Frontiers of Modern Computer Science Research

Academixs Open Access Publishers

Research Article

How to Implement Image Classification Techniques in Computer Vision

Muhammad Asif Nawaz*

Senior Software Engineer, UK

***Corresponding Author:**
Muhammad Asif Nawaz, Senior Software Engineer, UK.

Received Date: 27.02.2026 **Accepted Date:** 11.03.2026 **Published Date:** 15.03.2026

Abstract

In the review of this article, we are evaluating multi-label classification using the deep learning models, comparing four models in this research, such as CNNs, InceptionV3, VGG16, and ResNet50. So, we are developing this model's architecture with fewer convolutional layers. Originally, chest X-ray dataset got from Kaggle, in the dataset I have 11 million data points, but we are giving only 7936 datapoints and images (due to the resources of local machine), in the data preprocessing, we are splitting the dataset into three sub-set, test, validation and training randomly at ratio of 60%, 20% and 20%. We got the best model is InceptionV3, having 12% accuracy on positive datapoints. Using the data augmentation technique with lots of functions like rotation, shear, zoom, width & height shift range, and flipping sample-wise normalization as well.

Keywords: Deep Learning, Image Classification, Convolutional Neural Networks (CNN), Machine Learning, Multi-label classification, Computer Vision

Introduction

This article's main goal is to create an image classification system using machine learning to help with the diagnosis of various diseases, including pneumonia, cardiomegaly, emphysema, diffusion, mass, and other conditions that are scanned on chest X-rays. Convolutional neural networks (CNNs) are utilized by the system to extract features, taking advantage of their capacity to capture spatial hierarchies of images by means of multiple convolutional and pooling layers. Using transfer learning, pre-trained models like VGG16, ResNet50, and InceptionV3 are compared and optimized for best performance alongside CNN architecture. Deep learning is a subset of machine learning, which classifies tasks across multiple domains have been transformed. Convolutional neural networks have become the standard architecture in medical diagnosis because of their capacity to extract features from images. Multiple layers of convolutional and pooling operations make up CNNs, which enable the network to recognize complex patterns of images. Deep learning is especially useful for image recognition in machine learning via different media.

This statement is according to Igor Kononenko who is "Medical datasets were the initial source of design and application for machine learning algorithms [1]. These days, machine learning offers a number of essential instruments for astute data analysis. The digital revolution has made it relatively cheap and accessible to gather and store data, especially in the last few years. Large information systems collect and share data, and modern hospitals are well-equipped with monitoring and other data collection tools. Medical data analysis is currently a good fit for machine learning technology, and in particular, a lot of work has been done in the area of medical diagnosis in small, specialized diagnostic problems."

Data preprocessing techniques, including image augmentation (rotation, scaling, flipping, and color adjustments), are applied to improve the robustness of the model. The model performance is evaluated using metrics like precision, f1-score, accuracy, recall, and confusion matrix, providing a comprehensive assessment of its effectiveness. CNN-based methods have various strategies to increase the performance of image classification on datasets [2]. Oone method used in the article is data augmentation via imageDataGenerator with configurations.

Convolution neural networks are obtained by stacking one or more computational layers, the traditional transform-based data augmentation has better performance than generative adversarial network and other neural network-based methods. Deep learning needs a huge amount of data to get better results. Especially on medical problems to get and annotating the data is very important and timeconsuming process. There are some solutions to solve the problem, one already little bit discuss on

above is data augmentation which is prevent the overfitting and improves the accuracy on training data and another method for boosting the performance of in deep learning particularly CNNs which is called transfer learning. So, in this article, we will be utilizing a data augmentation method.

Background and Context

The article employs CNN and pretrained models to enhance chest X-ray image classification for medical diagnosis. By leveraging CNN ability to extract complex patterns, the study aims to improve diagnostic accuracy. Data augmentation and deep learning techniques are applied to boost model performance and address key challenges in medical image analysis.

Problem Statement

For thoracic diseases to be effectively treated and patient outcomes to be improved, an early and accurate diagnosis is essential. The most widely used imaging technique for diagnosing diseases like Atelectasis, Consolidation, and Infiltration is a chest X-ray. Nevertheless, the interpretation of these images is difficult and frequently arbitrary, greatly depending on radiologists' experience. This may result in inconsistent diagnoses, particularly in settings with limited resources where skilled radiologists may not be available. Radiologists are under a lot of pressure due to the growing number of chest X-rays that require analysis, which increases the risk of diagnostic mistakes and delays. Furthermore, a lot of thoracic diseases share similar visual features and symptoms, making it challenging to differentiate between them just by manual interpretation. This intricacy highlights the requirement for an automated, trustworthy instrument that can help with the precise categorization of thoracic diseases.

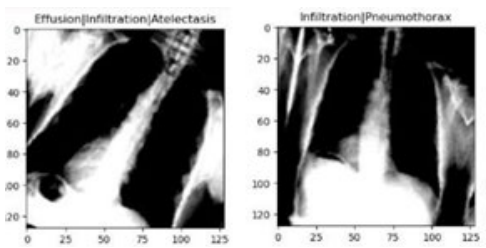
Research Rationale

Is it possible to create a multi-label classification model using deep learning that can effectively identify and categorize various medical conditions from chest X-ray images, thereby enhancing diagnostic accuracy/precision and enhancing patient outcomes?

- to create a deep learning model that can identify various medical conditions from chest X-ray images.
- To assess the suggested model's effectiveness in terms of recall, accuracy, precision, F1-score, and confusion matrices
- to assess how well the suggested model performs in comparison to current CAD (computer-aided diagnosis technology) systems and radiologists.

Scope and Objectives

Develop a robust and accurate deep learning model to automatically classify medical diseases from chest Xray images. The model should be able to detect multiple diseases from a single X-ray image, providing a reliable tool that can assist in medically diagnosing conditions such as Atelectasis, Consolidation, Infiltration, Pneumothorax, and other Hernia.



For multi-disease detection from single X-ray image: this objective requires the models to handle cases where multiple diseases might be present simultaneously in a single X-ray image. For example, a patient could have Infiltration and Hernia, and models should correctly predict both conditions. The Scope of the article will primarily be utilizing chest X-ray dataset which is a large collection of Xray images labelled with 14 various diseases and added extra column "File Path" in dataset for append path of the images in that column based on the Image Index column. I'm exploring the dataset to understand its structure, labels distribution, and image characteristics. This includes identifying missing data or inconsistencies with the dataset.

Achievements

We evaluated four models by using deep learning convolutional layers, as defined in methodology. Deep learning models are capable of accurately classifying multiple thoracic diseases from chest xray images. Models effectively address complex challenges such as multi-label classification, class imbalance, and the high dimensionality of medical images. Implemented techniques such as class weighting, data augmentation, and sample weighting, which ensured that the model performed well across both common and rare thoracic diseases. The best model with the highest accuracy among the four models is displayed overall in this article. Showing the results for the best model which is accuracy, recall, F1-score, precision, and classification report as well, showing the result for label support. As well as showing the confusion matrices for all predicted labels, and at the end deploy that model for integration.

Overview

All predicted models adhered to the same format. Like how to construct the first method that searches for the optimal parameters across batches and epochs, another method searches for the optimal parameters across learning rate, initialization mode, activation for dense layers, and dropout rate for the optimizer. Thus, as I indicated, there are two ways to look for the ideal parameters for a given model. Prior to fitting the training data to the model, the best fit will be assigned to a particular model and that model will be predicted on unseen data. Subsequently, the model is fitted using training data, and a report on model history visualization is displayed. After fitting the model and displaying the Roc report for the same data before and after training, predict the model once more on unseen data. The same processes are then taken for the other models, until the best accuracy model is obtained. The classification report for that model is then displayed, along with the confusion metrics, before the model is saved and deployed for use. So, the conclusion of the overview is that building this project for multi labels classification in medical field. It will overcome burden of radiologists.

State of the Art

Develop a multi-label deep learning system to detect multiple diseases from chest X-ray images using a structured workflow that includes data preparation, augmentation, model building, and evaluation. Various data augmentation techniques such as rotation, zoom, shear, brightness adjustments, and flipping, are applied to enhance robustness and reduce overfitting. Hyperparameters are optimized using randomized grid search with cross-validation, tuning components like batch size,

epochs, learning rate, optimizer, activation function, and dropout rate. Multiple models are trained and compared based on test accuracy, loss, and ROC performance. The best model is further evaluated using precision, recall, F1score, classification reports, and confusion matrices to ensure reliability. Ultimately, the highestperformance model is saved for deployment, supporting its integration into an X-ray diagnostic application.

So, the statement according to Yadav and Jadhav is that, in order to improve the performance of image classification on small datasets, this research uses a variety of machine learning techniques, including methods based on convolutional neural networks and multiple strategies [3]. One approach is data augmentation, which performs better than neural network- and generative adversarial network (GAN)based techniques. Data augmentation is an alternative to traditional transform-based methods. Transfer learning is an additional technique that, according to Kermany research, produced 92% accuracy on a small dataset of pneumonia X-ray images. But there are a few gaps that should be noted. One drawback of Kermany's work is that they employ the InceptionV3 model, but they don't retrain the InceptionV3 convolutional layer because of overfitting. Thus, this study will assess alternative models as well as the results of retraining the convolutional layer. Furthermore, did not contrast the capsule network's performance with that of other techniques. Thus, the following are the contributions of this report.

performance comparison of three distinct classification methods: training a capsule network from scratch, transfer learning of VGG16 and InceptionV3, and an SVM classifier with oriented fast and rotated binary robust independent elementary features (ORB).

An examination of how the CNN transfer learning method performed on the small chest X-ray dataset to determine how data augmentation, network complexity, optimized convolutional layers, and other anti-overfitting mechanisms affected the classification.

Methodology

This deep learning article involves chest X-ray image classification using models such as CNNs, VGG16, InceptionV3, and ResNet50, specifically targeting medical imaging data based on disease labels. Here we are dealing with several key components and methodologies, including data preprocessing, data augmentation, searching best parameters for models, model architecture definition, training process, and performance evaluation.

I will be utilizing 'NumPy', 'pandas', 'seaborn' and matplotlib for data manipulation, analysis, and visualization. Importing various modules like 'scikitlearn' for model selection, evaluation and preprocessing as well. 'TensorFlow' and 'Keras' for building and training deep learning models, imageDataGenerator for image augmentation and normalization. For the model evaluation, utilizing matrices ROC-AUC, F1 score, precision, recall and confusion matrices to evaluate model performance.

Data Understanding

I have all the necessary data organized in one location, with a CSV file named 'chest_X_ray_data' and an accompanying images folder. The CSV file contains columns for the (Image Index, Patient ID, Atelectasis, Consolidation, Infiltration, Pneumothorax, Edema, Emphysema, Fibrosis, Effusion, Pneumonia, Pleural thickening, Cardiomegaly, Nodule, Mass, and Hernia) labels, which will serve as features for evaluation except image index and patient id. I plan to enhance this data frame by adding a new column for the file path of each corresponding image index.

Data Preparation

Data preparation is the process of preparing data that is suitable for further processing and analysis on models. Key steps include collecting data, cleaning, and extracting features which is suitable for machine learning algorithms.

Loading the dataset from CSV file named 'chest_X_ray_data' from the directory. The nrows parameter is set to selected number of rows to read from the file. So, the resulting dataframe is stored in train_df_main.

Drop the column named 'No Finding' fom train_df_main and extract all predicted labels which is use for image classification. Utilizing the 'Glob' library for getting the image file from the directory so, the list of image paths is stored in the 'image_paths' variable.

Split the dataset in various part train_set, valid_set and test_set. The train_test_split function is used to split the data into training, validation and testing sets with a test size of 0.2. The 'random_state' parameter is set to the RANDOM_STATE variable.

Added 'FilePath' column in all datasets based on the Image index because we have image name and image Index value is same in file and directory. Also, remove the duplicate rows from the split datasets

Data Augmentation

The process of artificially creating new data from preexisting data, known as data augmentation. Modern machine learning models are very powerful, so this is a crucial step in the dataset building process. If these models are given too small datasets, they may begin to "overfit," which is a problem where the models simply memorize the mappings between their inputs and expected outputs.

In the code, I have two functions for data augmentation: one 'get_train_generator' is giving augmented generator for the training dataset, and other 'get_test_and_valid_generator' is giving augmented generator for the validation and testing datasets. One of the most important things, generating augmented data based on the flow_from_datafram here is the couple of configurations that I'm using in ImageDataGenerator for training data.

'samplewise_center = true': utilizing the center option each image by subtracting the mean pixel value from each pixel. So rather than doing this for whole dataset, it is done for each individual image. ❖ 'samplewise_std_normalization= true': Utilizing this option for standardization of images is achieved by subtracting the mean pixel value and dividing the result by the standard deviation of the pixel values.

'rotation_range= 32': this parameter specifies the maximum rotation angle (in degrees) that will be applied. In this project

I'm using maximum rotation angle is 32 degrees.

'shear_range= 0.1': parameter will apply a random shear transformation on the images. So, the maximum angle in radians that will be applied. Here, the maximum shear angle is 0.1 radians. ❖ 'zoom range= 0.15': parameter will apply a random zoom transformation on the images. So, the parameter specifies the maximum zoom factor that will be applied. In this case, the maximum zoom factor is 0.15, which means that the image will be zoomed in or out by up to 15%.

'width_shift_range=0.1': parameter specifies the maximum horizontal shift as a fraction of the image width that will be applied. In this case, the maximum horizontal shift is 0.1, which means that the image will be shifted by up to 10% width.

'height_shhift_range=0.05: parameter specifies the maximum vertical shift as fraction of the image height that will be applied. In this case, the maximum vertical shift is 0.05, which means that the image will be shifted by up to 5% height.

'rescale=1. /255': So, the rescale parameter specifies the scaling factor that will be applied on the pixel values.

'brightness range= (0.8, 1.2): this parameter specifies the minimum and maximum brightness multiplier that will be applied. In this case, the brightness will be adjusted by a factor between 0.8 and 1.2.

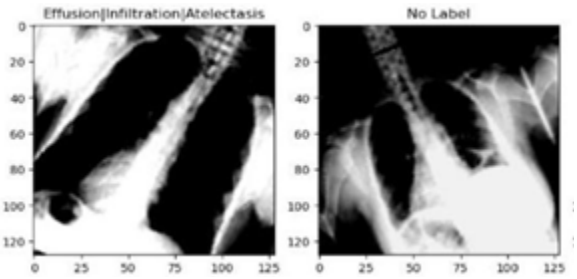
'horizontal flip=True: parameter applies a random horizontal flip to the images. If True, the image will be flipped horizontally with a probability of 0.5

'vertical flip=False: this parameter applies a random vertical flip to the images. So, in the case vertical flipping is disabled.

'fill mode=reflect': parameter mean image will be reflected at the boundary.

Data Visualization

Data visualization is the process of graphical representation of data, at this stage we have train, valid and test generator batch dataset with various configurations by the image data generator. For the multi-classification of an image that will show a single X-ray image having multiple diseases, which will predict at the end of evaluation. So, this figure shows the labels on each image that will represent diseases on that X-ray image. For example, a patient could have an Infiltration and an Effusion, and the model should correctly predict both conditions.



So, this image visualization develops a better understanding of the relationship between the images and their labels that can ultimately lead to better model performance and more accurate predictions. This plot shows us how the prevalence of positive cases varies greatly amongst the various pathologies. (These trends also correspond with those found in the entire dataset.)

We can see the label 'Hernia' has the greatest imbalance with the proportion of positive training cases being about 0.25%. Same as if we see the label 'Infiltration', which has the least amount of imbalance has only 17% of the training positive cases.

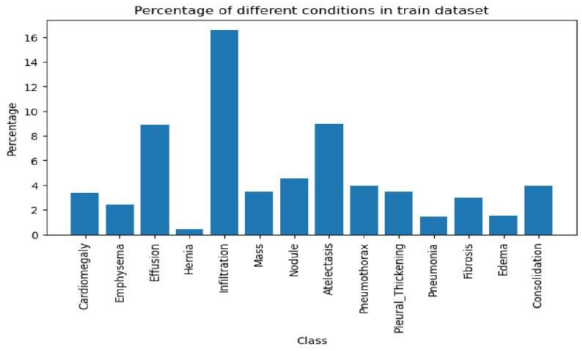


Figure 1: Percentage of Positive Labels

As we can see in the Figure 2, the contribution of the positive cases is significantly less than that of the negative cases in training dataset.

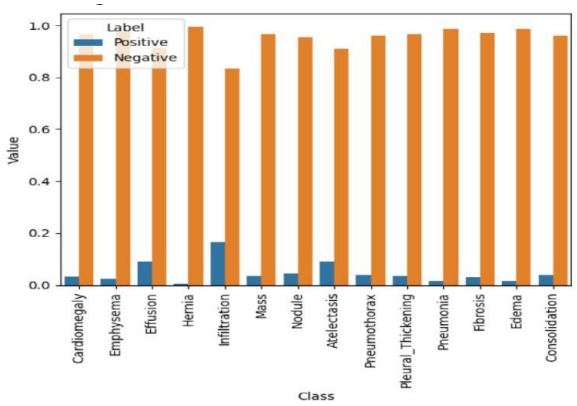


Figure 2: Percentage of Positive and Negative Labels

Model Building

Utilizing machine learning model convolutional neural Networks (CNNs) and pre-trained models such as (InceptionV3, VGG16, and ResNet50), ImageNet weights, excluding the top layers, typically involves various components like weights, including top and input shape.

Searching Best Parameters

Before fitting the model on the training and validation datasets, it's important to determine the best parameters using RandomizedSearchCV along with a Keras classifier. The following steps outline the process of finding the optimal parameters using this approach.

I created two methods for finding the best parameters on train generator dataset. The first method, named (get_batch_ parameter) feeding the model with default values in search the best batch size and epochs on one augmented batch, which is the default 32. Same as for the second searching method named (get_parameters_using_batch) feeding the model with default

values and batch size and epochs which got best from the first method. This method gives the best optimizer, learning rate, init mode, activation and dropout rate for that model.

We defined the array of all parameters and build dictionary for all parameters. Pass the keras model to the random search class as well as grid search dictionary and iteration step with cross-validation score.

CNN Architecture

I wrap this model in function having parameters with default values, and that parameters are using in function body.

Input Layer: Defined the sequential model then add the input layer with convolutional layer that has 64 filter, a 3*3 kernel size, and use the activation parameter with default value is relu and also add kernel initializer function with assign the default value of init_mode. So, the input shape matches the size of the images, and padding is assigning to same to preserve the spatial dimensions.

Conv2D Layers: Convolutional layers with 32, 64, and 128 filters are gradually added by the model. These layers capture various level of features, so from the low-level features like edges and textures in earlier layers to more complex patterns in deeper layers.

MaxPooling2D Layers: Pooling layers follow every set of convolutional layers, which overcomes the spatial dimensions and helping to generalize the learned features. This down sampling overcomes the computational load and helps prevent overfitting.

Dropout Layers: After every pooling layer, set the dropout layer with the parameter drop-out_rate default value is 0.5 where randomly deactivate neurons during training time, which is also help to overcomes overfitting.

Fully Connected Layer: The fully connected layer begins by flattening the output from the last convolutional layer into a one-dimensional vector. A dense layer with 1024 units and default ReLU activation follows, enabling the model to learn complex patterns from the extracted features. To mitigate overfitting, a dropout layer with a specified rate is applied after this dense layer. The output layer, designed for multi-label classification, has units equal to the number of labels and uses a sigmoid default activation function to generate probability scores for each label.

Compilation: For the CNN model compilation, binary_crossentropy is employed as the loss function, suitable for independent label predictions, with default Adam as the optimizer set to a 0.01 default learning rate, and accuracy as the metric to evaluate performance.

Pre-Trained Models Architecture

We are using three pre-trained models which is (RESNET50, VGG16 and InceptionV3) these models having the same architecture except base layer.

Every model wrapped in the individual function with parameters having default values, thus these default values are using at the time of best parameters searching. Every model will reinitialize after finding the best parameter values and then that values will be feed to specific model. Now defining the layers that will be common in for every model.

Input Layer: defined the input layer with the shape of image size is 128*128 and color channel is 3 (which is in the RGB).

Base Model: Defined the pre-trained model architecture, which is exclude the fully connected layers at the top of the model network, keeping only the convolutional base layer. Also initializes the base model with pre-trained weights with ImageNet dataset. So, at the end of this layer using inputs which is defined in the previous step.

Freezing Base Layer: Freez the layers of the base model which is prevent from being updated during training. This layer only to allow the model to use the pre-trained features.

Dense and Dropout Layers: dense layers are fully connected which help the model to learn on complex patterns. So, for the dense layer, filters are gradually added in convolutional layers which is 64, 128 and 512, and for the output connected convolutional layers is 1024. Dropout rate is defined in parameter which is 0.5.

Fully Connected Layer: the first layer is flattening the output of the previous layer into a onedimensional vector. So, the forth layer as mentioned in connected layer is dense with the filter is 1024. Also added the dropout rate layer with default value of the dropout_rate parameter. Add the next layer is dense with units are equal to the length of the predicted labels with searched activation function.

Compilation: In the model compilation, utilizing the 'binary_crossentropy' loss which is suitable for multi-label classification where each label is treated independently. For the model default optimizer function using Adam with the default learning rate is 0.01. Next, last and third parameter is metrics which is accuracy to evaluate the model performance.

Model Evaluation

In the phase of the model evaluation, we rank the highest performing model according to accuracy. As previously mentioned, we store the accuracy and loss of each model in the corresponding model variables, create a new array based on the model's variables, and display a visualization graph for all models.

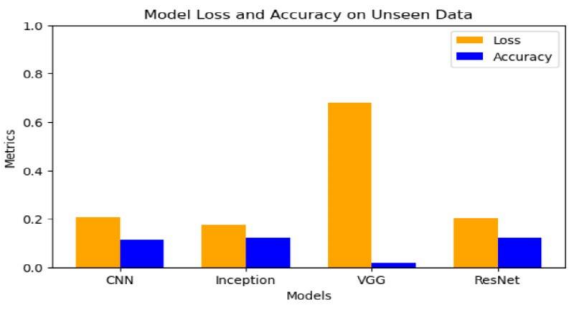


Figure 3: Models Comparison

Finally, the best model based on the accuracy is the InceptionV3

with 12% accuracy. Predicting that model on unseen data and getting the metrics like accuracy, F1-score, recall, precision, and classification report as well. We describe according to the best outputs of the model, True Positive, True Negative, False Positive, and False Negative are used to analyze and identify the performance of the model.

Here, four metrics are given as follows.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{FP + FN}$$

$$F1 = 2 \times \frac{precision}{precision + recall}$$

Also showing the confusion metric for all the predicted labels for the InceptionV3 model. So, the confusion matrix is a performance measurement for a machine learning classification problem, particularly for binary classification. We have a confusion matrix for each individual label.

- True Negative cases where the condition was correctly predicted.
- False Positive cases where the condition was incorrectly predicted.
- False Negative cases where the condition was incorrectly predicted.
- True Positive cases where the condition was correctly predicted.

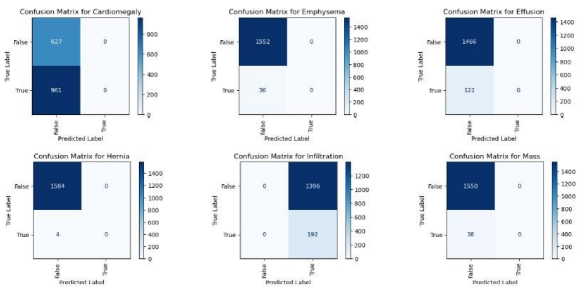


Figure 4: Confusion Matrices

Deployment

In the deployment phase, the best-performing model, identified as InceptionV3 with the highest accuracy, is saved for future use. This model is prepared for integration into a production environment where it can be utilized for real-time chest X-ray image classification. The deployment process includes finalizing the model's architecture, ensuring that it can efficiently handle unseen data, and exporting it in a format compatible with deployment platforms. The deployed model is expected to aid in medical diagnostics by accurately predicting multiple diseases from chest X-ray images.

Conclusion

This article focuses on the multi-label classification of chest X-ray images using deep learning models, aiming to predict multiple diseases from a single image. The methodology involves several key steps, including understanding the data, preparing the data, data augmentation, model building with hyperparameter tuning, and performance evaluation. Data augmentation techniques like rotation, shear, and flipping are employed to prevent overfitting and enhance model robustness.

This outlines a systematic approach to finding the optimal hyperparameters for the models. Two methods are used: one for searching for the best parameters across batches and epochs, and another for tuning learning rates, initialization modes, activation functions, and dropout rates. Once the best parameters are identified, they are applied to the models, which are then trained on the data. After training, the models are evaluated on unseen data, and performance is visualized through metrics like ROC curves. The best-performing model is then subjected to further evaluation, including the generation of confusion matrices, F1-score, recall, precision and classification reports.

Appendices

Appendix A: Visit this GitHub repository for access to the codebase, including implementation details, trained models, and experimental scripts.

References

1. Kononenko, I., & Kukar, M. (1995). Machine learning for medical diagnosis. Proceedings of the CADAM, 95.
2. Ding, J., Chen, B., Liu, H., & Huang, M. (2016). Convolutional neural network with data augmentation for SAR target recognition. *IEEE Geoscience and remote sensing letters, 13*(3), 364-368.
3. Yadav, S. S., & Jadhav, S. M. (2019). Deep convolutional neural network based medical image classification for disease diagnosis. *Journal of Big data, 6*(1), 1-18.

Citation: Muhammad Asif Nawaz., (2026). How to Implement Image Classification Techniques in Computer Vision. *Front Mod Comput Sci Res, 1*(1), 01-06.