

Title of the Project: Saudi Alphabet Sign Language
Translator: An AI-Powered Solution for Bridging
Communication Barriers

Assignment 6: Report

Name: Yasir Aljarooof

Institution: Midocean University

Advisor: Ashraf Elsayed

1. Dataset Description

- **Dataset Name:** Arabic Sign Language Letters Dataset (ArSL21L)
 - **Source and Contributor:** This dataset is attributed to Munkhjargal Gochoo.
 - **Size:** A total of **14,202 images**.
 - **Contains:**
 - Images include various backgrounds.
 - 32 distinct Arabic sign language letter signs.
 - Collected from 50 different subjects.
 - **Benchmarking:**
 - The dataset has been evaluated using state-of-the-art object detection models (four versions of YOLOv5), with the YOLOv5l model achieving a COCO mAP of 0.83.
 - Additionally, comparison tests against the ArSL2018 dataset have shown superior classification performance when using ArSL21L.
 - **Credits**
 - Munkhjargal Gochoo
 - United Arab Emirates University
 - Published on 11 February 2022
-

2. Literature Analysis and Next Steps

- **Literature Insights:**
 - CNN models are very effective in recognizing static sign language and achieving high accuracy which is making it a very useful tool for bridging communication gaps between deaf and hearing people.
 - LSTM models are very effective in recognizing dynamic sign language and achieving high accuracy.
- **Identified Next Steps:**
 - **Data Pre-processing:**
 - Normalize images and handle variability in background and lighting.
 - Resizing images to a consistent input size.
 - **Data Augmentation:**

- Apply transformations such as rotations, flips, shifts, and zoom to balance the dataset.
 - **Model Exploration:**
 - Since CNN seems to be the most popular choice and has a high accuracy from previous studies, I will choose to develop a custom CNN to learn distinctive features present in the dataset.
 - **Hyperparameter Tuning:**
 - Experiment with the number of convolutional layers, filter sizes, learning rates, and dropout rates.
-

3. Proposed Machine Learning Model and Explanation

- **Model:** Convolutional Neural Network (CNN).
 - **Convolutional Blocks:**
 - **Multiple Stages:** multiple convolutional layers (using kernels such 3×3 or 5×5), followed by a non-linear activation (ReLU) and a pooling layer (max pooling) to progressively extract and down-sample features.
 - **Regularization:** Incorporate Batch Normalization after convolutional layers and Dropout layers to reduce overfitting.
 - **Flattening Layer:** Converts the multi-dimensional feature maps into a one-dimensional feature vector.
 - **Fully Connected Layers:** dense layers to interpret the high-level features extracted by the convolutional layers.
 - **Output Layer:** A softmax layer with 32 neurons corresponding to the number of Arabic sign classes.
 - **Dropout layers between dense layers to mitigate overfitting.**
 - **Simplified Working Explanation:**
 - **Feature Extraction:** The convolutional layers learn hierarchical features (from simple edges to complex shapes) from the raw images.
 - **Dimensionality Reduction:** Pooling layers reduce the spatial dimensions, which cuts down on computational complexity while preserving important features.
 - **Classification:** The dense layers “translate” these features into class probabilities, with the softmax activation ensuring that outputs can be interpreted as probabilities for each class.
-

4. Implementation Timeline (Weeks)

- **Weeks 1–2: Data Preparation**
 - Checking and verifying the dataset and labels.
 - Pre-process images (resizing, normalization, augmentation).
 - Explore the dataset with visualizations to understand class distribution.
 - **Weeks 3–4: Initial Model Development**
 - Design and implement the basic CNN architecture as detailed above.
 - Train on a small subset of the data to validate the architecture's suitability.
 - **Weeks 5–7: Model Refinement & Hyperparameter Tuning**
 - Testing additional convolutional layers.
 - Experiment with different hyperparameters (learning rate, batch size, dropout rates).
 - **Weeks 8–9: Evaluation**
 - Generate performance metrics (accuracy, precision, recall, F1-score) and confusion matrices.
 - **Week 10: Integration into Application Prototype**
 - Integrate the trained CNN into the sign translator application.
 - **Weeks 11–12: Documentation**
 - Finalize the project report detailing the design, experiments, results, and any future recommendations.
-

5. Evaluation Metrics and Measuring Efficiency

- **Performance Metrics:**
 - **Accuracy:** The ratio of correctly classified images to the total number of images.
 - **Precision and Recall:** Per-class assessments to evaluate the model's performance, especially important given any class imbalance.
 - **F1-Score:** The harmonic mean of precision and recall to provide a balanced metric.
 - **Confusion Matrix:** For identifying specific misclassification trends among the 32 classes.

- **Efficiency Metrics:**

- **Inference Time:** Average time taken per image during prediction. This is critical for a real-time sign translator.
- **Resource Utilization:** Evaluation of computational resources during both training and inference.

