Tribhuvan University

Institute of Science and Technology

Central Department of Computer Science and Information Technology

A Dissertation

On

# Comparative Analysis of ResNet50 and Vision Transformer on Paddy Disease Classification

In partial fulfillment of the requirements for the Master's Degree in Computer Science and Information Technology

Submitted to

Central Department of Computer Science and Information Technology

Tribhuvan University

Kirtipur, Nepal

Submitted by

Bhawana Bhattarai

Roll No.: 536/076

T.U. Regd. No.: 5-2-0022-0342-2013

March 2025

Tribhuvan University

Institute of Science and Technology

Central Department of Computer Science and Information Technology

A Dissertation

On

# Comparative Analysis of ResNet50 and Vision Transformer on Paddy Disease Classification

In partial fulfillment of the requirements for the Master's Degree in Computer Science and Information Technology

Submitted to

Central Department of Computer Science and Information Technology

Tribhuvan University

Kirtipur, Nepal

Submitted by

Bhawana Bhattarai

Roll No.: 536/076

T.U. Regd. No.: 5-2-0022-0342-2013

Under Supervision of

Asst. Prof. Bikash Balami

CDCSIT, TU

March 2025

**Tribhuvan University**

**Institute of Science and Technology**

**Central Department of Computer Science and Information Technology**

## STUDENT’S DECLARATION

I hereby declare that I am the sole author of this work and that no sources other than those listed here have been used.

……………………………..

Bhawana Bhattarai

Date: 12th February, 2025

**Tribhuvan University**

**Institute of Science and Technology**

**Central Department of Computer Science and Information Technology**

## SUPERVISOR’S RECOMMENDATION

I hereby recommend that the dissertation prepared under my supervision by **Ms. Bhawana Bhattarai**, entitled **“Comparative Analysis of ResNet50 and Vision Transformer on Paddy Disease Classification,”** in partial fulfillment of the requirements for the degree of M.Sc. in Computer Science and Information Technology, be processed for evaluation.

……………………………..

**Asst. Prof. Bikash Balami**
Central Department of Computer Science and Information Technology
Tribhuvan University, Nepal

Date: 12th February, 2025

**Tribhuvan University**

**Institute of Science and Technology**

**Central Department of Computer Science and Information Technology**

## LETTER OF APPROVAL

This is to certify that we have read this dissertation and, in our opinion, it is satisfactory in scope and quality as a dissertation in partial fulfillment of the requirements for a Master's Degree in Computer Science and Information Technology.

## Evaluation Committee

……………………………..

**Asst. Prof. Sarbin Sayami**

**Head of Department**

Central Department of Computer

Science and Information Technology

Tribhuvan University, Nepal

……………………………..

**Asst. Prof. Bikash Balami**

**Supervisor**

Central Department of Computer

Science and Information Technology

Tribhuvan University, Nepal

……………………………..

**Assoc. Prof. Deo Narayan Yadav**

**External Examiner**

Patan Multiple Campus

Tribhuvan University, Nepal

……………………………..

**Asst. Prof. Bal Krishna Subedi**

**Internal Examiner**

Central Department of Computer

Science and Information Technology

Tribhuvan University, Nepal

**Date: 6th March, 2025**

# ACKNOWLEDGEMENTS

I would like to express my deepest gratitude and sincere thanks to my respected supervisor, Mr. Bikash Balami, Assistant Professor, Central Department of Computer Science and Information Technology, Tribhuvan University, for his invaluable guidance, encouragement, and assistance throughout this research project. His involvement at every stage was instrumental in the completion of this work.

I would also like to extend my heartfelt thanks to the esteemed Head of the Department, Asst. Prof. Sarbin Sayami, Central Department of Computer Science and Information Technology, for his wholehearted support. My sincere thanks go to all the Professors and Lecturers of the Department for their continuous support and guidance.

Finally, I would like to express my profound gratitude to my family, friends, and others who supported me directly or indirectly during the completion of this dissertation.

Bhawana Bhattarai

# ABSTRACT

Plant diseases seriously affect our food supply, thereby affecting farmers, those dependent on farming, and global food security. Early detection of plant disease is critical for effective treatment and minimized yield losses. One of the useful uses of computer vision is the identification of plant diseases by analyzing leaf images. This study compares the ResNet50 model and the ViT model on the paddy disease classification task. Specifically, it trains these models using the Paddy Doctor dataset to evaluate their performance by modifying the learning rate and the number of training epochs.

The Paddy Doctor dataset, which contains 16,225 images categoried into 13 different classes, was used to train and test the models. The ResNet50 model achieved a high training accuracy of 0.98. However, when evaluated on the test dataset, the model's performance decreased, achieving an accuracy of 0.92. On the other hand, the ViT model achieved a remarkably high training accuracy of 0.99. When evaluated on the test dataset, the ViT model maintained strong performance, with an accuracy of 0.93. These results indicate that the ResNet50 model outperforms the ViT model in terms of both training and test accuracy for the paddy disease classification task using the Paddy Doctor dataset.

**Keywords:** *Classification, Deep Learning, CNN, ResNet50, Disease, Paddy, Vision Transformer*

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF ABBREVIATIONS

| | |
|---|---|
| COCO: | Common Objects in Context |
| CNN | Convolutional Neural Network |
| FLoPs: | Floating-point Operations Per Second |
| GDP: | Gross Domestic Product |
| mAP: | Mean Average Precision |
| ResNet | Resduial Network |
| ReLU: | Rectified Linear Unit |
| RCNN: | Region-based Convolutional Neural Networks |
| RoI: | Region of Interest |
| ViT | Vision Transformer |
| VGG: | Visual Geometry Group |
| VGGNet: | Visual Geometry Group Network |

# CHAPTER 1: INTRODUCTION

## 1.1 Background of the Study

Machine learning is the ability of machines to learn on data sets to classify, predict, analyze, and complete a set of tasks on data sets without explicitly programmed to do so. Using machine learning has become a de facto standard in most of the fields, including agriculture. However, in the context of Nepal, this is not the case, especially in the agricultural sectors. Using machine/deep learning in crops to detect diseases, predict yield, and predict optimal harvesting time. The highest market price can have a lot of impact on our country’s GDP so, we still have to depend on foreign trade just for food.

Detecting and classifying plants diseases, especially paddy is a major staple, requires a lot of knowledge which might not be present everywhere. We, as a developing country, do not have a lot of farmers who know what to do when a disease outbreak occurs. This is where machine/deep learning applications can have a considerable impact. In the context of paddy fields, deep learning models can effectively classify various diseases based on visual symptoms observed in plant leaves and other parts.

This study is to analysis paddy disease classification with various models. Different models have been used to analysis paddy diseases classification, but hybrid model ResNet50 and ViT is new way to analysis. This study aims to analysis paddy disease data with hybrid model as defined. This paper is using various classification or other deep learning models to categorize diseases simply with photos of infected parts in the paddy plant. This hybrid framework expects to deliver improve predicted accuracy through leveraging the model's capabilities.

## 1.2 Problem of the Statement

Diseases in crops are a major problem for most farmers, particularly in underdeveloped and developing countries. Early detection, prevention, and treatment of crop diseases can save millions of kilograms of paddy. Today’s, the context of Paddy Disease Classification is existed. Farmers do not have handle tools to identify and classify their paddy leaf disease into due classes of paddy leaf disease. All depend on intuition and experiences. There is not scientific and technological bases to depend on. Implementing this proposed model to trace the paddy leaf disease classification so that it could identify the particular classes of disease. It will assist farmers to be sure about the paddy leaf disease so that they could follow the right treatment.

## 1.3 Objectives of the Study

The main objectives of this dissertation are:

1. To implement ResNet50 and ViT that identify paddy leaf disease classes.
2. To evaluate and compare the performance of ResNet50 and ViT architecture for paddy leaf diseases classification.

## 1.4 Dissertation Organization

This dissertation is organized into six chapters as follows: Chapter 1 consists of the introduction, background, problem statement and objectives of this dissertation. Chapter 2 includes the literature review of the existing works related to the ResNet50, ViT on Paddy Disease Classifications. Chapter 3 describes the methodology used to implement ResNet50 and ViT on paddy disease classification. Chapter 4 explains the implementation tools, test environment, and hyperparameters used for the research work. Chapter 5 presents the results and analysis of the outputs. Chapter 6 concludes the research work and discusses the possible future works in the study.

# CHAPTER 2: LITERATURE REVIEW

## 2.1 Literature Review

Before commencing this article, I reviewed several other papers to gain a better insight. This section summarizes noteworthy outcomes from relevant literature.

A pure transformer applied directly to picture patch sequences can achieve excellent results on image classification tasks, negating the need for CNNs. In comparison to state-of-the-art convolutional networks, Vision Transformer (ViT) achieves superior results when pre-trained on large amounts of data and then transferred to multiple mid-sized or small image recognition benchmarks (ImageNet, CIFAR-100, VTAB, etc.) while requiring significantly less computing power to train..

Bangladesh-based research works on a disease prediction system in tomato plants with a very light deep learning model using a region-based modified CNN. The architecture has three building blocks: Feature Map Generator Network, Region of Interest (RoI) Generator Network, and Detector Network. The dataset was an openly accessible repository containing 54,303 plant leaf images categorized into thirty classes of which only tomato leaf images were used from this dataset. The model was lightweight because it employed fewer trainable parameters than their faster RCNN baseline model. On a benchmark dataset, the model's mAP achieved 96.31%, while its recall and f2 scores achieved 97.71% and 98.07%, respectively, indicating satisfactory empirical performance.

Author Alex Wenda propose a method to identify the three major paddy plant diseases that share similar symptoms: blast, brown spot, and narrow brown spot. They achieve this by analyzing the texture of color segments within diseased leaf regions. Blobs are used to quantify objects, area, and perimeter whereas color segmentation is used for finding paddy leaf color parameters as of lesion boundary. To separate Region of Interest (RoI) it uses four methods for threshold value; Global, Local, Variable, and Otsu threshold. Using all these, its best accuracy from four of its methods was 90.7%. The main characteristics used for classification were lesion percentage, lesion type, spot lesion color, boundary lesion color, and lesion paddy leaf color. Using flash cameras and appropriately distancing objects, they manually created their datasets by capturing images of diseased paddy leaves.

Author Yingzi Huo provides an overview of the use of ViT in image classification tasks. It begins by introducing the process of implementing image classification and the fundamental architecture of the ViT model. Next, it examines and summarizes various image classification techniques, such as CNN-based, ViT-based, and traditional techniques, and compares CNN and ViT.

Before going over contemporary ViT techniques and talking about how ViT is applied in every area of image processing, Author [5] presented the fundamental concepts of vision transformers. Second, highlight significant research and compile a timeline of ViTs across a variety of domains (e.g., object identification, segmentation, classification, and point cloud). Lastly, use an ImageNet dataset to compare the accuracy of various ViTs and CNN techniques. ViTs can thereby resolve a variety of potential convolutional neural network (CNN) problems. All things considered, ViT performs admirably, and remarkable outcomes are achieved when CNN and the self-attention mechanism are coupled.

The mathematical techniques that make Vision Transformer effective are defined by Author [6], who also explores and explains cutting-edge approaches and evaluates how well they work in various application scenarios. The most effective ways to guarantee suboptimal estimating performances are examined in this work, and the Efficient Error Rate is introduced to standardize and compare model properties that impact hardware devices throughout inference time, including the number of parameters, bits, FLOPs, and model size.

In-depth empirical evidence regarding residual networks' ease of optimization and ability to improve accuracy with significantly greater depth is provided by Author [7]. ResNet evaluates on the ImageNet dataset with a depth of up to 152 layers, which is 8× deeper than VGG nets but still has lower complexity; on the COCO object detection dataset, it achieves a 28% relative improvement.

A novel model called EfficientRMT-Net, which combines the Vision Transformer (ViT) and ResNet-50 architectures, is capable of precisely and successfully identifying a variety of potato leaf diseases. The suggested model seeks to address the drawbacks of conventional approaches, which are frequently time-consuming, labor-intensive, and prone to errors because disease presentation is unpredictable. On a broad picture dataset, EfficientRMT-Net achieves 97.65% accuracy, whereas on a specialized dataset of potato leaf images, it achieves 99.12% accuracy [8].

# CHAPTER 3: METHODOLOGY

Without making any changes to the architecture, this study first evaluated the performance of the ResNet50 and ViT architectures on the Paddy Doctor dataset. Following that, hyperparameters that were influenced by the literature were implemented. The model's performance in classifying paddy disease on leaves was then evaluated.

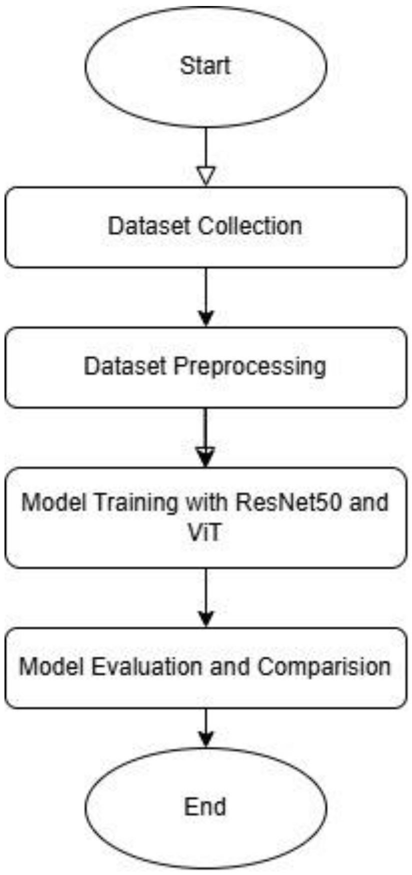

Start
Dataset Collection
Dataset Preprocessing
Model Training with ResNet50 and ViT
Model Evaluation and Comparision
End

Figure 3. 1: Overview of Methodological Steps

## 3.1 Data Collection

The dataset from Paddy Doctor Dataset [1] consists of total 16,225 images categorized into 13 classes which include a normal paddy plants class and 12 disease classes. The dataset covers a diverse range of paddy diseases which are Fungal (Brown Spot, Blast, and Downy mildew), Bacterial (Bacterial leaf streak, Bacterial leaf blight, and Bacterial panicle blight), Viral(Tungro), and Pests(Yellow stem borer, Black stem borer, Leaf roller, White stem borer, and Hispa). The images are of 1,080 x 1,440 pixels collected using a smartphone camera.

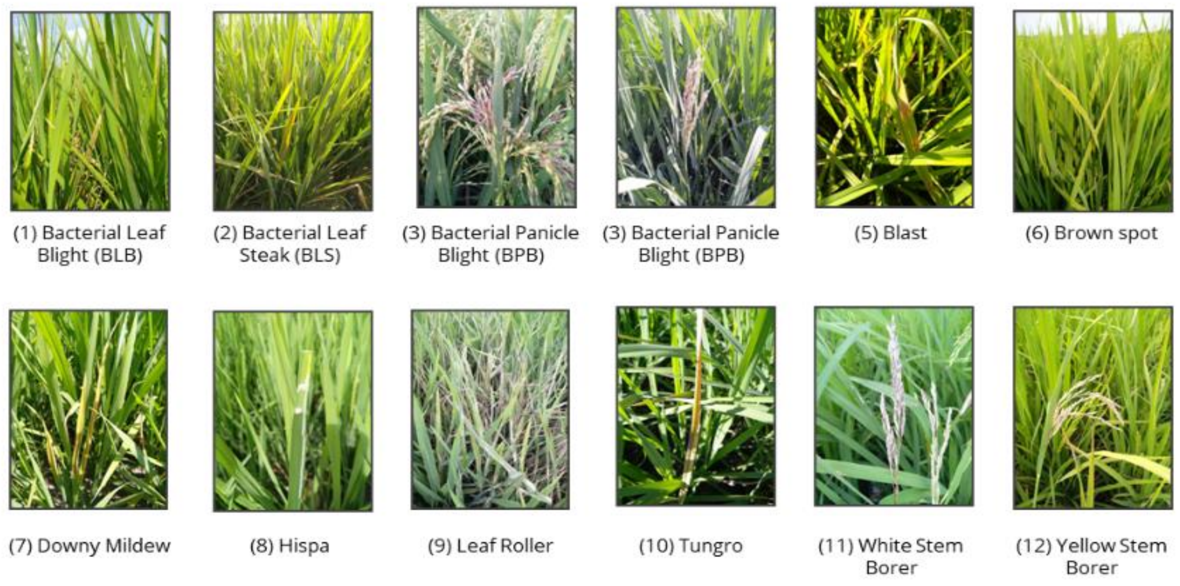

(1) Bacterial Leaf Blight (BLB)
(2) Bacterial Leaf Steak (BLS)
(3) Bacterial Panicle Blight (BPB)
(3) Bacterial Panicle Blight (BPB)
(5) Blast
(6) Brown spot
(7) Downy Mildew
(8) Hispa
(9) Leaf Roller
(10) Tungro
(11) White Stem Borer
(12) Yellow Stem Borer

Figure 3. 2: Sample Images from Paddy Doctor Dataset [1]

## 3.2 Data Preprocessing

The entirety of the data preprocessing constitutes the following steps:

i. Image Resizing

Resizing images to 256x256 or 380x380 dimensions depending upon the experiment performed.

ii. Image Normalization

The 3-channel RGB images with pixel values between 0 and 256 are normalized to the range of 0 and 1 to ease computation by convolution networks and vision transformers.

iii. Image Filtering

Few images in the dataset that were wrongly labeled, blurry, and had unusual aspect ratios were filtered out, especially the images collected locally.

## 3.3 Paddy Disease Classification

For classification task, various state-of-the-art classification models are experimented with ResNet50 and the Vision Transformer. These models are fine-tuned on the Paddy Doctor dataset starting with pre-trained Imagenet models. By exploring a diverse range of classification methodologies, the study aims to evaluate the performance and effectiveness of different model architectures in accurately categorizing single paddy diseases at a time.

### 3.3.1 CNN Architectures

CNN architectures include of convolutional layers for feature extraction, pooling layers for data down sampling, activation functions for non-linearity, and fully connected layers for feature combining. Its ability to recognize patterns and extract features from grid-like data, such as images, makes it a valuable tool for computer vision tasks including image classification, object recognition, and segmentation. The prominent CNN architectures are ResNet, VGGNet, MobileNet, AlexNet, and EfficientNet. ResNet is one of these architecture that has been applied to our dataset.

#### 3.3.1.1 ResNet

The general concept "the deeper the better" applies to CNN architecture since there is a larger parameter space for the models to explore, increasing their adaptability to any space. However, it is noted that the vanishing gradient problem, in which data crucial for training previous layers is lost during backpropagation, causes performance to deteriorate at a certain depth.

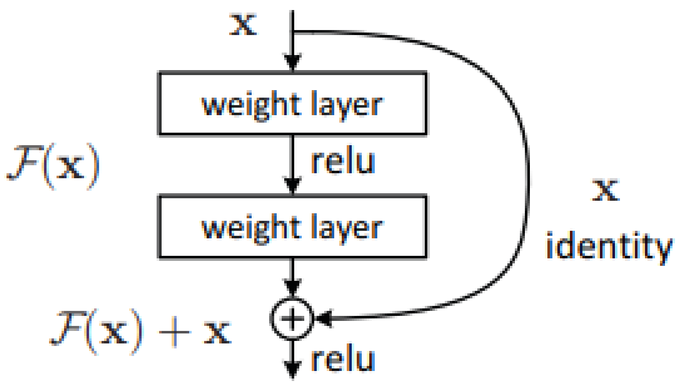

x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 3. 3: Building Block of Residual Network [7]

A residual block learns a residual function—that is, the difference between the input and the desired output—instead of a direct mapping from input to output. This is accomplished by adding the input to the output of the skipped levels using a "shortcut connection" that skips one or more layers within the block. Consequently, the skip function serves as an identity function. The weight layer and skip connection outputs are combined. If the dimensions of the output from the weight layer and the skip connection do not match, the skip connections pass through a CNN layer.

**<u>Components of ResNet</u>**

ResNet's main components are:

i. **Convolutional Layers**

   These layers generate activation maps by using kernels that move over the input image to extract features. Depending on its variety, ResNet frequently combines 3x3 and 1x1 convolutional layers for effective feature extraction.

ii. **Batch Normalizations**

   This approach normalizes activations within each batch, minimizing the internal covariate shift and stabilizing the training process. It accelerates training and enhances model performance.

iii. **ReLU Activation and Softmax Layer**

   Rectified Linear Unit (ReLU) is a non-linear activation function that aids the network in discovering complex correlations between features. It reduces all negative values to zero while preserving positive values. The softmax layer computes the probability distribution of the input image across distinct classes. It uses the feature vector created by the previous layer in order to predict the class with the greatest probability.

iv. **Bottleneck Design**

   By lowering the network's parameter count, this design significantly improves efficiency. In order to reduce the number of channels processed by the computationally costly 3x3 layer, it employs a 1x1 convolutional layer both before and after the 3x3 convolutional layer in the residual block.

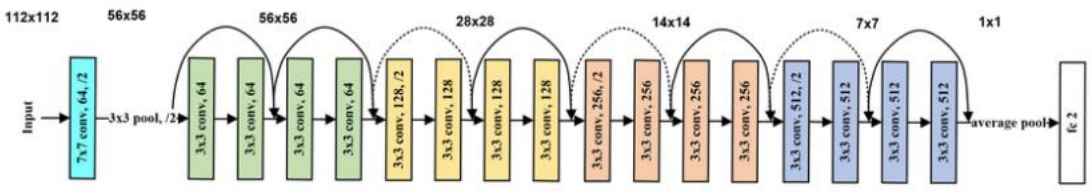


Figure 3. 4: ResNet Architecture [10]

## 3.3.2 Vision Transformer

For computer vision tasks, a new architecture called the Vision Transformer (ViT) is outperforming CNN systems. It makes use of the strength of transformer-inspired self-attention mechanisms that were first created for applications involving natural language processing. Self-attention layers take the place of convolutional layers in conventional CNN, allowing for direct communication between all input patches. For tasks like image classification, image segmentation and object detection, this architecture is quite effective since it is excellent at capturing global dependencies and contextual relationships inside images.

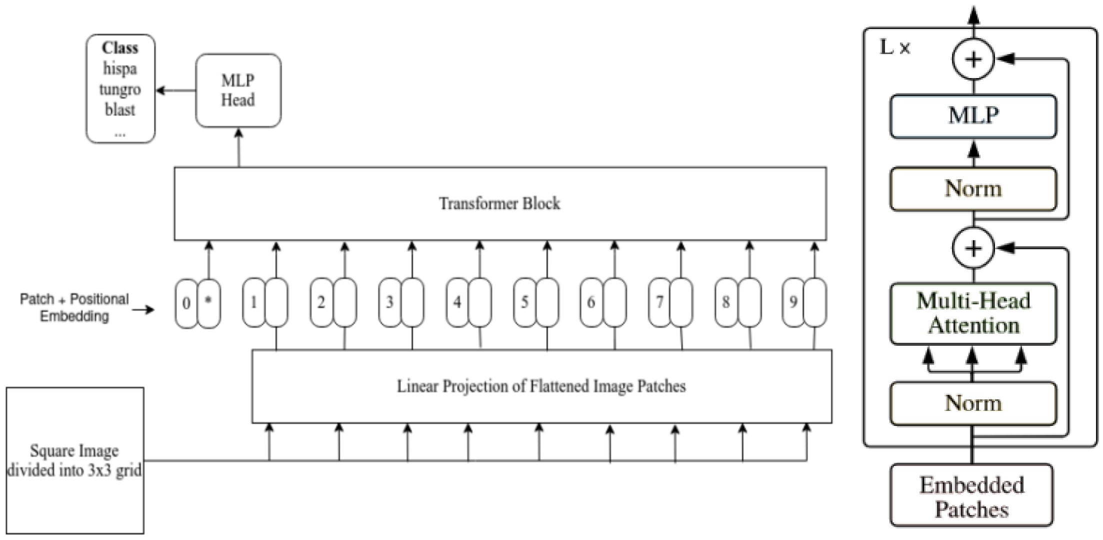


Figure 3. 5: Vision Transformer Architecture and Transformer Encoder [2]

The patch embedding layer, the initial layer of Vision Transformer, splits input images into fixed-size patches and treats each one as a token, similar to how language processing tasks work. Every patch is linearly incorporated into a representation with less dimensions. In order to prepare picture patches for processing by the transformer layers, this embedding phase transforms them into vector sequences.

The remainder of the design is similar to the transformer architecture that is employed for jobs involving natural language processing. Vision Transformers, as opposed to CNNs, process images as a sequence of patches, which enables them to focus on several areas of the image at once and recognize complex patterns throughout the image. Although it requires more training and inference time, this global self-attention mechanism is very useful for detecting diffuse and subtle disease symptoms in paddy leaves.

## 3.4 Evaluation Metrics

Accuracy, precision, recall, F1-score, and confusion matrix are used in evaluation metrics for the classification.

**Accuracy**

Accuracy measures the proportion of correct predictions out of the total predictions, providing an overall performance indicator of the model.

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

Where,

TP (True Positives): Correctly predicted positive cases.

TN (True Negatives): Correctly predicted negative cases.

FP (False Positives): Incorrectly predicted positive cases.

FN (False Negatives): Incorrectly predicted negative cases.

**Precision**

Precision indicates the proportion of true positive predictions out of the total predicted positives, reflecting the model's ability to avoid false positives.

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

**Recall**

Recall, also known as sensitivity, is the proportion of true positive predictions among actual positives, demonstrating the model's ability to capture relevant instances.

$$Recall = \frac{TP}{TP + FN}$$

**F1-Score**

A single metrics that balances the trade-off between precision and recall is the F1-score, which is the harmonic mean of precision and recall.

$$F1 - Score = 2.\frac{Precision.Recall}{Precision + Recall}$$

**Confusion Matrix**

The confusion matrix is a table that displays the counts of false positives (FP), false negatives (FN), true positives (TP), and true negatives (TN) to provide an overview of a classification model's performance.

# CHAPTER 4: IMPLEMENTATION

## 4.1 Environment and Hardware

This study was conducted using online resources, specifically Google Colab. The GPU and TPU are both offered by Google Colab. These two hardware resources were both utilized. The resources listed below were used.

i. GPU: T4GPU, L4GPU

ii. TPU: TPUv2.

## 4.2 Programming Language and Libraries

### 4.2.1 Programming Language

In this research, Python was used as the programming language.

### 4.2.2 Libraries and packages

In this study, TensorFlow, Keras, Numpy, and Matplotlib were used.

## 4.3 Hyperparameters

The following hyperparameters were used in this research work:

**Training Hyperparameters**

- **Epochs**

  The number of epochs determines how often the model iterates over the entire training dataset.
- **Batch Size**

  The number of samples that can propagate over the network simultaneously is determined by the batch size. It has an affect on training speed and memory usage.
- **Learning Rate**

  The size that a model modifies its parameters during training is determined by its learning rate.

**Additional Hyperparameters**

- **Adam Optimizer**

  To reduce the error or loss function during training, an optimizer modifies weights and learning rates. It determines the model's ideal collection of parameters, or weights. The Adam optimizer is employed. Adam's adjustable learning rate characteristics frequently lead to speedier convergence. Adam Optimizer was used to train each model in this study.

- **Loss Functions**

  A loss function evaluates a model's performance. It calculates the difference between the actual labels, or the true distribution, and the model-predicted probability distribution. The cross-entropy loss function is categorical.

## 4.4 Steps of Implementation

The research was formulated in the following manner. The first step was to prepare the dataset and preprocess the images. After these processes were completed, the processed photos were utilized to train the model and evaluate its performance.

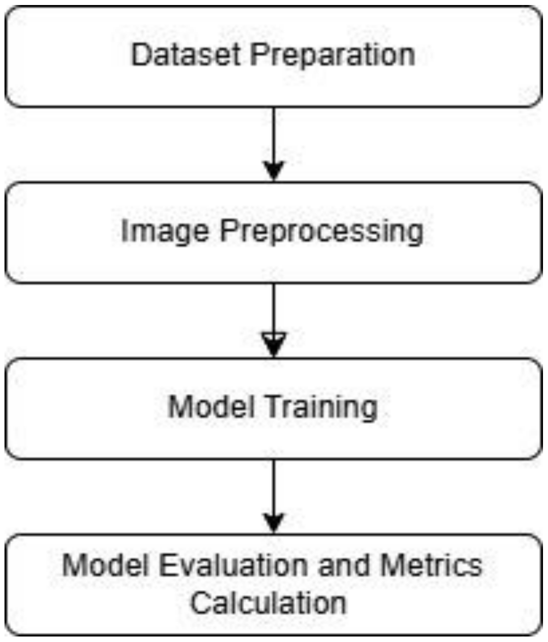


Figure 4. 1: Inclusive Framework of Technical Section

### 4.4.1 Data Preparation

Paddy Doctor dataset contains 16,225 RGB images. The following is how the images were split into Training and Testing sets.

a. Training Set (80%)- 12,980 Images.
b. Testing Set (20%)- 3,245 Images

The data was split into training (80%) and testing (20%) sets before augmentation was applied to ensure sufficient data for model training and evaluation.

### 4.4.2 Image Preprocessing

All images were resized to either 256x256 or 384x384 pixels depending on the experiment, with pixel values normalized using the mean and standard deviation of the ImageNet dataset for each of the three channels.

### 4.4.3 Model Training

The models were trained with a batch size of 16 and iterates for a maximum of 15 epochs. The learning rate started higher at 0.01 in the initial experiments and gradually decreased to as low as 1e-3 depending on the specific experiment. All the models were trained with Adam Optimizer. Each model, pre-trained on ImageNet, was fine-tuned on our dataset to leverage its unique strengths, such as hierarchical feature extraction in ResNet50 and global self-attention mechanism in ViT.

### 4.4.4 Model Evaluation

After model training, the model's performance on the test set is evaluated to establish accuracy and overall effectiveness. The evaluation provides metrics to assist evaluate how effectively the model generalizes to previously unseen data.

# CHAPTER 5: RESULTS AND ANALYSIS

## 5.1 Results

ResNet50 and ViT architectures were experimented with best-performing models in the initial experiments were manually hype tuned and trained further. Experimental results of some of the best-performing models are summarized in Table 1.

Table 5. 1: Experiment Results obtained on different architecture

| Experiment Name | Learning Rate | Batch Size | Image Size | Epochs | Train Accuracy | | Test Accuracy | |
|---|---|---|---|---|---|---|---|---|
| | | | | | Acc. | F1-Score | Acc. | F1-Score |
| ResNet50 | 1e-2 | 16 | 256 | 8 | 73 | 73 | 69 | 69 |
| ResNet50 | 1e-3 | 16 | 256 | 8 | 97 | 97 | 91 | 91 |
| ResNet50 | 1e-3 | 16 | 256 | 10 | 98 | 99 | 92 | 92 |
| ViT-16x16 | 1e-3 | 32 | 384 | 8 | 94 | 93 | 85 | 85 |
| ViT-16x16 | 1e-3 | 32 | 384 | 10 | 99 | 99 | 93 | 92 |

## 5.2 Analysis

To determine the best model for paddy disease classification, the study evaluates the accuracy metrics of the ResNet50 and ViT models. The experiment determines whether the ResNet50 model has equivalent accuracy to the ViT model for paddy disease classification on the Paddy Doctor dataset.

The ResNet50 model achieved a training accuracy of 0.98 and a validation accuracy of 0.92 over 10 epochs. These results suggest that while the ResNet50 model performs well on the training data, there is a slightly drop in performance when applied to the test data, indicating potential overfitting.

The ViT model achieved a training accuracy of 0.99 and a validation accuracy of 0.93 over 10 epochs. This training accuracy indicating an excellent fit to the training data. High training accuracy in both models suggests that they were able to effectively capture the features of the paddy disease image present in the Paddy Doctor dataset. This consistency in higher accuracy points towards the ViT model's robustness and generalization capabilities.

In this study, the ViT model with a patch size of 16x16 pixels performed best than the ResNet50 models on the Paddy Doctor dataset in terms of accuracy metrics. The ViT model consistently achieved higher accuracy, both on the training and test datasets.

Followings are the confusion matrix for the best-performing model, Vision Transformer with a patch size of 16x16 pixels pre-trained on ImageNet data.

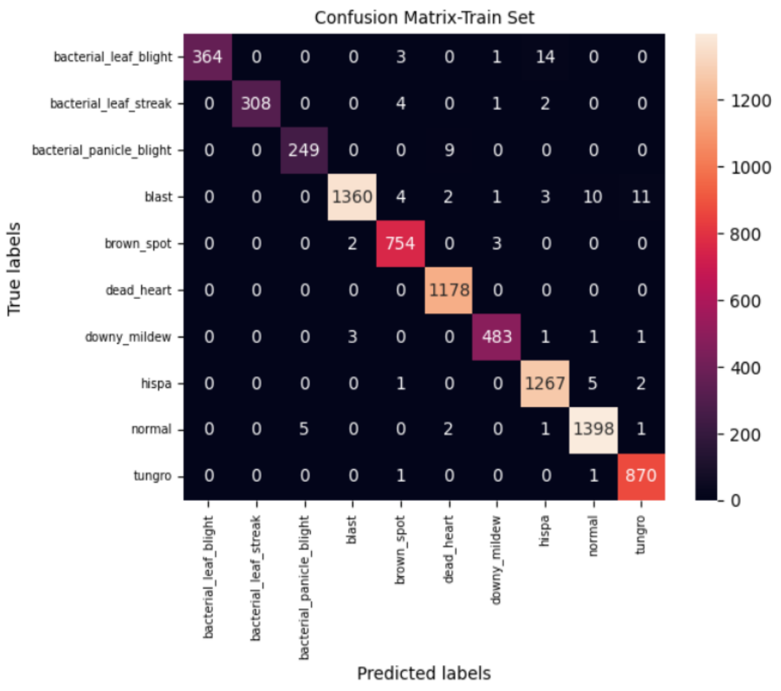
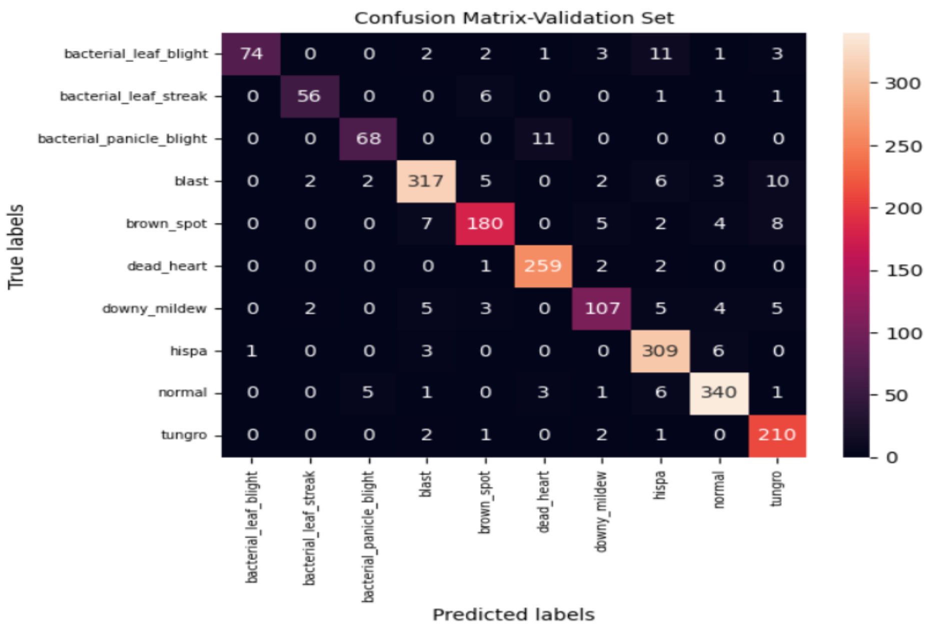


Figure 5. 1: Confusion Matrix for Train Set



| | bacterial_leaf_blight | bacterial_leaf_streak | bacterial_panicle_blight | blast | brown_spot | dead_heart | downy_mildew | hispa | normal | tungro |
|---|---|---|---|---|---|---|---|---|---|---|
| bacterial_leaf_blight | 74 | 0 | 0 | 2 | 2 | 1 | 3 | 11 | 1 | 3 |
| bacterial_leaf_streak | 0 | 56 | 0 | 0 | 6 | 0 | 0 | 1 | 1 | 1 |
| bacterial_panicle_blight | 0 | 0 | 68 | 0 | 0 | 11 | 0 | 0 | 0 | 0 |
| blast | 0 | 2 | 2 | 317 | 5 | 0 | 2 | 6 | 3 | 10 |
| brown_spot | 0 | 0 | 0 | 7 | 180 | 0 | 5 | 2 | 4 | 8 |
| dead_heart | 0 | 0 | 0 | 0 | 1 | 259 | 2 | 2 | 0 | 0 |
| downy_mildew | 0 | 2 | 0 | 5 | 3 | 0 | 107 | 5 | 4 | 5 |
| hispa | 1 | 0 | 0 | 3 | 0 | 0 | 0 | 309 | 6 | 0 |
| normal | 0 | 0 | 5 | 1 | 0 | 3 | 1 | 6 | 340 | 1 |
| tungro | 0 | 0 | 0 | 2 | 1 | 0 | 2 | 1 | 0 | 210 |

Figure 5. 2: Confusion Matrix for Validation Set

# CHAPTER 6: CONCULSION & FUTURE RECOMMENDATION

## 6.1 Conclusion

This study aimed to compare the performance of ResNet50 and ViT models for paddy disease classification tasks using the Paddy Doctor dataset. The Paddy Doctor dataset, comprising 16,225 images categorized into 13 different classes. 80% (12,980) of the data was used for training, while 20% (3,245) was utilized for testing both the ResNet50 and ViT models. The study followed a systematic approach encompassing data preprocessing, model training, inference, and evaluation. The study evaluated key metrics including accuracy to determine the effectiveness of each model.

The ResNet50 model achieved a training accuracy of 0.98, indicating strong initial performance on the training data. However, when evaluated on the test dataset, the model's performance decreased, with an accuracy of 0.92.

In contrast, the ViT model achieved a high training accuracy of 0.99, suggesting an excellent fit to the training data. When evaluated on the test dataset, the ViT model maintained strong performance, with an accuracy of 0.93.

Overall, the ViT model consistently achieved higher training accuracy compared to the ResNet50 model, indicating better generalization and robustness in paddy disease classification tasks. Therefore, the VIT model is recommended for real-time paddy disease classification applications as it demonstrated superior performance over the ViT model in classifying paddy disease from the Paddy Doctor dataset, where accuracy and generalization are critical.

## 6.2 Future Recommendation

Future research and development should focus on enhancing these models utilizing a more varied set of training images covering more diseases and expanding their applicability in order to guarantee broader impact and adoption among farmers. Including support for additional crops beyond paddy, such as wheat, maize, potato, or soybean, to create a more complete agricultural disease management platform. Implementing a feedback mechanism within the application to gather user input and continuously improve the accuracy and performance of the disease identification and treatment suggestions. Enhancing mobile application with additional features like a community forum, field notes and progress tracking, feedback and support, and multilingual support.

# REFERENCES

[1] P. A. M. Petchiammal A, "Paddy Doctor: A Visual Image Dataset for Automated Paddy Disease Classification and Benchmarking," 18 November 2023. [Online]. Available: 10.21227/hz4v-af08.

[2] D. Alexey and B. Lucas , "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," *ICLR,* 2021.

[3] H. Kaiming and Z. Xiangyu , "Deep Residual Learning for Image Recognition," *arXiv:1512.03385v1,* 10 Dec 2015.

[4] S. Kashif and Q. Imran , "EfficientRMT-Net–An Efficient ResNet-50 and Vision Transformers Approach for Classifying Potato Plant Leaf Diseases," *Sensors,* 2023.

[5] P. Lorenzo and R. Paolo , "A survey on efficient vision transformers: algorithms, techniques, and performance benchmarking," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 2024.

[6] M. M. Hamed and L. Yuchuan , "Detecting Flashover in a Room Fire based on the Sequence of Thermal Infrared Images using Convolutional Neural Networks," 2022.

[7] M. S. Mia and A. Nai, "ViTs are Everywhere: A Comprehensive Study Showcasing Vision Transformers in Different Domain," *IEEE,* 2023.

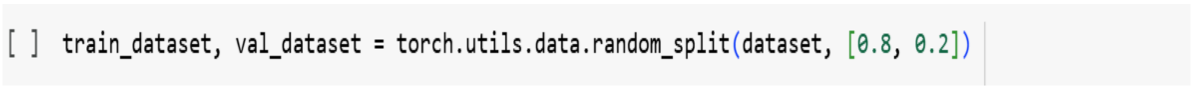
[8] H. Yingzi , J. Kai and C. Jiahong , "Vision Transformer (ViT)-based Applications in Image Classification," *IEEE,* May 2023.

[9] W. Alex and P. Inggih , "Identification of paddy leaf diseases based on texture analysis of Blobs and color segmentation," *TELKOMNIKA Telecommunication, Computing, Electronics and Control,* vol. 18, no. 4, August 2020.

[10] R. Hasin and M. H. Ali, "Plant Disease Detection using Region-Based Convolutional Neural Network," *arXiv:2303.09063v2,* p. 19, 2023.

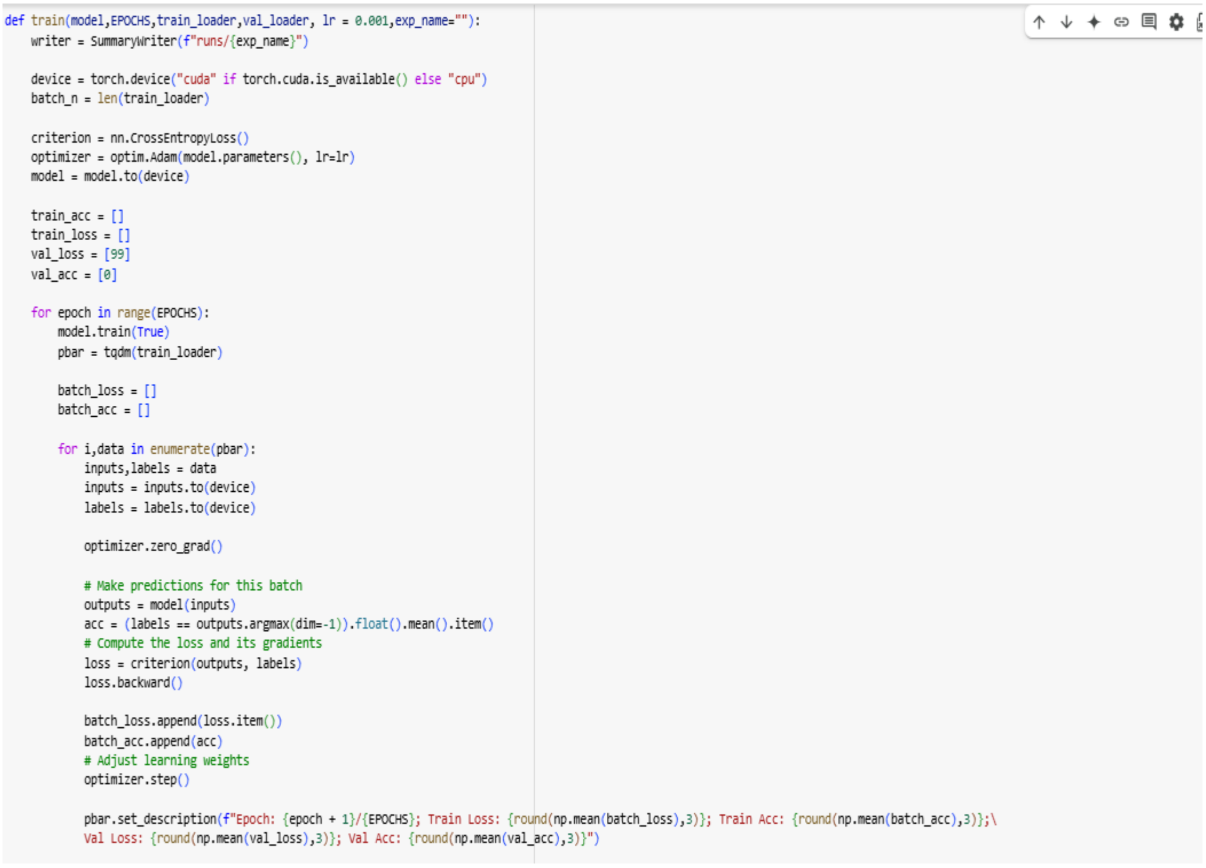
# APPENDIX

A example code for splitting using Python Random Library.

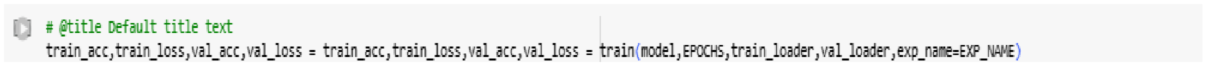


```
[ ]  train_dataset, val_dataset = torch.utils.data.random_split(dataset, [0.8, 0.2])
```

Example code for training model

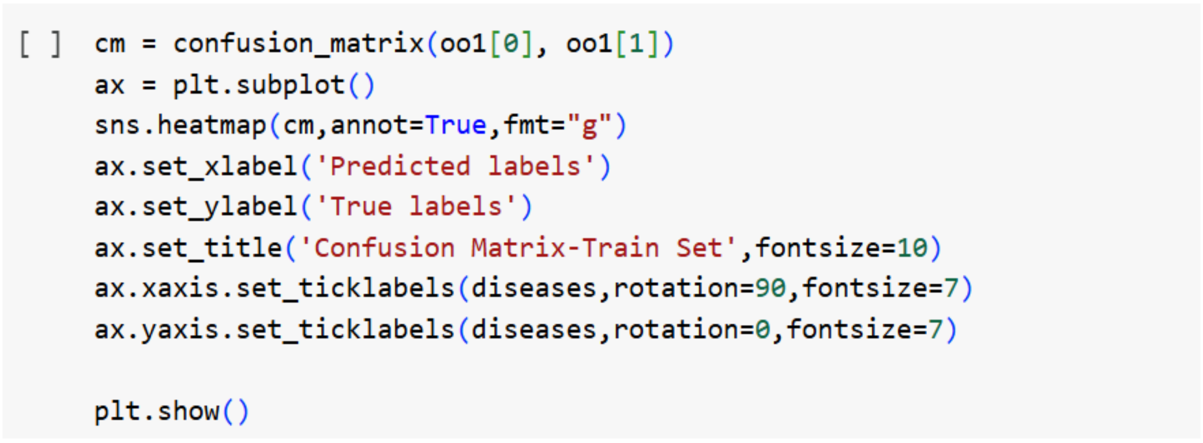


```python
def train(model,EPOCHS,train_loader,val_loader, lr = 0.001,exp_name=""):
    writer = SummaryWriter(f"runs/{exp_name}")

    device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
    batch_n = len(train_loader)

    criterion = nn.CrossEntropyLoss()
    optimizer = optim.Adam(model.parameters(), lr=lr)
    model = model.to(device)

    train_acc = []
    train_loss = []
    val_loss = [99]
    val_acc = [0]

    for epoch in range(EPOCHS):
        model.train(True)
        pbar = tqdm(train_loader)

        batch_loss = []
        batch_acc = []

        for i,data in enumerate(pbar):
            inputs,labels = data
            inputs = inputs.to(device)
            labels = labels.to(device)

            optimizer.zero_grad()

            # Make predictions for this batch
            outputs = model(inputs)
            acc = (labels == outputs.argmax(dim=-1)).float().mean().item()
            # Compute the loss and its gradients
            loss = criterion(outputs, labels)
            loss.backward()

            batch_loss.append(loss.item())
            batch_acc.append(acc)
            # Adjust learning weights
            optimizer.step()

            pbar.set_description(f"Epoch: {epoch + 1}/{EPOCHS}; Train Loss: {round(np.mean(batch_loss),3)}; Train Acc: {round(np.mean(batch_acc),3)};\
            Val Loss: {round(np.mean(val_loss),3)}; Val Acc: {round(np.mean(val_acc),3)}")
```



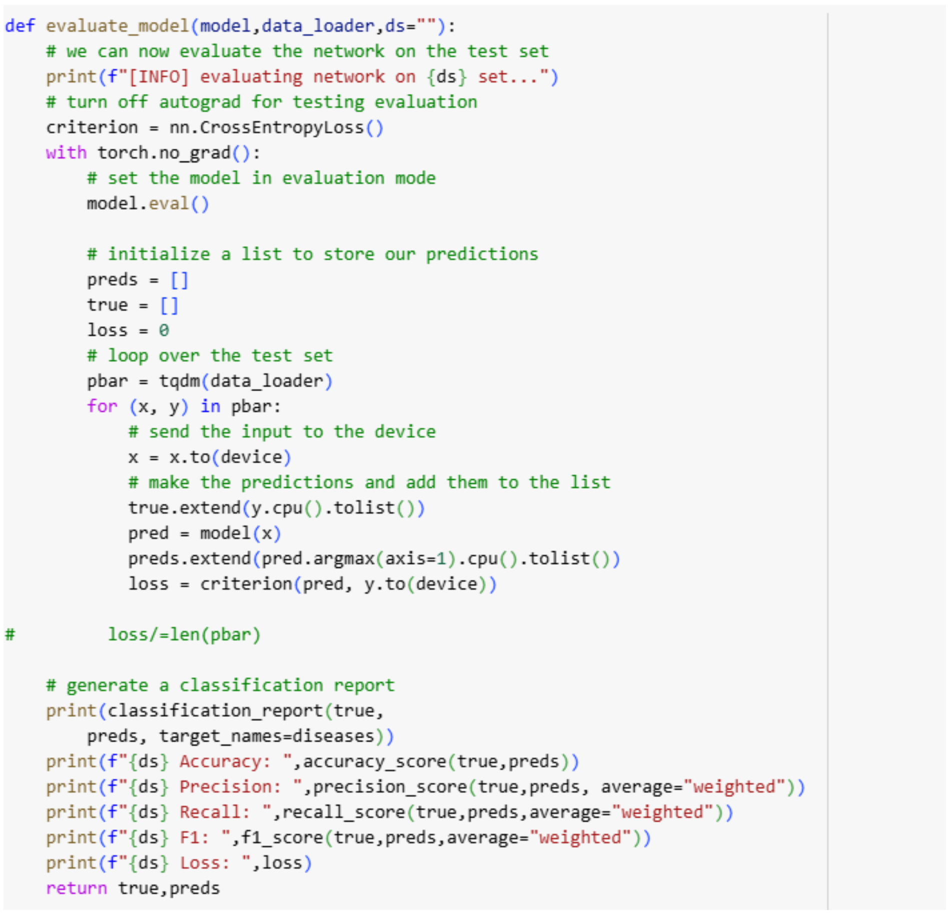
```python
# @title Default title text
train_acc,train_loss,val_acc,val_loss = train_acc,train_loss,val_acc,val_loss = train(model,EPOCHS,train_loader,val_loader,exp_name=EXP_NAME)
```

Example Code for creating Confusion Matrix using Sklearn Library



```python
[ ]  cm = confusion_matrix(oo1[0], oo1[1])
     ax = plt.subplot()
     sns.heatmap(cm,annot=True,fmt="g")
     ax.set_xlabel('Predicted labels')
     ax.set_ylabel('True labels')
     ax.set_title('Confusion Matrix-Train Set',fontsize=10)
     ax.xaxis.set_ticklabels(diseases,rotation=90,fontsize=7)
     ax.yaxis.set_ticklabels(diseases,rotation=0,fontsize=7)

     plt.show()
```

Example code for evaluating model



```python
def evaluate_model(model,data_loader,ds=""):
    # we can now evaluate the network on the test set
    print(f"[INFO] evaluating network on {ds} set...")
    # turn off autograd for testing evaluation
    criterion = nn.CrossEntropyLoss()
    with torch.no_grad():
        # set the model in evaluation mode
        model.eval()

        # initialize a list to store our predictions
        preds = []
        true = []
        loss = 0
        # loop over the test set
        pbar = tqdm(data_loader)
        for (x, y) in pbar:
            # send the input to the device
            x = x.to(device)
            # make the predictions and add them to the list
            true.extend(y.cpu().tolist())
            pred = model(x)
            preds.extend(pred.argmax(axis=1).cpu().tolist())
            loss = criterion(pred, y.to(device))

#         loss/=len(pbar)

    # generate a classification report
    print(classification_report(true,
        preds, target_names=diseases))
    print(f"{ds} Accuracy: ",accuracy_score(true,preds))
    print(f"{ds} Precision: ",precision_score(true,preds, average="weighted"))
    print(f"{ds} Recall: ",recall_score(true,preds,average="weighted"))
    print(f"{ds} F1: ",f1_score(true,preds,average="weighted"))
    print(f"{ds} Loss: ",loss)
    return true,preds
```