**DOCUMENTATION**

I have chosen a Convolutional Neural Network (CNN) for multi-class classification. The model architecture is as follows:

**model = tf.keras.models.Sequential([**

**data\_augmentation,**

**tf.keras.layers.Conv2D(32, (3, 3), activation='relu', input\_shape=(128, 128, 3)),**

**tf.keras.layers.MaxPooling2D((2, 2)),**

**tf.keras.layers.Flatten(),**

**tf.keras.layers.Dense(256, activation='relu'),**

**tf.keras.layers.Dropout(0.5),**

**tf.keras.layers.Dense(512, activation='relu'),**

**tf.keras.layers.Dense(num\_classes, activation='softmax')**

**])**

* Input Layer: Data augmentation using RandomFlip, RandomRotation, and RandomZoom.
* Convolutional Layer: 32 filters with a kernel size of (3, 3), ReLU activation.
* MaxPooling Layer: Reduces spatial dimensions.
* Flatten Layer: Flattens the output for dense layers.
* Dense Layers: Two fully connected layers with ReLU activation.
* Output Layer: Dense layer with num\_classes units and softmax activation for multi-class classification.

**Training Process:**

* The model is compiled with Adam optimizer and Categorical Crossentropy loss. The training process is executed with the following code:

**model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])**

**y\_train\_one\_hot = to\_categorical(y\_train, num\_classes=5)**

**history = model.fit(x\_train, y\_train\_one\_hot, epochs=100, batch\_size=128, validation\_split=0.1)**

**Critical Findings:**

1. **Data Preparation:**
   * Images are loaded and resized to (128, 128) pixels.
   * Data is split into training and testing sets.
2. **Data Augmentation:**
   * Utilized RandomFlip, RandomRotation, and RandomZoom for augmenting training data.
3. **Normalization:**
   * Image data is normalized using tf.keras.utils.normalize.
4. **Model Architecture:**
   * Convolutional layers followed by dense layers for feature extraction.
   * Dropout is used for regularization.
   * Softmax activation in the output layer for multi-class classification.
5. **Training:**
   * Model is trained for 100 epochs with a batch size of 128.
6. **Evaluation:**
   * Model performance is evaluated on the test set using model.evaluate.
   * Classification report is generated to assess precision, recall, and F1-score.