times |x_i(t) - x_i(t-1)| \quad (2)$$

Here, $(x_i(t+1))$ represents the new position of the i -th dung beetle at the next iteration, adjusted by the current position $(x_i(t))$ and influenced by the change in direction, which is guided by the tangent of the deflection angle (θ) . The term $(\tan(\theta))$ determines how sharply the beetle changes direction, with (θ) constrained within the interval $[0, \pi]$ to ensure a manageable turning angle that mimics natural beetle behavior. The difference $(|x_i(t) - x_i(t-1)|)$ denotes the magnitude of the distance between the current and previous positions, representing the beetle's past trajectory, which helps inform its new course. This update equation allows the beetle to adjust its rolling direction smoothly, balancing between its previous path and the environmental cues (e.g., the sun's position or terrain unevenness) represented by the deflection angle.

3.3.2. Boundary Selection Strategy

The boundary selection strategy mimics the dung beetle's behavior in which the insect selects appropriate ground in which to deposit eggs. The spawning area boundaries are dynamically defined by

$$Lb^* = \max(X^* \times (1 - R), Lb) \quad (3)$$

$$Ub^* = \min(X^* \times (1 + R), Ub) \quad (4)$$

where

- X^* is the current position known to be the best in local search.
- Lb and Ub represent, respectively, the lower and upper limits of the problem space.
- $R = 1 - t/T_{max}$ is adapted iteratively with the iteration number t and the maximum number of iterations T_{max} .

This will make the area of spawning active and, thus, the search area dynamic, which helps to enrich the pattern of the DBO algorithm further when searching for the solution. This extension not only extends the optimization techniques but also enriches the usage of bio-inspired algorithms in problem-solving processes by mimicking the peculiar behaviors that are found in nature.

3.4. Fick's Law Optimization

Fick's law is a basic law in physics and chemistry that defines the linear movement of substances in a concentration gradient. Based on this principle, Fick's law algorithm (FLA) continuously reenacts the movement of diffusion in searching for better solutions. Another important principle in diffusion is Fick's law, which states that the rate of diffusion increases with an increase in concentration gradient. The FLA further distorts this notion in such a way that by relocating the positions of the molecular fragmentations in different districts, they end up stabilizing positions, and the process works best. The optimization process begins by initializing a population matrix X , represented as

$$\begin{bmatrix} x_{1,1} & x_{1,2} & \dots & \dots & x_{1,D} \\ x_{2,1} & x_{2,2} & \dots & \dots & x_{2,D} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{N,1} & x_{N,2} & \dots & \dots & x_{N,D} \end{bmatrix} \quad (5)$$

where N is the population size, and D is the dimensionality per individual member of the population. A typical element, $X_{i,j}$ of the matrix represents the j -th dimension of the i -th molecule. The matrix is initialized with the following equation:

$$x_{i,j} = lb + rand(1, D) \times (ub - lb) \quad (6)$$

where lb and ub are the lower and upper bounds of the search space, respectively. The population is split into two subfamilies, N_1 and N_2 , to measure their respective fitness. Molecules move from regions of high concentration to low, guided by the iteration parameter TF^t , defined by the following equation:

$$TF^t = \sin h\left(\frac{t}{T}\right)^{c1} \quad (7)$$

The representation of this algorithm is as follows, where t is the current iteration number, and T is the maximum number of iterations. The molecular position update mechanism involves three distinct stages: diffusion operator (DO), equilibrium operator (EO), and steady-state operator (SSO), which are used to find the best value for the system. The updated equation is

$$X_i^t = \begin{cases} DO & \text{if } TF^t < 0.9 \\ EO & \text{if } 0.9 \leq TF^t \leq 1 \\ SSO & \text{if } TF^t > 1 \end{cases} \quad (8)$$

where (TF^t) in Equation (7) represents the transition factor at the current iteration (t), and it is calculated using the hyperbolic sine function $\left(\sin h\left(\frac{t}{T}\right)^{c1}\right)$, where (t) is the current iteration number, (T) is the maximum number of iterations, and $(c1)$ is a control parameter that determines the behavior of the transition function. The transition factor (TF^t) influences the algorithm's ability to explore or exploit the search space over time.

In Equation (8), (X_i^t) denotes the position of the (i) -th molecule at the (t) -th iteration, which is updated through different operators based on the value of (TF^t) . If $(TF^t < 0.9)$, the diffusion operator (DO) is applied, facilitating exploration of the search space. If $(0.9 \leq TF^t \leq 1)$, the equilibrium operator (EO) is used, balancing exploration and exploitation. Finally, if $(TF^t > 1)$, the steady-state operator (SSO) is applied, emphasizing exploitation to refine the solution in the later stages of optimization.

3.4.1. Exploration Phase

In the diffusion-only phase where $TF^t < 0.9$, the transports of molecules from one region to the other occur with respect to the concentration gradient. The parameter guiding this transfer, T_{DO}^t , is defined by

$$T_{DO}^t = C_5 \times TF_t - r, \quad (9)$$

The constant C_5 is normally set to 2, and r is a random number generated from a uniform distribution with expectations in the ranges 0 and 1. The direction of molecule movement is determined by

$$X_{p,i}^t \begin{cases} \text{from } i \text{ to } j; & T_{DO}^t < rand \\ \text{from } j \text{ to } i; & \text{others} \end{cases} \quad (10)$$

The number of molecules moving from region i to j is calculated as follows:

$$NT_{ij} \approx X_i \times r1 \times (C_4 - C_3) + N_i \times C_3 \quad (11)$$

where C_3 , and C_4 are other constants, and $r1$ is another random number between 0 and 1. This optimization framework is able to capture the diffusion process, and the search space can be adapted dynamically in accordance with the natural diffusion laws alluded to by Fick's second law.

In Equation (9), (T_{DO}^t) represents the transfer parameter during the diffusion operator phase, which determines the movement of molecules from one region to another. The parameter (T_{DO}^t) is calculated using a constant (C_5), typically set to 2, and the current transition factor (TF_t). The term (r) is a random number generated from a uniform distribution between 0 and 1, introducing randomness to simulate natural diffusion behavior.

In Equation (10), the molecule movement direction between regions (i) and (j) is determined by comparing the value of (T_{DO}^t) with a random number ($rand$) from a uniform distribution. If (T_{DO}^t) is less than ($rand$), molecules move from region (i) to region (j); otherwise, they move from region (j) to region (i). In Equation (11), the number of molecules (NT_{ij}) moving from the region (i) to the region (j) is calculated based on the concentration of molecules in the region (i), denoted by (X_i), multiplied by a random number ($r1$) between 0 and 1. The constants (C_3) and (C_4) represent diffusion coefficients that affect the movement, where (C_3) accounts for local adjustments, and (C_4) influences the overall gradient. The term (N_i) represents the number of molecules initially present in the region (i), and the product ($N_i \times C_3$) ensures the diffusion process respects the molecule distribution in the current region.

3.4.2. Transition Phase from Exploration to Exploitation

As the system moves from exploration to exploitation, binding modes become extremely important in setting all those molecular positions and adjusting them so that they will be stable at an equilibrium position. The result will be at equilibrium in which all parts of the region across a level surface are equally flat, and this process makes it easy to conduct very detailed searching near the current best solutions. This systematic transition is critical to both expose potential solutions more thoroughly and avoid missing local nuances within the search space by subsetting out observations based on probabilities. The FLA can navigate complex optimization landscapes using switches between wide exploratory movements and fine exploitative adjustments to identify global optima without prematurely converging upon local optima. This balanced methodology is fundamental to the effectiveness of this algorithm with a variety of optimization problems. The second phase is stricken in Fick's law algorithm (FLA) as it indicates the proper transition from the initial exploration phase, when individuals appear to discover fresh things, to focused exploitation, gathering the last bits of information. It occurs via the equilibrium operator (EO), which makes molecules move to an equilibrium state, where their concentration is

stable in all parts of the system. This is a key turning point; it focuses the search on the best prospects that have been identified during exploration.

3.4.3. Equilibrium Operator (EO)

During the EO phase, molecules adapt their positions to balance out the concentration gradients that become nearly negligible, effectively simulating the natural diffusion process reaching a state of equilibrium. The position update during this phase is governed by the following equation:

$$X_{p,g}^{t+1} = X_{EO,p}^t + Q_{EO,g}^t \times X_{p,g}^t + Q_{EO,g}^t \times \left(MS_{p,EO}^t \times X_{EO,g}^t - X_{p,g}^t \right) \quad (12)$$

where

- $X_{p,g}^t$ and $X_{EO,g}^t$ denote the positions in groups p and g .
- $MS_{p,EO}^t$ represents the relative quantity of group g .
- $Q_{EO,g}^t$ is the diffusion rate factor for the region of group g , calculated as follows:

$$Q_{EO,g}^t = R_1^t \times DF_g^t \times DRF_{EO,g}^t \quad (13)$$

where

- DF_g^t is the directional factor (± 1).
- R_1^t is a random number between 0 and 1.
- $DRF_{EO,g}^t$ represents the diffusion rate, defined by:

$$DRF_{EO,g}^t = \exp\left(-\frac{J_{p,EO}^t}{TF^t}\right) \quad (14)$$

- $J_{p,EO}^t$ is the flux, computed as

$$J_{p,EO}^t = -D \frac{dc_{g,EO}^t}{dx_{p,EO}^t} \quad (15)$$

where $dc_{g,EO}^t$ and $dx_{p,EO}^t$ denote the concentration and position differences between regions g and p , respectively. The flow of molecules is directed by minimizing the concentration differences, effectively simulating the diffusion process as dictated by Fick's law.

3.4.4. Optimization in Stable Phase

The stable phase of the algorithm involves refining the molecule positions. Where each molecule moves towards the optimal position, the distance of movement is inversely proportional to the fitness; molecules with higher fitness have smaller movement distances. The updated formula for molecule positions in this stable phase is

$$X_e = D_{alphs} \times e^{II} \times \cos(II \times 2\pi), \quad (16)$$

where

- X_e : represents the new position. This is the result of the equation and indicates the updated or new position after applying the transformation based on the other variables.
- D_{alphs} : This variable is a coefficient or parameter that scales the exponential and cosine functions. It could represent a distance factor, a scaling factor, or some characteristic value related to the system being modeled.
- e : This is the base of the natural logarithm (approximately equal to 2.71828). It is used here in an exponential function to model growth or decay.
- II : This is a random number that balances global and local search efforts. It is calculated as

$$II = (Fc - 1) \times \text{rand} + 1 \quad (17)$$

where

- rand: A random number, typically between 0 and 1.
- Fc: A control factor used to balance the exploration (global search) and exploitation (local search) phases of an algorithm. It determines the extent to which the search process should focus on exploring new areas versus exploiting known good areas. When Fc is high, the algorithm tends to explore more globally; when Fc is low, it focuses more on local searches.
- $\cos(\Pi \times 2\pi)$: This is the cosine function, which introduces oscillatory behavior into the equation. The argument $\Pi \times 2\pi$ ensures that the cosine function cycles through its periodic behavior, contributing to the new position X_e .

In summary, this equation X_e uses a combination of exponential growth/decay and oscillatory (cosine) behavior, influenced by a random factor Π that is itself controlled by Fc, to determine a new position in a search or optimization process. The aim is to balance between exploring new areas and exploiting known good areas in the search space. Total time complexity calculation: The total time complexity of the FLAS across all phases, considering the complexity contributions from different phases, can be aggregated as follows:

$$O(FLA) = O(N \times D) + O(T \times (O(NT_{12} \times D) + O((N/2 - NT_{12}) \times D))) + O((N/2 - N_{transfer}) \times D) + 6 \times O(N/2 \times D) \quad (18)$$

Simplifying this, considering the largest terms dominate in Big O notation, particularly when T is large:

$$\begin{aligned} O(FLA) &= O(N \times D) + O(T \times 8 \times N/2 \times D) \\ &= O(N \times D + 4 \times T \times N \times D) \\ &= O(N \times D \times (1 + 4 \times T)) \end{aligned} \quad (19)$$

This formula illustrates that the time complexity of FLA is linear with respect to both the population size and the dimensionality, and it scales linearly with the number of iterations multiplied by a factor of three. This demonstrates that while the algorithm is efficient per iteration, its overall computational expense grows significantly with increasing iterations and population size, which is a common trait among population-based optimization algorithms.

In Equation (18), the total time complexity of the FLA, ($O(FLA)$), is calculated by summing the contributions of different phases in the optimization process. The term ($O(N \times D)$) refers to the complexity of initializing the population with (N) population members and (D) dimensions. The subsequent terms, ($O(T \times (O(NT_{12} \times D)))$) and ($O((N/2 - NT_{12}) \times D)$), represent the complexities associated with managing the diffusion of molecules and the handling of nontransferred molecules during the optimization process across (T) iterations. The terms ($O(N_{transfer} \times D)$) and ($O(N/2 - N_{transfer} \times D)$) correspond to the complexities related to the molecule transfer phase. The final term, ($6 \times O(N/2 \times D)$), accounts for additional operations involved in the optimization.

In Equation (19), the simplified form of the total time complexity emphasizes that the most dominant terms contribute to the overall computational expense. By simplifying the expression, we see that the total time complexity, ($O(N \times D \times (1 + 4 \times T))$), grows linearly with both the population size (N) and the dimensionality (D), as well as with the number of iterations (T). This result highlights that while the algorithm is efficient on a per-iteration basis, its overall computational load increases with larger population sizes and more iterations, which is a characteristic trait of population-based optimization algorithms.

In this section, we elaborate on the dung beetle optimizer (DBO) and Fick's law algorithm (FLA) to explain how each formula contributes to the optimization process and how the variables and parameters interact to drive the solution toward the optimal point. In Equation (1), the key variables such as ($x_i(t + 1)$) (representing the updated position of

the i th individual) and $(x_i(t)), (x_i(t - 1))$ (representing current and past positions) allow the DBO to balance exploration based on prior movements. Parameters like (α) and (k) adjust for the effects of external factors and directional changes, while (Δx) ensures that the algorithm can avoid local optima by considering the worst position observed. This enables the DBO to efficiently explore the solution space and make meaningful adjustments.

Similarly, Equation (2) governs the rolling direction adjustment of the population based on environmental cues, using the tangent function to fine-tune the position updates. The deflection angle (θ) , within the interval $[0, (\pi)]$, controls how much adjustment is made, allowing the optimizer to maintain its path while correcting direction when necessary.

In FLA, diffusion principles are applied in Equation (8) and the subsequent related equations. These equations simulate molecular diffusion as a way to spread out and refine solutions based on the concentration gradients between areas of the search space. Variables such as (T_{DO}) , (NT_{ij}) , and the iterative concentration updates $((\Delta c))$ ensure that solutions are adjusted with precision, moving towards equilibrium through controlled diffusion. Equilibrium operators (Equation (12)) stabilize the molecular positions, ensuring balanced search efforts.

The final time complexity equations (Equations (18) and (19)) offer a comprehensive evaluation of the computational expense of the algorithm, ensuring that the proposed method maintains efficiency even as the number of iterations or population size grows.

By explaining these equations in detail, we clarify how DBO and FLA integrate exploration and exploitation mechanisms, systematically driving the population toward optimal solutions. This breakdown highlights the precision with which the optimization is carried out, leveraging biological and physical principles to improve model performance.

Algorithm 1 describes the detailed process of hybrid optimization, combining the dung beetle optimizer (DBO) and Fick's law algorithm (FLA), integrating 18 equations that collectively form the backbone of this optimization framework. Each of these equations plays a critical role in shaping the exploration–exploitation balance, parameter updates, and boundary conditions essential for the hybrid optimization's success.

The process begins with the initialization phase, governed by equations that determine the initial positions and fitness evaluations of the population members. Equation (5) defines the population matrix, while Equation (6) initializes the individual elements within this matrix. These equations ensure that the search space is appropriately covered and that each individual starts with a diverse set of parameter configurations.

During the exploration phase, the DBO's update rules, encapsulated in Equation (1), drive the population's movement across the search space. The role of Equation (1) is to emulate the navigational strategy of dung beetles, which helps the optimizer efficiently traverse the solution landscape and escape local optima. Meanwhile, boundary selection strategies, defined by Equations (3) and (4), ensure that the individuals remain within the allowable search space by dynamically adjusting the bounds based on the best-known solutions. This guarantees that the optimizer maintains control over the search domain while enabling sufficient exploration.

As the algorithm transitions from exploration to exploitation, FLA comes into play, with Equation (2) governing how solutions are fine-tuned based on the fitness evaluations of neighboring individuals. This refinement process is critical for ensuring that the solutions converge toward the global optimum by focusing on exploitation and fine-tuning the results after a broader exploration phase. The transition between the exploration and exploitation phases is further reinforced by equations like Equation (7), which defines the iteration parameter guiding the switching behavior between these phases.

Within the exploitation phase, Equations (8)–(12) dictate the update strategies for refining the solution space. These equations introduce concepts from Fick's law, simulating diffusion processes that allow the optimizer to adjust molecular positions effectively. This phase ensures that solutions are thoroughly explored within localized areas, refining the precision of the optimizer's movements toward the most promising regions of the search space.

Additionally, the boundary adjustments and population refinement techniques are continually updated using Equations (13)–(15), which define the diffusion rate factors and concentration gradients. These equations simulate molecular interactions, ensuring that the optimization process smoothly transitions from global to local searches while maintaining accuracy in the adjustments. Equation (16) provides further refinement by updating the positions based on exponential and cosine functions, balancing exploration and exploitation.

The iterative nature of the algorithm is captured by Equation (18), which calculates the total time complexity of the process. This complexity depends on population size, dimensionality, and iteration counts, showing the efficiency of the algorithm across different scales. Equation (19) simplifies the complexity, demonstrating that while the algorithm is computationally intensive, it maintains efficiency even as iterations increase.

Together, these 18 equations define how DBO and FLA interact to guide the population toward optimal solutions, balancing broad exploratory searches with precise exploitation. The interaction between these equations ensures that the hybrid optimization is robust, adaptable, and capable of producing high-quality solutions for the pneumonia detection model. Each step of the algorithm is grounded in these equations, providing a mathematically rigorous framework that supports the optimization's convergence to an optimal set of parameters.

Algorithm 1: Detailed hybrid DBO and FLA optimization process.

```

1: Input: Population size N, parameter bounds parameter_bounds, objective function obj_func,
   maximum iterations
   max_iterations
2: Output: Best found parameters and corresponding loss
3: Initialize population with N individuals randomly within parameter_bounds
4: Evaluate fitness of each individual in population using obj_func
5: Identify best_idx with the lowest fitness
6: for iteration  $\leftarrow$  1 to max_iterations do
7:   Exploration phase using DBO
8:   for i  $\leftarrow$  1 to N do
9:     Apply DBO update rule to population[i] using the best individual population[best_idx]
   See Equation (1)
10:    Ensure population[i] respects parameter_bounds by clipping or wrapping
11:   end for
12: Evaluate fitness of the updated population
13: Sort population by fitness ascending
14: Exploitation phase using FLA
15: for i  $\leftarrow$  1 to N – 1 do
16:   if fitness[i] > fitness[i + 1] then Condition for FLA interaction
17:     Apply FLA update between population[i] and population[i + 1] based on their fitness
   See Equation (2)
18:   end if
19: end for
20: Re-evaluate fitness of the modified population
21: Update current_best_idx if a new lower fitness is found
22: if fitness[current_best_idx] < fitness[best_idx] then
23:   best_idx  $\leftarrow$  current_best_idx
24:   Update global best solution with population[best_idx] and its fitness
25: end if
26: Optionally record history or intermediate outputs for analysis
27: end for
28: return population[best_idx], fitness[best_idx]

```

Although all 18 equations play a critical role in shaping the hybrid optimization framework, we chose to integrate only Equations (1) and (2) explicitly in Algorithm 1 to maintain clarity and focus on the core mechanics of the algorithm. These two equations encapsulate the primary operations of the dung beetle optimizer (DBO) and Fick's law algorithm (FLA), which are the fundamental drivers of exploration and exploitation in the optimization process. Equation (1) governs the DBO's position update mechanism, which allows the algorithm to explore the search space efficiently by simulating the dung beetle's navigation strategy. Similarly, Equation (2) directs the fine-tuning adjustments during the FLA exploitation phase, ensuring precise refinement of solutions.

By highlighting these two equations, we emphasize the novel contributions of the hybrid optimization process without overburdening the reader with excessive mathematical details. The remaining equations, though crucial, serve as supportive mechanisms for initialization, boundary adjustments, diffusion rates, and convergence criteria, which are standard components in most evolutionary algorithms. Therefore, while Equations (3)–(18) ensure the algorithm's completeness and effectiveness, they are not explicitly included in Algorithm 1 to avoid overwhelming complexity. This approach allows us to strike a balance between showcasing the algorithm's innovative components and maintaining readability for those less familiar with the intricate mathematical foundations of the entire process.

3.5. CNN Architecture

The input size 224×224 is used to process each image with three color channels in the convolutional neural network coded for this study. The architecture begins with a convolutional layer containing 32 filters, each of size 3×3 with an ReLU activation function, as it allows for nonlinearity to be introduced into our model. Next, a max-pooling layer with a 2×2 window to perform spatial dimensionality reduction lowers the computational cost. Each subsequent layer copies this structure and contains 32 more filters of the same size (32 in total) before one more 2×2 max pool layer. In order to prevent overfitting, after the second pooling operation, a large dropout rate of 80% is used, which somehow turns off a share of feature detectors during the training at random and thus makes the net learn more robust features. Following convolutional and pooling layers, the network flattens the multidimensional output of the convolutional and pooling layers into a one-dimensional array that feeds into a fully connected neural network. The fully connected segment of the network has 128 neurons and constitutes a dense layer with ReLU activation function again to keep up with nonlinearity. The output layer is the final layer that has two neurons for the two class predictions and uses a SoftMax activation function since we want the predicted class labels to be output as a probability distribution over all the class labels. Adam optimizer is used to optimize the architecture with a learning rate of 0.001. The network is trained to minimize cross-entropy loss, which is a commonly used choice in classification problems. The model performance is further improved by using early stopping, which stops training if the validation loss does not decrease for ten consecutive epochs, together with the learning rate reduction technique, which decreases the learning rate by five times if no improvement is observed in five epochs, down to a minimum value of 0.0001. It does so by automatically training the model based on different learning rates while at the same time adjusting the learning rate to validation performance, thus avoiding overfitting on the training set and maintaining robustness with unseen data. The model has trained 25 epochs by feeding it batches of 16 images and data augmentation being performed in real time (random rotations, width, and height shifts, horizontal flips) to allow the model to generalize. The result of this architecture of the model is presented in Table 2.

Table 2. Parameters of the convolutional neural network architecture.

Parameter	Value
Input Image Size	$224 \times 224 \times 3$
Convolution Layers	2
Filters per Conv Layer	32
Filter Size	3×3
Activation Function (Conv Layers)	ReLU
Pooling Layers	2
Pooling Type	Max Pooling
Pooling Window Size	2×2
Dropout Rate	0.8
Dense Layers	1
Neurons in Dense Layer	128
Activation Function (Dense Layer)	ReLU
Output Layer Neurons	2
Activation Function (Output Layer)	Softmax
Optimizer	Adam
Learning Rate	0.001
Loss Function	Categorical Cross-entropy
Batch Size	16
Epochs	25
Data Augmentation	Rotation, Width and Height Shift, Horizontal Flip
Early Stopping Patience	10 epochs
Reduce LR on Plateau Factor	0.2
Minimum LR	0.0001

4. MobileNet Architecture

In the next subsection, the architecture of the MobileNet pretrained frameworks is used based on depthwise separable convolution. That structure factors the convolution into depthwise and pointwise layers, reducing the load of computation and model size drastically while retaining robustness. This is optimized for MobileNet and a great use case in embedded vision applications, reducing the sizes of parameters well and working quite satisfactorily to date. The model is designed for medical image analysis, with an input shape of 224×224 pixels and three color channels. To avoid overfitting, a modified version of MobileNet was obtained by integrating two dense layers—one with 1024 neurons and another with 512—each with attached dropout rates of 10% and 50%, respectively. We used a final softmax layer to classify the images into two classes (normal and pneumonia).

Extensive tuning of hyperparameters was performed in order to improve the performance of the MobileNet model. Different key hyperparameters like learning rate, dropout rate, number of dense layers, and batch size were carefully tuned in order to have an optimal model learning using the Adam optimizer while setting the loss function as categorical cross-entropy. In order to improve the outcome of this experiment further, some augmentation techniques like random rotation, horizontal flip, width, and height shift were performed during training time. These techniques are important in preventing overfitting in a model, as are dropout and normalization, enhancing the generalizing capability from training data onto new, unseen images. By exposing the model to a wider array of scenarios that naturally occur in medical imaging, we make it significantly more robust and applicable to real-world clinical settings.

It was applied on 25 epochs, incorporating early stopping and learning rate reduction strategies to tune, check convergence, and avoid overfitting. Data are divided into training, validation, and test sets to ensure that the model generalizes unseen statistics well. It was compared to the baseline CNN model to check the performance of the MobileNet model. With MobileNet, the performance in classification was improved with a more homogeneous accuracy, improving behavior across classes, especially relevant in medical diagnostics, which is the balance of misclassifications.

Accuracy of MobileNet Model: 98.19% Above the CNN model, the minimum was around 95.35%. Precision, recall, and F1-score: Other metrics from the confusion matrix also supported the good performance of the MobileNet-based model, showing that both precision (edge towards false positive) and recall values are high, i.e., good sensitivity and low false positive rate. This amount of scrutiny interpolated to the confusion matrix analysis also showed how well our model can categorize both types. A pipeline was designed for evaluation using various criteria; after experimental results from a good dataset, it was concluded that with its features, the model is able to distinguish pneumonia from nonpneumonia diseases more efficiently.

5. Results and Discussion

As a result of the experiments, images were split into two categories (CNN), and MobileNet architectures were used. In terms of classification for this particular task, these two operated like black boxes: NORMAL and PNEUMONIA. The CNN model achieved an overall accuracy rate of $95.35 \pm 1.48\%$. The precision for NORMAL was $96 \pm 1.38\%$ while that of PNEUMONIA was $94 \pm 1.67\%$, so considering F1-scores exceeding $95 \pm 1.53\%$ from both categories, this is all very good performance. For NORMAL, sensitivity was roughly $94.38 \pm 1.63\%$, with a specificity of $96.38 \pm 1.32\%$. The results were the opposite of PNEUMONIA. Its sensitivity, on the other hand, hit a higher $96.38 \pm 1.32\%$ and specificity $94.33 \pm 1.63\%$, respectively. MobileNet, however, still came out ahead with superior results: its accuracy reached $97.94 \pm 1.00\%$. It achieved a precision of $99 \pm 0.70\%$ for NORMAL as well as $97 \pm 1.20\%$ resistance in the case of pneumonia, while F1-scores for these were both $98.98 \pm 0.99\%$. For NORMAL, sensitivity zoomed all the way up to $97.97 \pm 1.20\%$. A specificity of $98.97 \pm 0.71\%$ was equally good. PNEUMONIA sensitivity was even higher than this at $99 \pm 0.70\%$ and specificity only slightly lower, or $96.91 \pm 1.22\%$. It achieved an accuracy of $98.19 \pm 0.94\%$. MobileNet was further optimized using a combination optimization technology. In this optimized environment, NORMAL achieved an impeccable precision of 100% and a sensitivity of $97 \pm 1.20\%$, plus an outstanding specificity of $99.74 \pm 0.31\%$. In the case of PNEUMONIA, its performance followed suit: the precision rate came to $97 \pm 1.2\%$, with sensitivity standing at an impressive 100% and specificity reaching $96.65 \pm 1.27\%$. These results are displayed in Table 3.

Comparing what is in the table, this analysis shows how well our convolutional neural network (CNN) and MobileNet designs detected pneumonia compared with other research models that produced lower results for performance indicators. An accuracy of $95.35 \pm 1.48\%$ was achieved by our main CNN model, along with identical precision and recall, as well as an F1-score that was equal in all categories. This indicates balanced performance even across various categories in which different classes of pneumonia cases are diagnosed. When the hyperparameters of the MobileNet architecture were fine-tuned, accuracy improved to $97.94 \pm 1.00\%$. Also, when a hybrid optimizer was further applied to the MobileNet, we saw a slight increase in accuracy, now to $98.19 \pm 0.94\%$. At the same time, high-performance levels were maintained, as measured by all metrics. This optimization highlights that just making minor changes can have a major impact on how efficient and accurate the model is when ultimately used for medical imaging tasks. Figure 3 shows the performance of these models.

Table 3. Comparison of classification models.

Metric	CNN	MobileNet	Optimized MobileNet
Accuracy	95.35 $\pm$ 1.48%	97.94 $\pm$ 1.00%	98.19 $\pm$ 0.94%
NORMAL			
Precision	96 $\pm$ 1.38%	99 $\pm$ 0.70%	100%
Recall	94.38 $\pm$ 1.63%	97 $\pm$ 1.20%	97 $\pm$ 1.20%
F1-score	95 $\pm$ 1.53%	98 $\pm$ 0.99%	98 $\pm$ 0.99%
Specificity	96.38 $\pm$ 1.32%	98.97 $\pm$ 0.71%	99.74 $\pm$ 0.31%
PNEUMONIA			
Precision	94 $\pm$ 1.67%	97 $\pm$ 1.20%	97 $\pm$ 1.2%
Recall	96.38 $\pm$ 1.32%	99 $\pm$ 0.70%	100%
F1-score	95 $\pm$ 1.53%	98 $\pm$ 0.99%	98 $\pm$ 0.99%
Specificity	94.33 $\pm$ 1.63%	96.91 $\pm$ 1.22%	96.65 $\pm$ 1.27%

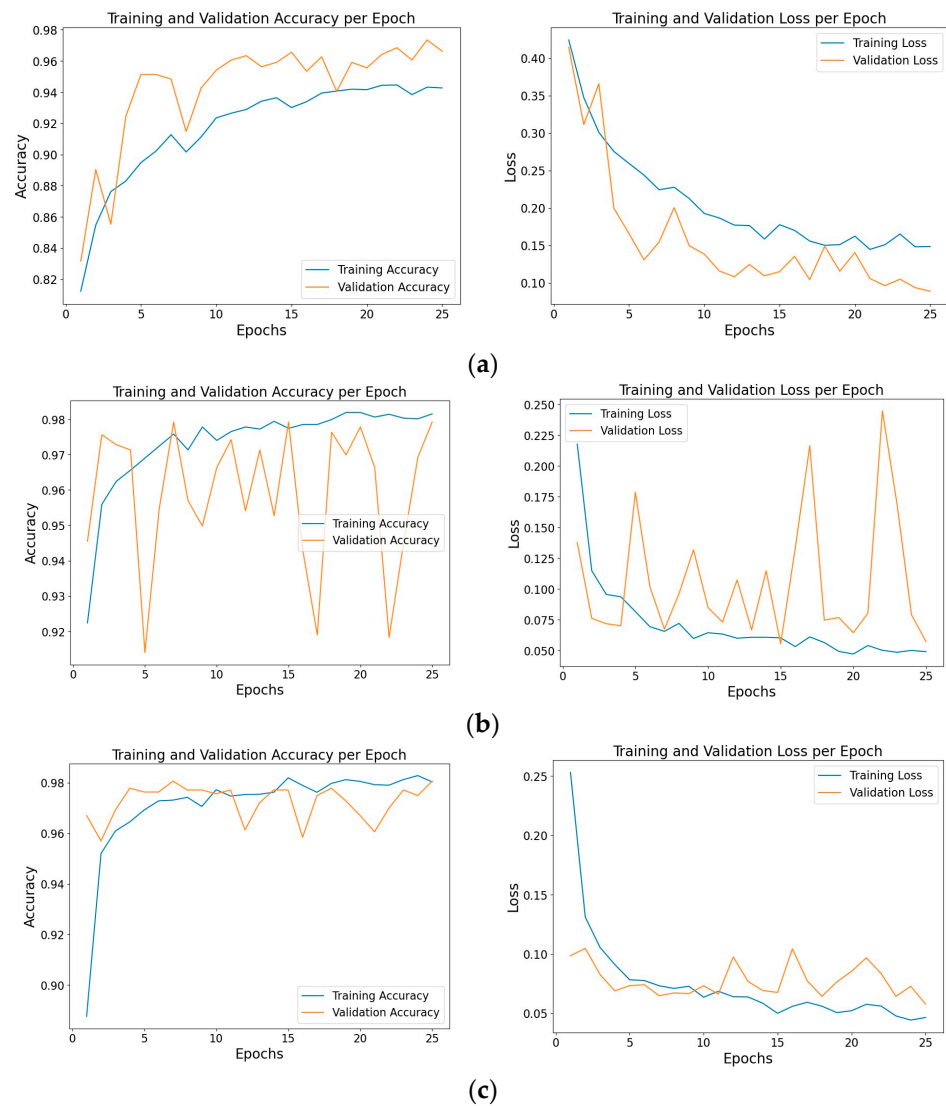
**Figure 3.** Comparative performance analysis of CNN (a), MobileNet (b), and optimized MobileNet (c) models across training epochs.

Figure 4 illustrates a comparison between the actual and predicted classifications for a chest X-ray labeled as “NORMAL”. The left panel displays the original chest X-ray image, showcasing a clear lung structure with no visible abnormalities indicative of a healthy condition. The right panel presents the corresponding predicted result from the model, overlaid with a Grad-CAM (gradient-weighted class activation mapping) heatmap. This heatmap highlights the regions of the image that the model considered most influential in making its classification decision. The areas in warmer colors (yellow, red) denote the regions with higher relevance to the model’s decision-making process, while cooler colors (blue, green) indicate regions of lower importance. The model correctly classifies the image as “NORMAL”, as evidenced by the alignment of the heatmap’s focus on relevant areas, such as the lung fields, confirming the model’s interpretability and the accuracy of its classification.

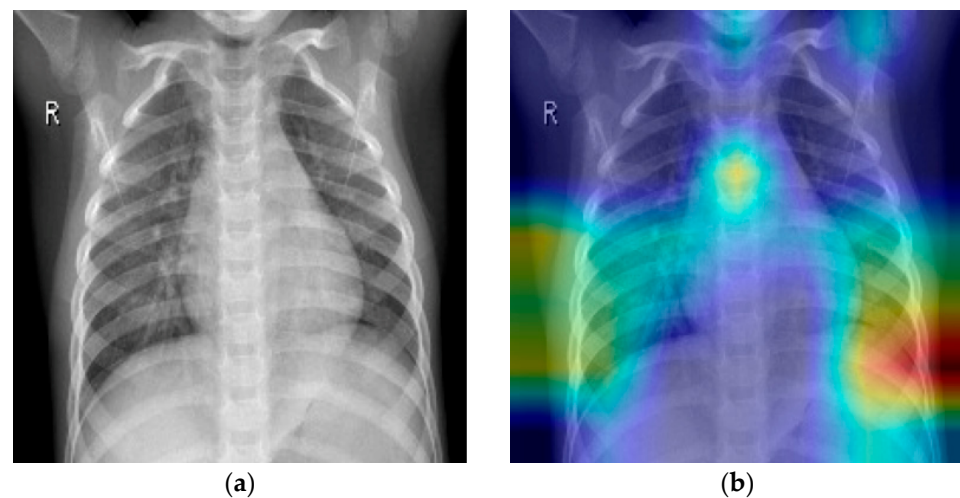


Figure 4. Comparison of actual chest X-ray NORMAL (a) and model prediction with Grad-CAM heatmap highlighting key regions influencing the classification (b).

The references selected for this research led us to the conclusion that these articles produced subpar results. For example, one study utilized a U-Net architecture and transfer learning for CXR picture classification, achieving a mere 93.06% accuracy, which is far lower than our own methods. Another study employing a fuzzy-enhanced deep learning technique known as CovNNet achieved an accuracy of 80.9%. This information is shown in Table 4. This comparison demonstrates our models’ superior ability to accurately differentiate pneumonia cases using imaging data. It also confirms the benefits of our advanced machine learning techniques and optimization strategies by showing that other studies report findings that have a lesser impact on scientific contribution. Figure 3 shows the result of our system’s prediction.

Table 4. Comparison of AI methods for chest X-ray detection.

Ref.	Method	Accuracy	Precision	Recall	F1-Score
[27]	Trained output-base transfer learning model	96.82%	95.97%	97.54%	96.74%
[28]	U-Net architecture and transfer learning	93.06%	88.97%	96.78%	92.71%

Table 4. Cont.

Ref.	Method	Accuracy	Precision	Recall	F1-Score
[29]	fuzzy-enhanced deep learning method called CovNNet	80.9%	85.2%	82.5%	-
[33]	ANN with ABC for region-based fractal analysis	92.37%	-	-	-
[34]	Combines handcrafted, radiomic, deep features	94.1%	94.5%	94.1%	94.0%
Proposed	hybrid model combining MobileNet with attention mechanism	$98.19 \pm 0.94\%$	$98 \pm 0.99\%$	$98 \pm 0.99\%$	$98 \pm 0.99\%$

In addition to presenting statistical results such as algorithm recognition rates, we conducted a comprehensive exploratory analysis of the diagnostic data utilized in our study. This analysis provides insights into the overall distribution and characteristics of the X-ray images, including both NORMAL and PNEUMONIA cases. By examining the entire dataset rather than selected samples, we aim to offer a more intuitive understanding of the diagnostic challenges and the algorithm's performance across diverse cases. Furthermore, we discuss the intrinsic limitations of our model, which prevent a 100% recognition rate. Factors such as image quality, variations in disease presentation, and inherent ambiguities in medical imaging data contribute to these limitations. These aspects underscore the complexities of medical diagnosis and highlight why certain images remain challenging for automated systems to interpret correctly. This discussion is crucial for contextualizing the capabilities and boundaries of computer-assisted diagnostic systems in real-world medical settings.

6. Conclusions

Previous works within the field that have explored detailed learning experiences of a deep convolutional neural network, CNN, and MobileNet models for pneumonia recognition showcased a significant performance in classifying X-ray images labeled as "NORMAL" or "PNEUMONIA". Not only are they highly accurate, but they are also very time-efficient in medical diagnostics, which is crucial during health crises like the COVID-19 pandemic. CNN model: Our CNN showed a good result, and it gave an overall accuracy of 95.35%, which is high in comparison with most works, especially having balanced precision, recall, and F1-scores on both classes (NORMAL, PNEUMONIA). These results were further boosted by a MobileNet architecture to 97.94% accuracy, and the hybrid optimization method was able to push this number up slightly more, reaching an accuracy of 98.19%. These results provide confirmation of the models, as well as demonstrate that deep learning and optimization techniques may be utilized to improve diagnostic accuracy even more. Additionally, the utilization of a range of optimization algorithms, e.g., dung beetle optimization, as well as utilizing Fick's law, was incorporated in the process to enhance and explain further algorithmic efficiency patterns assumed within digital image processing frameworks. These strategies have expanded the field of AI in healthcare, providing some innovative measures to approach complicated diagnostics.

This work demonstrates the power of using state-of-the-art neural nets in the analysis of medical images and emphasizes the importance of using innovative AI technologies to enhance the diagnostic process, which is crucial for timely disease identification. In the future, this research may look at expanding on the dataset using multimodal data and how we can supplement metaheuristic algorithmic changes to even further extend the possibilities of AI capabilities in healthcare.

One limitation of the current study lies in its reliance on a single modality of data—chest X-ray images. While the results indicate that the proposed hybrid optimization framework improves pneumonia detection, its applicability to other pathological diagnoses using different medical imaging techniques remains unexplored. Furthermore, the dataset

size, although sufficient for this task, may not fully capture the variability seen in larger and more diverse populations, potentially limiting the model's generalizability to different patient groups. Future research should focus on expanding this approach to multimodal data, integrating CT scans, MRI, or even clinical records and genetic data to enhance the robustness and accuracy of diagnoses across a broader spectrum of diseases. Additionally, exploring the application of this framework to other critical diseases, such as tuberculosis or lung cancer, could provide valuable insights into its versatility. Further work on optimizing computational efficiency and testing the model in real-world clinical settings would also be beneficial.

Author Contributions: Conceptualization, H.K.; methodology, H.K. and A.M.S.; software, A.M.S.; validation, H.K.; formal analysis, H.K.; investigation, A.M.S.; resources, H.K. and A.M.S.; data curation, A.M.S.; writing—original draft preparation, A.M.S.; writing—review and editing, H.K. and A.M.S.; visualization, H.K.; supervision, H.K.; project administration, H.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: <https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia>. accessed on 12 December 2023.

Conflicts of Interest: The authors have no conflicts to disclose.

References

1. Rønn, C.; Sivapalan, P.; Eklöf, J.; Kamstrup, P.; Biering-Sørensen, T.; Bonnesen, B.; Harboe, Z.B.; Browatzki, A.; Kjærgaard, J.L.; Meyer, C.N. Hospitalization for Chronic Obstructive Pulmonary Disease and Pneumonia: Association with the Dose of Inhaled Corticosteroids. A Nation-Wide Cohort Study of 52,100 Outpatients. *Clin. Microbiol. Infect.* **2023**, *29*, 523–529. [CrossRef] [PubMed]
2. Cillóniz, C.; Torres, A.; Niederman, M.S. Management of Pneumonia in Critically Ill Patients. *BMJ* **2021**, *375*, e065871. [CrossRef] [PubMed]
3. Ferreira-Coimbra, J.; Sarda, C.; Rello, J. Burden of Community-Acquired Pneumonia and Unmet Clinical Needs. *Adv. Ther.* **2020**, *37*, 1302–1318. [CrossRef] [PubMed]
4. Torres, F.A.; Orio, P.; Escobar, M.-J. Selection of Stimulus Parameters for Enhancing Slow Wave Sleep Events with a Neural-Field Theory Thalamocortical Model. *PLoS Comput. Biol.* **2021**, *17*, e1008758. [CrossRef] [PubMed]
5. Puttagunta, M.; Ravi, S. Medical Image Analysis Based on Deep Learning Approach. *Multimed. Tools Appl.* **2021**, *80*, 24365–24398. [CrossRef]
6. Suganyadevi, S.; Seethalakshmi, V.; Balasamy, K. A Review on Deep Learning in Medical Image Analysis. *Int. J. Multimed. Inf. Retr.* **2022**, *11*, 19–38. [CrossRef]
7. Deo, R.C. Machine Learning in Medicine: Will This Time Be Different? *Circulation* **2020**, *142*, 1521–1523. [CrossRef]
8. Funahashi, K. Big Data in Health Care—Predicting Your Future Health. *S. Cal. L. Rev.* **2020**, *94*, 355.
9. Too, J.; Liang, G.X.; Chen, H.L. Memory-Based Harris Hawk Optimization with Learning Agents: A Feature Selection Approach. *Eng. Comput.* **2022**, *38*, 4457–4478. [CrossRef]
10. Zhao, S.W.; Wang, P.J.; Heidari, A.A.; Zhao, X.; Ma, C.; Chen, H.L. An Enhanced Cauchy Mutation Grasshopper Optimization with Trigonometric Substitution: Engineering Design and Feature Selection. *Eng. Comput.* **2023**, *38*, 4583–4616. [CrossRef]
11. Hussien, A.G.; Hassanien, A.E.; Houssein, E.H.; Bhattacharyya, S.; Amin, M. S-Shaped Binary Whale Optimization Algorithm for Feature Selection. In *Recent Trends in Signal and Image Processing: ISSIP 2017*; Springer: Singapore, 2019; pp. 79–87.
12. Taradeh, M.; Mafarja, M.; Heidari, A.A.; Faris, H.; Aljarah, I.; Mirjalili, S.; Fujita, H. An Evolutionary Gravitational Search-Based Feature Selection. *Inf. Sci.* **2019**, *497*, 219–239. [CrossRef]
13. Ghosh, M.; Guha, R.; Sarkar, R.; Abraham, A. A Wrapper-Filter Feature Selection Technique Based on Ant Colony Optimization. *Neural Comput. Appl.* **2020**, *32*, 7839–7857. [CrossRef]
14. Zheng, Y.; Li, Y.; Wang, G.; Chen, Y.; Xu, Q.; Fan, J.; Cui, X. A Novel Hybrid Algorithm for Feature Selection Based on Whale Optimization Algorithm. *IEEE Access* **2019**, *7*, 14908–14923. [CrossRef]
15. Mohamed, A.A.; Hassan, S.A.; Hemeida, A.M.; Alkhalaf, S.; Mahmoud, M.M.; Eldin, A.M. Parasitism—Predation Algorithm (Ppa): A Novel Approach for Feature Selection. *Ain Shams Eng. J.* **2020**, *11*, 293–308. [CrossRef]
16. Cao, B.; Li, M.; Liu, X.; Zhao, J.; Cao, W.; Lv, Z. Many-Objective Deployment Optimization for a Drone-Assisted Camera Network. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 2756–2764. [CrossRef]
17. Zhang, M.; Chen, Y.; Susilo, W. Ppo-Cpq: A Privacy-Preserving Optimization of Clinical Pathway Query for e-Healthcare Systems. *IEEE Internet Things J.* **2020**, *7*, 10660–10672. [CrossRef]
18. Chao, M.; Kai, C.; Zhang, Z. Research on Tobacco Foreign Body Detection Device Based on Machine Vision. *Trans. Inst. Meas. Control* **2020**, *42*, 2857–2871. [CrossRef]

19. Zhang, M.; Wu, Q.; Chen, H.; Heidari, A.A.; Cai, Z.; Li, J.; Abdelrahim, E.M.; Mansour, R.F. Whale Optimization with Random Contraction and Rosenbrock Method for COVID-19 Disease Prediction. *Biomed. Signal Process. Control* **2023**, *83*, 104638. [[CrossRef](#)]
20. Sun, Y.; Chen, Y. Multi-Population Improved Whale Optimization Algorithm for High Dimensional Optimization. *Appl. Soft Comput.* **2021**, *112*, 107854. [[CrossRef](#)]
21. Chen, H.; Yang, C.; Heidari, A.A.; Zhao, X. An Efficient Double Adaptive Random Spare Reinforced Whale Optimization Algorithm. *Expert Syst. Appl.* **2020**, *154*, 113018. [[CrossRef](#)]
22. Kim, C.; Batra, R.; Chen, L.; Tran, H.; Ramprasad, R. Polymer Design Using Genetic Algorithm and Machine Learning. *Comput. Mater. Sci.* **2021**, *186*, 110067. [[CrossRef](#)]
23. Ahmad, M.F.; Isa, N.A.M.; Lim, W.H.; Ang, K.M. Differential Evolution: A Recent Review Based on State-of-the-Art Works. *Alex. Eng. J.* **2022**, *61*, 3831–3872. [[CrossRef](#)]
24. Sarfi, A.M.; Karimpour, Z.; Chaudhary, M.; Khalid, N.M.; Ravanelli, M.; Mudur, S.; Belilovsky, E. Simulated Annealing in Early Layers Leads to Better Generalization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 20205–20214.
25. Gad, A.G. Particle Swarm Optimization Algorithm and Its Applications: A Systematic Review. *Arch. Comput. Methods Eng.* **2022**, *29*, 2531–2561. [[CrossRef](#)]
26. Yi, R.; Tang, L.; Tian, Y.; Liu, J.; Wu, Z. Identification and Classification of Pneumonia Disease Using a Deep Learning-Based Intelligent Computational Framework. *Neural Comput. Appl.* **2023**, *35*, 14473–14486. [[CrossRef](#)] [[PubMed](#)]
27. Kumar, S.; Mallik, A. COVID-19 Detection from Chest x-Rays Using Trained Output Based Transfer Learning Approach. *Neural Process. Lett.* **2023**, *55*, 2405–2428. [[CrossRef](#)] [[PubMed](#)]
28. Manickam, A.; Jiang, J.; Zhou, Y.; Sagar, A.; Soundrapandiyan, R.; Dinesh Jackson Samuel, R. Automated Pneumonia Detection on Chest X-ray Images: A Deep Learning Approach with Different Optimizers and Transfer Learning Architectures. *Meas. J. Int. Meas. Confed.* **2021**, *184*, 109953. [[CrossRef](#)]
29. Ieracitano, C.; Mammone, N.; Versaci, M.; Varone, G.; Ali, A.R.; Armentano, A.; Calabrese, G.; Ferrarelli, A.; Turano, L.; Tebala, C.; et al. A Fuzzy-Enhanced Deep Learning Approach for Early Detection of COVID-19 Pneumonia from Portable Chest X-ray Images. *Neurocomputing* **2022**, *481*, 202–215. [[CrossRef](#)] [[PubMed](#)]
30. Alqudah, A.M.; Qazan, S.; Masad, I.S. Artificial Intelligence Framework for Efficient Detection and Classification of Pneumonia Using Chest Radiography Images. *J. Med. Biol. Eng.* **2021**, *41*, 599–609. [[CrossRef](#)]
31. Szepesi, P.; Szilágyi, L. Detection of Pneumonia Using Convolutional Neural Networks and Deep Learning. *Biocybern. Biomed. Eng.* **2022**, *42*, 1012–1022. [[CrossRef](#)]
32. Mannepalli, D.P.; Namdeo, V. A Cad System Design Based on HybridMultiscale Convolutional Mantaray Network for Pneumonia Diagnosis. *Multimed. Tools Appl.* **2022**, *81*, 12857–12881. [[CrossRef](#)]
33. Punitha, S.; Stephan, T.; Kannan, R.; Mahmud, M.; Kaiser, M.S.; Belhaouari, S.B. Detecting COVID-19 from Lung Computed Tomography Images: A Swarm Optimized Artificial Neural Network Approach. *IEEE Access* **2023**, *11*, 12378–12393. [[CrossRef](#)]
34. Ho, T.K.K.; Gwak, J. Feature-Level Ensemble Approach for COVID-19 Detection Using Chest X-ray Images. *PLoS ONE* **2022**, *17*, e0268430. [[CrossRef](#)] [[PubMed](#)]
35. Trivedi, M.; Gupta, A. A Lightweight Deep Learning Architecture for the Automatic Detection of Pneumonia Using Chest X-Ray Images. *Multimed. Tools Appl.* **2022**, *81*, 5515–5536. [[CrossRef](#)] [[PubMed](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.