Comprehensive Report

Project Documentation for COVID-19 Radiography Classification(X-RAY)

1. Introduction:

This project aims to classify chest X-ray images into categories based on the presence of COVID-19, pneumonia, and normal lung conditions using a **Convolutional Neural Network (CNN)**. The dataset used for this task is the **COVID-19 Radiography Dataset** sourced from Kaggle, which includes labeled images of COVID-19, pneumonia, and normal (healthy) lungs.

The four categories of images are:

- **COVID**: X-ray images of patients with COVID-19.
- **Lung Opacity**: X-ray images showing lung opacity indicative of pneumonia.
- **Normal**: Healthy lung X-rays.
- **Viral Pneumonia**: X-ray images of viral pneumonia patients.

2. Dataset:

The dataset consists of images classified into four categories. Each image undergoes preprocessing before being fed into the model:

- **Resizing**: Images are resized to a fixed dimension of 128x128 pixels.
- **Grayscale Conversion**: The images are converted to grayscale to reduce complexity.
- **Normalization**: Pixel values are normalized to a range of [0, 1].

The data is split into:

- **Training Set**: 80% of the dataset
- **Testing Set**: 20% of the dataset

```
COVID: Found 3616 images.
Lung_Opacity: Found 6012 images.
Normal: Found 10192 images.
Viral Pneumonia: Found 1345 images.
Number of samples: 21165
Number of labels: 21165
Unique labels: {'Viral Pneumonia', 'Lung_Opacity', 'Normal', 'COVID'}
```

3. Algorithm and Model Overview:

A **Convolutional Neural Network (CNN)** is used for image classification. The CNN architecture consists of several layers designed to extract features from the images and classify them into four categories.

Model Architecture:

The architecture is defined as follows, with two variations based on the number of filters, dropout rates, and complexity of layers:

Variation 1:

```python
# Build the CNN model
model = Sequential([
    Conv2D(32, (3, 3), activation="relu", input_shape=(img_size[0], img_size[1], 1)),
    MaxPooling2D(pool_size=(2, 2)),
    Conv2D(64, (3, 3), activation="relu"),
    MaxPooling2D(pool_size=(2, 2)),
    Flatten(),
    Dense(128, activation="relu"),
    Dropout(0.5),
    Dense(len(subsets), activation="softmax")
])
```

Explanation:

- **Conv2D Layers**: The first convolution layer uses 32 filters with a 3x3 kernel, followed by a second layer with 64 filters.

- **MaxPooling2D**: These layers reduce the spatial dimensions after each convolution layer to retain only the most relevant features.
- **Flatten**: Converts the 2D feature map into a 1D vector to be passed into dense layers.
- **Dense Layer**: A fully connected layer with 128 neurons, followed by a dropout rate of 50% to avoid overfitting.
- **Output Layer**: The output layer has a softmax activation function to classify the images into one of the four categories (COVID, Lung_Opacity, Normal, Viral Pneumonia).

Variation 2:

```
model = Sequential([
    Conv2D(16, (3, 3), activation='relu', input_shape=(128, 128, 1)),
    MaxPooling2D(pool_size=(2, 2)),

    Conv2D(64, (3, 3), activation='relu', padding='same'),
    MaxPooling2D(pool_size=(2, 2)),
    Dropout(0.25),

    Conv2D(128, (3, 3), activation='relu', padding='same'),
    MaxPooling2D(pool_size=(2, 2)),
    Dropout(0.3),

    Conv2D(128, (3, 3), activation='relu', padding='same'),
    MaxPooling2D(pool_size=(2, 2)),
    Dropout(0.4),

    Flatten(),

    Dense(128, activation='relu'),
    Dropout(0.25),

    Dense(64, activation='relu'),

    Dense(4, activation='softmax')
])
```

Explanation:

1. **Conv2D Layers**: The model uses more filters in the first and second convolutional layers compared to the first model. It starts with 16 filters and increases them to 64 and 128 in subsequent layers.
2. **MaxPooling2D**: Max pooling is used after each convolutional layer to reduce dimensionality.
3. **Dropout**: Various dropout rates (25%, 30%, 40%) are used at different stages to prevent overfitting by randomly setting a portion of the nodes to zero.
4. **Flatten**: Converts the 2D feature map into a 1D vector before passing it to fully connected layers.

5. **Dense Layers**: This model has two dense layers with 128 and 64 neurons, with a dropout rate applied to both to avoid overfitting.
6. **Output Layer**: Like Variation 1, the final dense layer has 4 neurons with softmax activation to classify the image into one of the four categories.

Compilation and Training:

- **Optimizer**: Adam optimizer for gradient descent.
- **Loss Function**: Categorical cross-entropy to handle multi-class classification.
- **Metrics**: Accuracy is used as the evaluation metric.

4. Code Structure:

Main Functions:

- **Load _ data ()**: This function loads images from the dataset directories, resizes, converts them to grayscale, normalizes, and splits the data into training and testing sets.
- **Build _model ()**: Constructs the CNN model with convolutional, pooling, flatten, dense, and dropout layers.
- **train model ()**: Compiles the model and trains it using the training data, validating on the testing data.
- **Predict _ disease ()**: Takes a new image, preprocesses it, and makes a prediction about the disease category (COVID, Lung _Opacity, Normal, or Viral Pneumonia).
- **Evaluate _ model ()**: Evaluates the trained model using metrics like accuracy and precision.
- **Display _ images _ in _ directory ()**: Displays three images from a given directory, along with their predicted labels.
- **Display _ images _in _directory ()**: Displays three images from a given directory, along with their predicted labels.

5. Problem-Solving Approach:

The project follows a standard supervised learning workflow:

1. **Data Preprocessing**:
    a. Resize each image to 128x128 pixels.
    b. Convert images to grayscale.
    c. Normalize the pixel values to a [0, 1] range.
    d. Split the dataset into training (80%) and testing (20%) sets.
2. **Model Construction**:
    a. A CNN architecture is built with two convolutional layers followed by max-pooling layers and dense layers.
    b. Dropout is used to avoid overfitting during training.
    c. The model is compiled using the Adam optimizer and categorical cross-entropy loss function.
3. **Model Training**:
    a. The model is trained for 10 epochs with a batch size of 32.
    b. Training performance is evaluated using the validation dataset (test set).
4. **Prediction and Evaluation**:
    a. For new images, the model predicts whether the image is of a "COVID", "Lung _Opacity", "Normal", or "Viral Pneumonia" category.
    b. The model's performance is evaluated using overall accuracy and precision on the test set.

6. Results and Evaluation:

- **Overall Accuracy**: The CNN model achieved an accuracy of **0.87**(Variation 1) and **0.90**(variation 2) on the test set.

- **Precision**: The CNN model achieved a precision of 0.88(Variation 1) and **0.90**(Variation 2) on the test set.

Classification report:

	precision	recall	f1-score	support
COVID	0.87	0.88	0.88	701
Lung_Opacity	0.81	0.82	0.82	1246
Normal	0.89	0.89	0.89	2024
Viral Pneumonia	0.97	0.88	0.92	262
accuracy			0.87	4233
macro avg	0.88	0.87	0.87	4233
weighted avg	0.87	0.87	0.87	4233

Variation – 1

	precision	recall	f1-score	support
COVID	0.93	0.93	0.93	701
Lung_Opacity	0.90	0.81	0.85	1246
Normal	0.89	0.94	0.91	2024
Viral Pneumonia	0.96	0.95	0.96	262
accuracy			0.90	4233
macro avg	0.92	0.91	0.91	4233
weighted avg	0.90	0.90	0.90	4233

Variation – 2

TEAM MEMBERS:

P CHAITRA SE22UCSE193

P SRIROOP SE22UCSE190

P MADHUKAR SE22UCSE191

P AKSHAY SE22UCSE192

P TEJA REDDY SE22UCSE189