Advanced Machine Learning
MSc Data Science – IHU

DEEP LEARNING PNEUMONIA DETECTION IN X-RAY IMAGES:

Alexiadis Vasileios *Douvikas Vasileios*

ABSTRACT

Pneumonia, a significant contributor to global mortality stemming from lung diseases, tragically claimed the lives of millions of individuals during the pandemic [1]. The outbreak of the Covid-19 pandemic has further intensified the urgency surrounding pneumonia diagnosis. However, the accurate identification of pneumonia in chest X-ray images remains an intricate puzzle, complicated by the scarcity of skilled radiologists within healthcare facilities. To address this pressing challenge, we present a harnessing of the power of ensemble modeling—a voting system composed of three distinct models—to effectively discriminate between three classes: absence of infection, bacterial pneumonia, and viral pneumonia in chest x-ray images. Our proposed model showcases an accuracy rate of 83.2%. The Pneumonia Classification Dataset, sourced from the Chest X-Ray Images (Pneumonia) dataset hosted on Kaggle .

1. DATA & PROBLEM DESCRIPTION

In the given dataset [2], the focus is on chest X-ray images for pneumonia classification. The dataset is sourced from the Chest X-Ray Images (Pneumonia) dataset. It consists of two main folders: "train" and "test." The "train" folder contains 4,672 JPG images categorized into three classes: normal x-ray images (1,227 samples), x-ray images with bacterial pneumonia (2,238 samples), and x-ray images with viral pneumonia (1,207 samples). Each image is associated with a class label provided in a corresponding CSV file, where class_id: 0 represents normal, class_id: 1 represents bacterial pneumonia, and class_id: 2 represents viral pneumonia.

The dataset also includes a separate "test" folder with 1,168 images for evaluating the model's performance. Notably, the images in the dataset are specifically selected from pediatric patients aged one to five years old.

The primary objective of this project is to accurately classify chest X-rays into their respective classes and achieve performance levels comparable to human experts. By developing a robust model, we aim to assist in the early diagnosis and treatment of pneumonia, ultimately leading to improved clinical outcomes.

To accomplish this, various data science techniques and machine learning algorithms can be employed. Convolutional neural networks (CNNs) have proven to be effective in image classification tasks [3]. Additionally, leveraging transfer learning, where pre-trained models are used as a starting point, can enhance the model's performance by leveraging existing knowledge and features.

Training the model on the provided dataset involves optimization techniques such as hyperparameter tuning and regularization to achieve accurate classification results. The model's performance can be evaluated using metrics such as validation loss and validation accuracy.

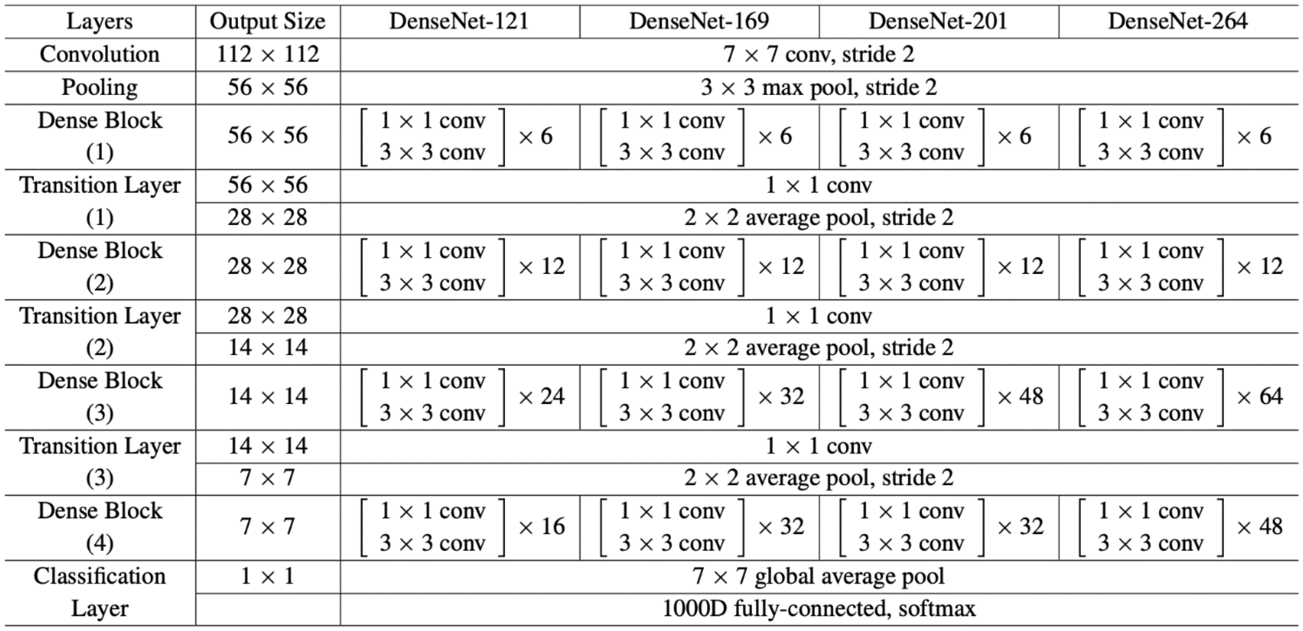
Furthermore, ethical considerations, data privacy, and responsible data science practices should be upheld throughout the project.

2. DESCRIPTION OF MODELS USED

DenseNet-121 is a popular convolutional neural network (CNN) architecture that has been widely used for various computer vision tasks, including image classification. It was introduced by Huang et al. in their paper "Densely Connected Convolutional Networks" published in 2017 [4]. DenseNet-121 is an extension of the DenseNet family, which aims to address some limitations of traditional CNN architectures such as vanishing gradients and the need for extensive parameter tuning.

Key features and details of DenseNet-121 include:

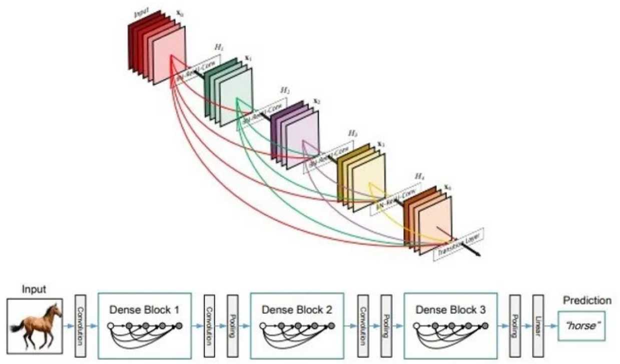
1. **Dense connectivity:** DenseNet-121 introduces the concept of dense connectivity, where each layer is directly connected to every other layer in a feed-forward fashion. This design choice promotes feature reuse and facilitates information flow throughout the network, enabling better gradient flow and alleviating the vanishing gradient problem.



Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112×112	7×7 conv, stride 2			
Pooling	56×56	3×3 max pool, stride 2			
Dense Block (1)	56×56	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	56×56	1×1 conv			
	28×28	2×2 average pool, stride 2			
Dense Block (2)	28×28	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	28×28	1×1 conv			
	14×14	2×2 average pool, stride 2			
Dense Block (3)	14×14	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 24$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 64$
Transition Layer (3)	14×14	1×1 conv			
	7×7	2×2 average pool, stride 2			
Dense Block (4)	7×7	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 16$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$
Classification Layer	1×1	7×7 global average pool			
		1000D fully-connected, softmax			

Table 1: DenseNet architectures for ImageNet

2. **Dense block:** DenseNet-121 is composed of several dense blocks, where each block consists of multiple layers that are densely connected to each other. Within each dense block, feature maps from all preceding layers are concatenated together, forming a dense set of inputs for subsequent layers. This dense connectivity pattern allows feature maps to have direct access to a rich collection of features from previous layers, aiding in information propagation and enhancing model performance.



convolutional layer and average pooling. They reduce the number of channels while maintaining spatial resolution, facilitating efficient information flow and reducing computational complexity.

4. **Global average pooling and classifier:** At the end of the network, a global average pooling layer is used to aggregate spatial information into a single feature vector. This feature vector is then fed into a fully connected layer with softmax activation for multi-class classification. The global average pooling operation enables DenseNet-121 to have a fixed-size feature representation regardless of input size, making it adaptable to varying input dimensions.

5. **Pre-training and transfer learning:** DenseNet-121, like other CNN architectures, can benefit from pre-training on large-scale datasets such as ImageNet. Pre-training enables the network to learn generic visual features, which can then be fine-tuned or transferred to specific tasks with limited labeled data.

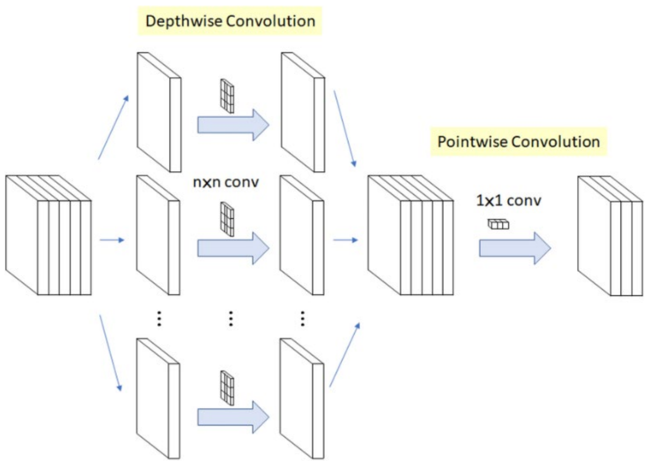
DenseNet-121 has demonstrated strong performance on various image classification benchmarks and medical imaging tasks, including pneumonia classification in chest X-ray images. Its dense connectivity and information flow enable it to capture fine-grained features, leading to robust representations and accurate predictions.

It is worth noting that the specific implementation and variations of DenseNet-121 may differ across frameworks and libraries. Researchers and practitioners often refer to the original paper [4] for a detailed understanding of the architecture, including layer configurations, hyperparameters, and implementation specifics.

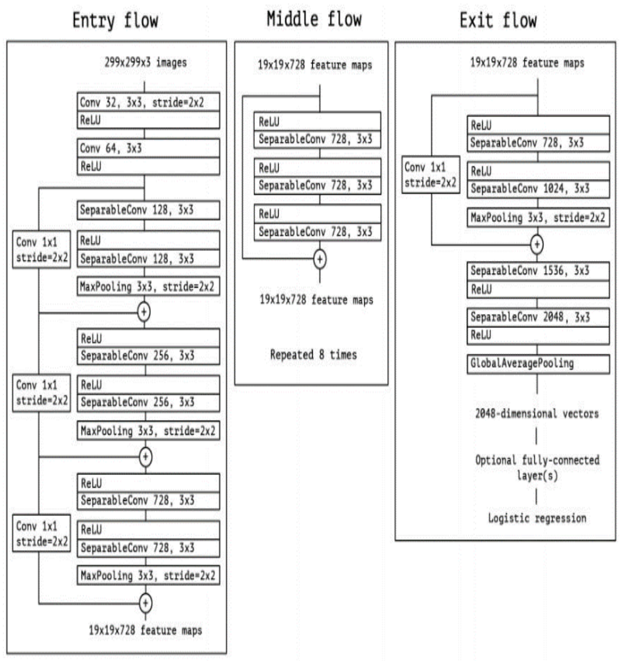
Xception is another widely used convolutional neural network (CNN) architecture that has gained popularity in the computer vision community. It was introduced by Chollet in the paper "Xception: Deep Learning with Depthwise Separable Convolutions" published in 2017 [5]. Xception is designed to improve the efficiency and performance of traditional CNNs by employing depthwise separable convolutions.

Here are some key features and details of the Xception architecture:

1. **Depthwise separable convolutions:** Xception utilizes depthwise separable convolutions as the fundamental building block. Unlike traditional convolutions, which perform convolution operations across all input channels, depthwise separable convolutions separate the spatial convolution (depthwise convolution) from the pointwise convolution (1x1 convolution). This separation significantly reduces the number of parameters and computational complexity, leading to a more efficient network.



2. **Extreme Inception module:** Xception's architecture is inspired by the Inception module, which is known for its ability to capture multi-scale information. However, Xception takes it a step further by replacing the traditional convolutional layers in the Inception module with depthwise separable convolutions. This modified module, known as the "Extreme Inception module," enables Xception to achieve both high accuracy and efficiency.



4. **Multiple stacked modules:** Xception consists of several stacked Extreme Inception modules, forming a deep network architecture. The number of modules and their configurations can be adjusted based on the specific task requirements. These modules enable Xception to learn hierarchical features at different scales and complexities, contributing to its strong performance in various image recognition tasks.

5. **Global average pooling and classifier:** Similar to other CNN architectures, Xception uses global average pooling to convert the output feature maps into a fixed-length feature vector. This vector is then fed into a fully connected layer with softmax activation for final classification.

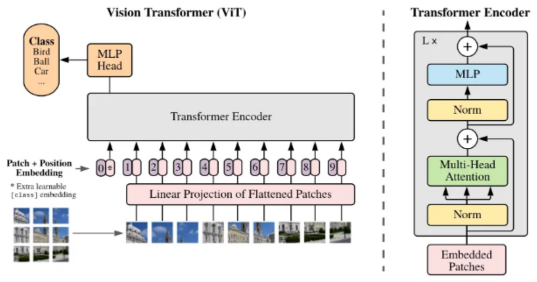
Xception has demonstrated impressive results (Table 2) on various image classification benchmarks and has been successfully applied to diverse computer vision tasks. It provides an efficient alternative to traditional CNN architectures, offering a good trade-off between accuracy and computational complexity.

Model	Top-1 Acc	Top-5 Acc
VGG-16	0.715	0.901
ResNet-152	0.770	0.933
Inception V3	0.782	0.941
Xception	**0.790**	**0.945**

Table 2 *Comparison in terms of accuracy of various models for Xception*

Visual transformers, also known as Vision Transformers or **ViTs**, have emerged as a powerful class of neural network architectures for computer vision tasks. Unlike traditional convolutional neural networks (CNNs) that rely on convolutional layers for feature extraction, visual transformers employ a self-attention mechanism to capture global dependencies and model long-range interactions within an image [6].

The self-attention mechanism in visual transformers allows the model to focus on different parts of the image by computing attention weights for each position in the input feature map. This enables the model to attend to relevant image regions while capturing relationships between all pairs of positions, facilitating the modeling of global dependencies.



To process images with visual transformers, the input image is divided into smaller non-overlapping patches. Each patch is treated as a separate token and undergoes linear embedding to obtain a fixed-dimensional representation. This patch-based representation allows visual transformers to handle images of arbitrary sizes and facilitates parallel processing of image patches.

Typically, they adopt a modified version of the Transformer encoder architecture. The encoder consists of multiple stacked layers, each containing a multi-head self-attention mechanism followed by feed-forward neural networks. Layer normalization and residual connections are applied to improve training and gradient flow within the network.

Incorporating positional information is crucial for visual transformers to reason about the relative positions of patches and capture spatial relationships. Positional encoding is added to the patch embeddings to represent the spatial location of each patch in the input image.

At the output of the visual transformer, a classification head is attached to predict the class labels or regression values. The architecture of the classification head can vary depending on the specific task, such as using fully connected layers, additional convolutional layers, or a combination of both.

Visual transformers are often pre-trained on large-scale image datasets, such as ImageNet, using self-supervised or supervised learning tasks. Pre-training enables the model to learn meaningful visual representations, which can be transferred to downstream tasks. Fine-tuning is then performed on task-specific labeled data to adapt the model for specific classification, detection, or segmentation tasks.

Moreover, they have achieved remarkable performance across various computer vision benchmarks, demonstrating their effectiveness in tasks such as image classification, object detection, semantic segmentation, and image generation. The Vision Transformer (ViT) architecture proposed by Dosovitskiy et al. [6] is a notable example that has shown strong performance on image classification tasks.

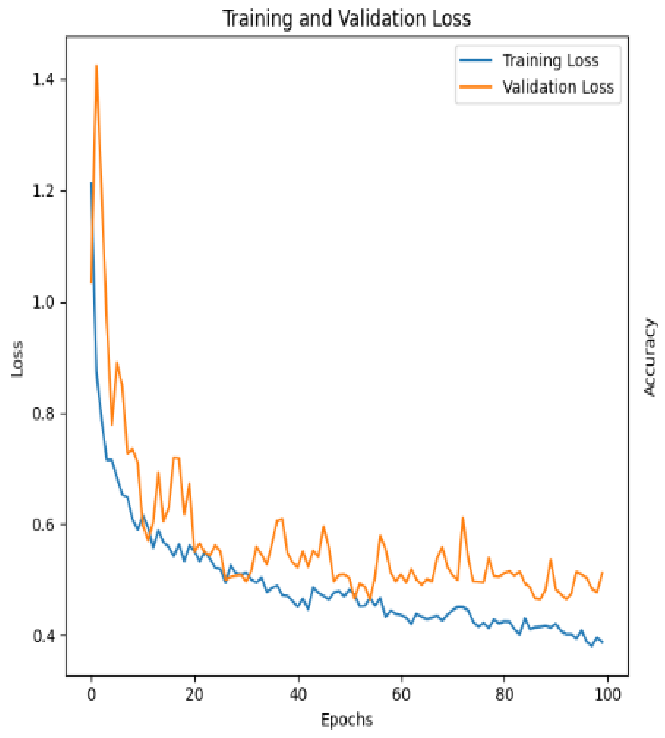
It's important to note that visual transformers are an active area of research, and new variations and advancements continue to emerge.

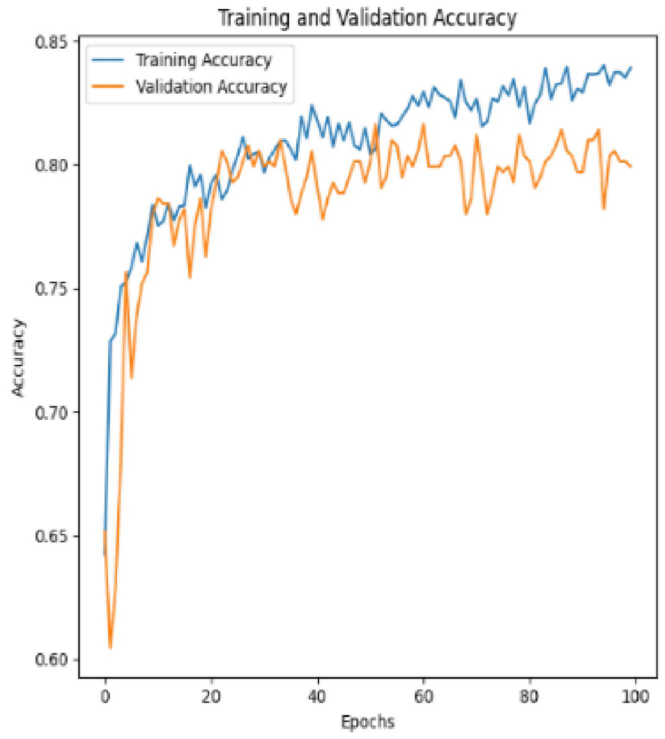
3. EXPERIMENTS & RESULTS

For all three models we conducted trials with many epochs to find the version that performed the best in terms of validation accuracy after running an initial experiment, fine-tuning the models, and the final experiment. By keeping an eye on both the train correctness and validation loss, we took care to avoid overfitting. We used data augmentation techniques and an 80/20 train-validation data split for all three models.

A. DenseNet-121

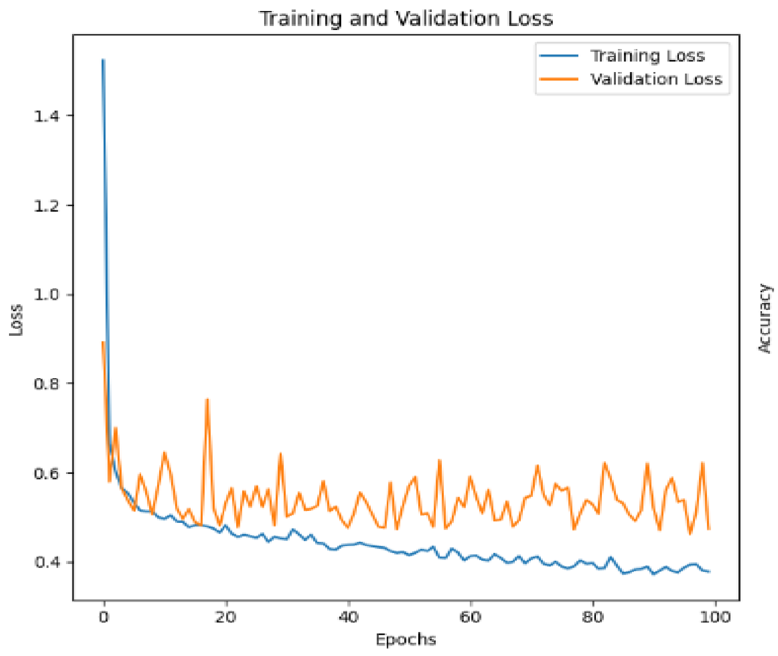
Hyperparameter	Value
random_state	42
stratify	True
rescale	1./255
rotation_range	10
zoom_range	0.2
width_shift_range	0.2
height_shift_range	0.2
horizontal_flip	False
fill_mode	nearest
batch_size	200
target_size	224,224
epochs	100
lr	0.0001
weights	'imagenet'
include_top	False
input_shape	(224, 224, 3)
optimizer	RMSprop
loss	'categorical_crossentropy'
metrics	['accuracy']
checkpoint_filepath	'/content/drive/MyDrive/AM
monitor	'val_accuracy'
save_best_only	True
save_freq	'epoch'
verbose	1

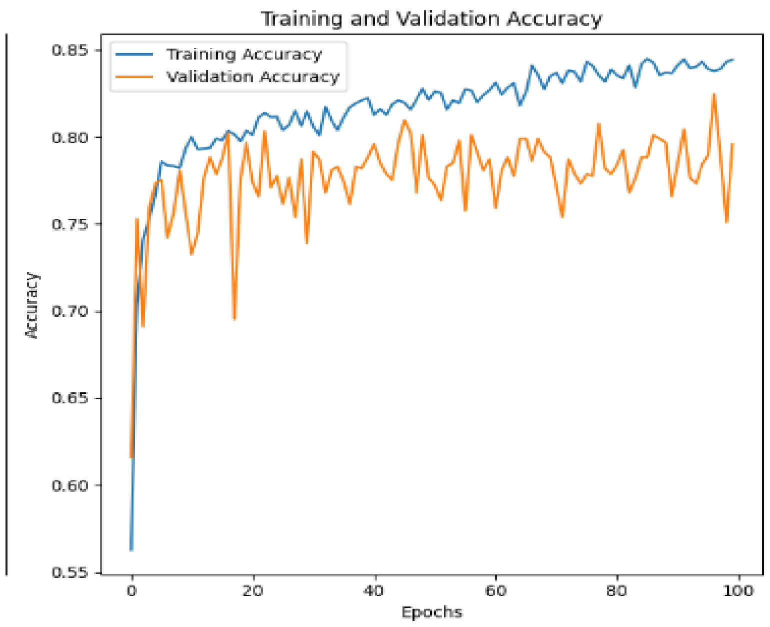




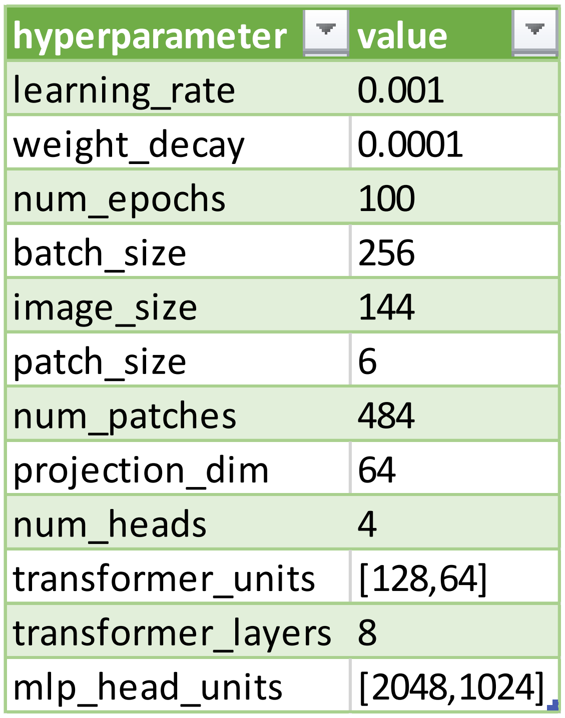
B. Xception

Hyperparameter	Value
batch_size	32
target_size	(224, 224)
epochs	80
lr	0.0001
num_folds	4
checkpoint_filepath	'/content/drive/MyDrive/AMI
monitor	'val_accuracy'
save_best_only	True
save_freq	'epoch'
verbose	1
preprocessing_function	preprocess_input
rotation_range	20
zoom_range	0.2
width_shift_range	0.2
height_shift_range	0.2
horizontal_flip	False
fill_mode	'nearest'
weights	'imagenet'
include_top	False
input_shape	(224, 224, 3)
optimizer	Adam
loss	'categorical_crossentropy'
metrics	['accuracy']
patience	3
factor	0.5
min_lr	0.000005

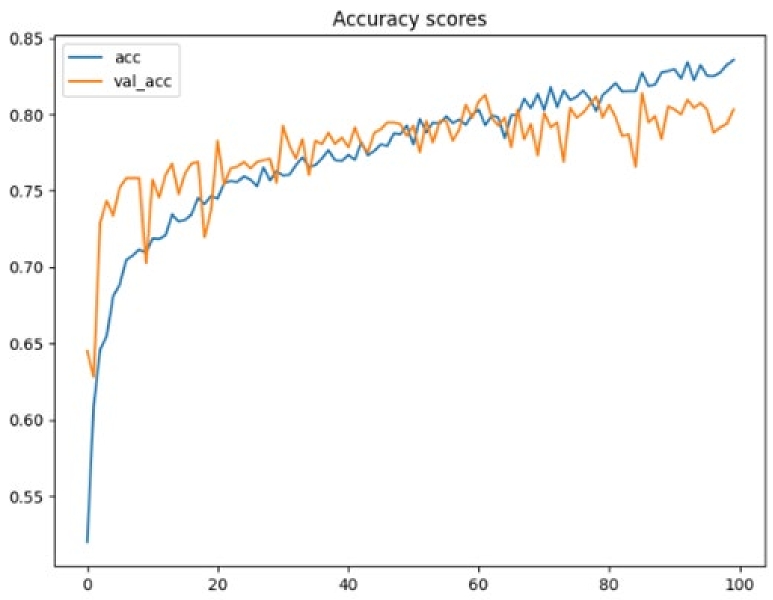




C. VIT



hyperparameter	value
learning_rate	0.001
weight_decay	0.0001
num_epochs	100
batch_size	256
image_size	144
patch_size	6
num_patches	484
projection_dim	64
num_heads	4
transformer_units	[128,64]
transformer_layers	8
mlp_head_units	[2048,1024]



MAJORITY VOTE:

We constructed a simple majority voting system where using the predictions from the three models we calculated the mode (most frequent class value) of the every prediction.

12. REFERENCES

[1] Global Burden of Disease Collaborative Network. Global Burden of Disease Study 2019 (GBD 2019) Results. Seattle, United States: Institute for Health Metrics and Evaluation (IHME), 2020. Available from: http://ghdx.healthdata.org/gbd-results-tool

[2] https://www.kaggle.com/competitions/detect-pneumonia-spring-2023/data

[3] Rajpurkar, P., Irvin, J., Ball, R. L., Zhu, K., Yang, B., Mehta, H., ... & Langlotz, C. P. (2017). Deep learning for chest radiograph diagnosis: A retrospective comparison of the CheXNeXt algorithm to practicing radiologists. PLOS Medicine, 15(11), e1002686.

[4] Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 4700-4708.

[5] Chollet, F. (2017). Xception: Deep Learning with Depthwise Separable Convolutions. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1251-1258.

[6] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale. arXiv preprint arXiv:2010.11929.