EDA REPORT

HAND GEASTURE RECOGNITION SYSTEM

Table of Contents

1. Introduction

This report documents the Exploratory Data Analysis (EDA) and Data Augmentation Pipeline used in the Hand Gesture Recognition System.

The objective is to process, analyze, and understand the LeapGestRecog dataset, then prepare it for training a deep learning model capable of recognizing hand gestures in real time.

2. Dataset Description

The project uses the LeapGestRecog dataset downloaded from Kaggle. The dataset contains:

- 10 gesture classes
- 10 subjects
- ~20,000 grayscale images of hands
- Resolutions around 128 × 128
- Images stored in folders by subject, then gesture type

Each gesture corresponds to a unique folder label, for example:

- 01_palm
- 02_L
- 03_fist
- 04_fist_moved

Each image is a single-channel grayscale frame taken under consistent lighting and background conditions.

3. Data Import & Extraction Summary

The notebook loads and extracts images using:

- kagglehub to download dataset
- Standard Python libraries (os, cv2, numpy, PIL) to iterate over images
- A mapping dictionary to encode class names → integer labels
- A metadata DataFrame containing:
    - subject
    - gesture_label
    - label_int

Total collected:

- X_data: NumPy array of gesture images
- y_labels: Integer-encoded classes
- label_lookup: Class mapping dictionary

4. Exploratory Data Analysis (EDA)

4.1 Class Distribution

A seaborn.countplot() was used to visualize the number of samples per gesture class.

All gesture categories are balanced, which is ideal for training.

4.2 Subject Distribution

A second count plot confirmed that:

- Each subject contributed an equal number of gesture samples
- No missing or corrupted batches from specific individuals

This ensures no accidental bias from a single subject dominating the dataset.

4.3 Random Sample Visualization

Random grids of gesture images were displayed using:

- 16 random samples
- Titles showing the decoded gesture class

Insights:

- All images are centered on the hand
- Minimal background noise
- The gestures are visually distinguishable
- Lighting is consistent

4.4 Pixel Intensity Analysis

Histogram plots were generated for each gesture class to inspect pixel distribution.

Findings:

- Images are mostly mid-to-high contrast
- Low noise levels
- No extreme outliers
- Suitable for normalization into 0–1 scale

5. Data Preprocessing Overview

5.1 Image Reshaping

Each image is reshaped into:

(IMG_SIZE, IMG_SIZE, 1)

Where:

- IMG_SIZE = 128
- Single grayscale channel

5.2 Normalization

Pixel values are normalized using:

X = X / 255.0

- This improves training stability for CNN models.

5.3 Train / Validation / Test Split

Split ratios:

- 70% Train
- 15% Validation
- 15% Test

Performed using:

train_test_split(..., stratify=y)

- Ensures equal class proportion in each split.

5.4 One-Hot Encoding

- Labels were converted to categorical format using:

to_categorical(..., num_classes)

6. Data Augmentation Pipeline Documentation

To improve model generalization and reduce overfitting, a Keras ImageDataGenerator was used.
The augmentation parameters were chosen specifically for hand gesture recognition—allowing natural variations while preserving gesture shape.

6.1 Training Data Augmentation Configuration

- Rotation ±10° simulates natural wrist movement
- Shift 10% handles small hand misalignment
- Shear 0.1 introduces mild distortion from real-world movement
- Zoom 20% handles varying hand distance from camera
- No horizontal flip avoids turning left-hand gestures into right-hand ones, which would confuse the model

6.2 Validation / Test Generators

No augmentation is applied—validation and test sets must remain pure for accurate performance evaluation.

6.3 Generator Creation

Batch size: 64 images

7. Final Output Saved

The preprocessed dataset and metadata were saved to:

- processed_data.npz
- metadata.pkl

This allows the model training notebook to load prepared data instantly without re-processing.

8. Summary

This EDA and preprocessing stage ensured:

- The dataset is clean, balanced, and consistent
- Images are normalized and reshaped properly
- Stratified splitting prevents label imbalance
- Augmentation strengthens model robustness
- Final processed data is saved for fast reuse

This pipeline prepares the dataset for building a deep learning-based real-time hand gesture recognition model.

Thanks.

HAND GEASTURE RECOGNITION
SYSTEM TEAM