Preprocessing & Augmentation — Technical Report

Project: Arabic Sign Language (ASL) Image Classification
Prepared by: Abanob Yousry (Documentation)
Team: Youssef (EDA & splitting), Ziad (Preprocessing), Belal (Augmentation)
October 1, 2025

1 Executive Summary

This report outlines the foundational phase of the Arabic Sign Language Recognition project. The primary objective of this phase was to meticulously organize, analyze, and prepare a dataset of Arabic sign language images for training a deep learning classification model.

The process involved three key stages:

- **Dataset Organization:** The raw data was systematically split into training, validation, and test sets. A stratified splitting methodology was employed to ensure a consistent class distribution across the validation and test sets, which is critical for unbiased model evaluation. The files were then physically organized into a directory structure suitable for common deep learning frameworks.
- **Exploratory Data Analysis (EDA):** A thorough analysis of the class distributions was performed for each dataset split. This confirmed that the dataset is well-balanced, with a minimal variance in the number of images per class. Visual inspection of sample images provided qualitative insights into the data's characteristics.
- **Preprocessing & Augmentation Pipeline:** A robust data preprocessing and augmentation pipeline was developed using TensorFlow/Keras. This pipeline handles essential tasks like image resizing and normalization and applies various augmentations—including rotation, flipping, and brightness adjustments—to enhance the model's ability to generalize and prevent overfitting.

The successful completion of this phase yields a clean, well-structured, and augmented dataset, establishing a solid groundwork for the subsequent model development and training phase.

2 Dataset Organization and Splitting

The initial dataset of Arabic Sign Language images was organized to create a standard machine learning workflow with distinct training, validation, and test sets.

2.1 Splitting Strategy

The dataset was divided as follows:

- **Training Set:** A dedicated set of images used exclusively for training the model.
- **Validation & Test Sets:** An initial validation pool was strategically split into two equal halves (50/50) to create the final validation set and test set.

The split was performed using a stratified sampling approach based on the image labels (stratify=y). This technique ensures that the proportional representation of each sign language class is maintained across both the validation and test sets, preventing distributional skew.

2.2 Final Directory Structure

Following the split, all image files were categorized and moved into a hierarchical directory structure. Each primary folder (train, val_final, test_final) contains 32 subfolders, where each subfolder is named after one of the Arabic sign language classes (e.g., ain, al, aleff).

This "image-in-folder" structure is highly compatible with the flow_from_directory method in TensorFlow and PyTorch.

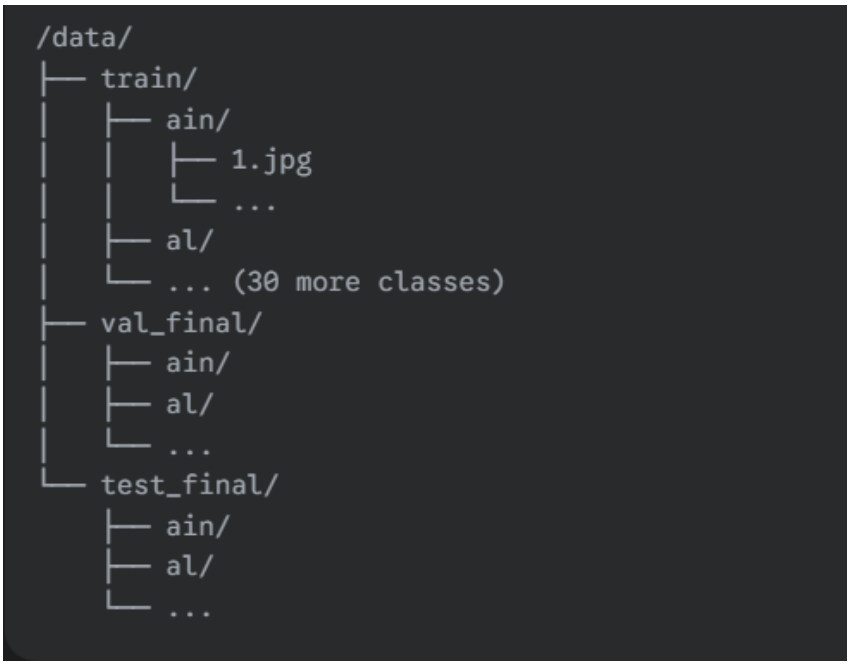


Figure 1: Final directory structure of the dataset

3 Exploratory Data Analysis (EDA)

EDA was conducted to understand the dataset's underlying characteristics and identify any potential biases.

3.1 Class Distribution Analysis

Histograms were generated to visualize the number of images per class for the training, validation, and test sets.

Training Set Distribution: The training dataset is well-balanced. It contains 9,984 images distributed across 32 classes.

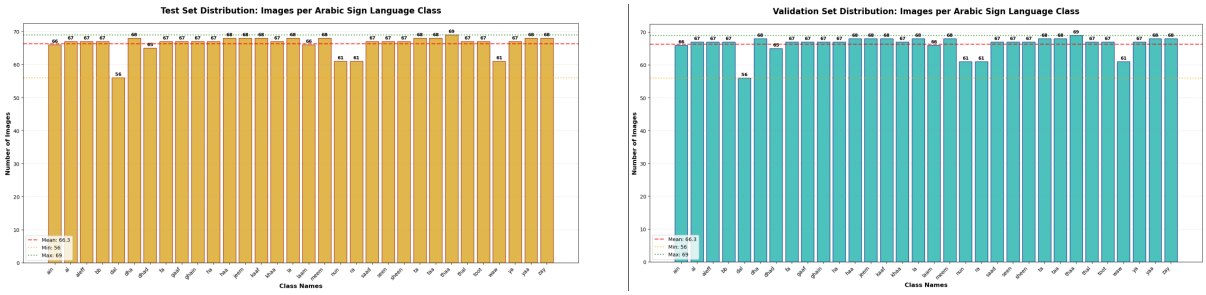
Average Images per Class: 312
Maximum Images per Class: 315
Minimum Images per Class: 290 (for class dal)
Standard Deviation: 7.8 images

This low standard deviation indicates a very consistent number of samples for each class, which is ideal for preventing the model from developing a bias towards more frequently represented classes.

Validation and Test Set Distribution: As a result of the 50/50 stratified split, the validation and test sets are nearly identical in their distribution, each containing 2,130 images.

Average Images per Class: 66.6
Maximum Images per Class: 69
Minimum Images per Class: 56

The distributions are highly consistent, confirming the success of the stratified splitting strategy.



(a) Test Set Distribution (b) Validation Set Distribution

Figure 2: Class distribution histograms

3.2 Sample Image Analysis

A qualitative review of sample images was performed to understand their visual characteristics.

Key observations include:

- Background: The images feature a relatively consistent, non-distracting light-colored background.
- Lighting: Lighting conditions appear uniform across samples, minimizing shadows that could obscure hand gestures.
- Subject: The images focus clearly on the hand and arm, capturing the specific gesture for each sign.
- Pixel Intensity: Histograms of sample images show a broad distribution of pixel values, indicating good contrast and detail without significant over or under-exposure.

4 Data Preprocessing and Augmentation Pipeline

To prepare the images for a neural network and to enhance model robustness, a comprehensive preprocessing and augmentation pipeline was built using `tf.keras.preprocessing.image.Imag`

4.1 Preprocessing

All images loaded into the model will undergo two standard preprocessing steps:

- Resizing: Every image is resized to a uniform dimension (e.g., 224x224 pixels).
- Normalization: Pixel values, which are originally in the range [0, 255], are scaled to a smaller range (typically [0, 1]).

4.2 Data Augmentation

Data augmentation artificially expands the training dataset by creating modified versions of existing images. This exposes the model to a wider variety of visual scenarios, improving its ability to generalize to new, unseen data and significantly reducing the risk of overfitting.

The following augmentations have been implemented:

- Rotation: Randomly rotates images by up to 20 degrees.
- Shifting: Randomly shifts images horizontally and vertically by up to 10% of their dimensions.
- Shearing: Applies a shear transformation of up to 20%.
- Zooming: Randomly zooms into images by up to 20%.
- Flipping: Randomly flips images horizontally.
- Brightness: Randomly adjusts image brightness within a range of [0.8, 1.2].

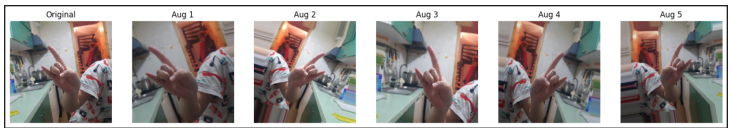


Figure 3: Example of data augmentation effects

5 Conclusion and Next Steps

The data preparation phase of the Arabic Sign Language Recognition project has been successfully completed. The dataset is now cleaned, split, and organized into a structure optimized for deep learning workflows. The exploratory analysis confirms the dataset is well-balanced, and a robust preprocessing and augmentation pipeline is in place to ensure effective model training.

The project is now ready to proceed to the next phase: Model Development and Training. The immediate next steps will be:

- Establish a baseline model using a simple Convolutional Neural Network (CNN).
- Implement and train advanced models using Transfer Learning with pre-trained architectures such as ResNet50 or EfficientNetV2.

- Systematically train the models using the prepared training and validation data generators.
- Evaluate the final trained model on the unseen test set to report its performance on real-world data.