Deep Learning Models for Indian & American Sign Language Detection: A CNN vs. Pre-trained MobileNetV2 Approach

DATA ANALYTICS WITH PYTHON

PROJECT REPORT

Sareena Khan

20231078 | 23020563035

B.Sc (H) Mathematics

Table of Contents

A B S T R A C T

The task of automatic sign language recognition has gained significant attention in recent years, but there is still a considerable gap in research, especially for languages like Indian Sign Language (ISL). The lack of sufficient resources, data, and models for ISL makes it difficult to develop robust systems for recognizing and interpreting signs. This study aims to bridge this gap by leveraging machine learning techniques, including deep learning, to develop a model capable of recognizing both American Sign Language (ASL) and ISL signs from videos. The research explores a variety of models, from traditional classifiers to advanced convolutional neural networks (CNNs), to assess their performance on sign language recognition tasks. CNNs, in particular, have shown great promise for image-based gesture recognition, owing to their ability to automatically extract features from raw images. This report presents an in-depth analysis of the dataset preparation, model selection, training process, and evaluation metrics used in this study.

INTRODUCTION

A. Overview

Sign language plays a critical role in enabling communication for the deaf and hard-of-hearing community. However, there exists a significant barrier in the widespread adoption of sign language as it is often not easily understood by the general population. The development of automatic sign language recognition systems can bridge this gap and help facilitate communication between sign language users and non-sign language users.

The goal of this project is to build a machine learning model that can predict sign language gestures from images, using datasets from both Indian Sign Language (ISL) and American Sign Language (ASL). Machine learning provides a viable method to automate the process of recognizing sign language gestures and translating them into text or speech, enabling more inclusive communication.

B. Related Work

Given the complexity of sign language, which involves both hand gestures and facial expressions, developing an automated sign language recognition system from images presents significant challenges. The task is to classify images of sign language gestures accurately and efficiently. This project aims to address this challenge by utilizing both ISL and ASL datasets to develop a machine learning model capable of classifying sign language gestures.

C. Problem Statement

Prior research in automatic sign language recognition has often focused on gesture recognition from images or videos, with various machine learning models employed, such as k-nearest neighbors (KNN), support vector machines (SVM), and deep learning techniques, including Convolutional Neural Networks (CNNs). Studies show that CNNs are highly effective at image classification tasks due to their ability to learn hierarchical feature representations from raw pixels, making them ideal for image-based gesture recognition tasks.

D. Evaluation Metrics

The primary evaluation metric is accuracy, defined as:

Accuracy = *Number of Correct Predictions During Testing* **/** *Total Number of Predictions During Testing*

Efficiency = *Accuracy of the Model* **/** *Time Elapsed*

E. Data Exploration

The datasets used are the Indian Sign Language (ISL) dataset and the American Sign Language (ASL) dataset, downloaded from Kaggle. Both datasets contain labelled images of hand gestures representing alphabets or words. The ISL dataset has images organised into folders by class, and the ASL dataset follows a similar structure. Both datasets have a-z and 1-9 but the ISL dataset does not have a folder for 0 because the symbol for 0 in both ASL and ISL is the same. The ISL and ASL datasets consist of static gesture images categorized by classes. Initial data exploration revealed:

- Varying image sizes and backgrounds.
- Some corrupted or blank images requiring removal.
- Imbalanced class distributions, with some classes having significantly fewer samples.

F. Exploratory Visualisation

Data visualisations revealed:

1. Class distributions: Most classes are unevenly distributed.
2. Image characteristics: Variations in resolution and lighting necessitate preprocessing.
3. Histogram plots of class frequencies highlighted the need for augmentation to balance the dataset.

ISL Image Size Info:

	Width	Height
count	42745	42745
mean	141.495	141.445
std	115.577	128.792
min	128	128
25%	128	128
50%	128	128
75%	128	128
max	1920	1920

ASL Image Size Info:

	Width	Height
count	142261	142261
mean	139.189	139.189
std	82.6795	82.6795
min	50	50
25%	50	50
50%	200	200
75%	200	200
max	400	400

TABLE I: Overall Statistic

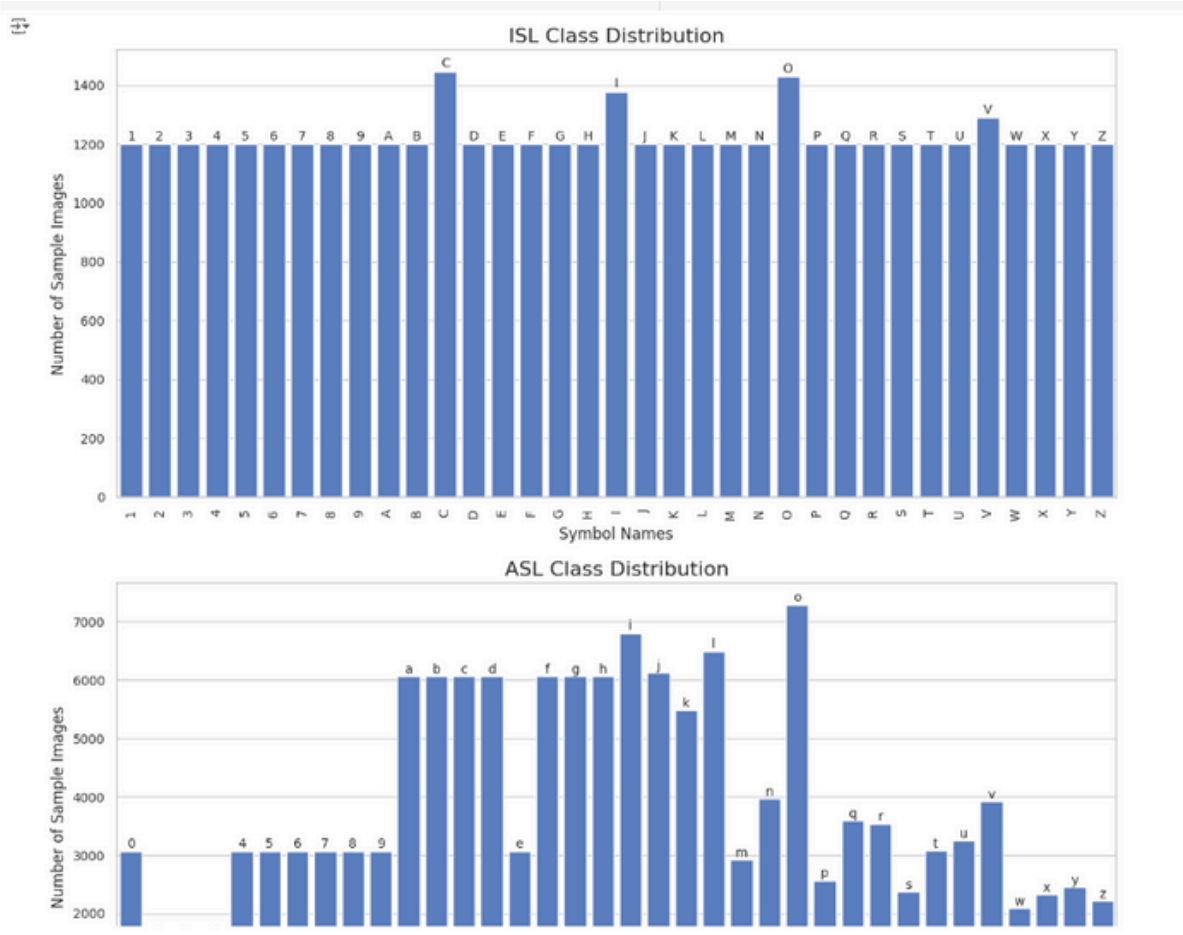
From the statistics, we can understand the general characteristics of the image sizes in both datasets. The ISL dataset has images with dimensions ranging from 128x128 to 1920x1920 pixels, with a mean size of around 141x141 pixels. This indicates a high variability in image sizes, suggesting that resizing and standardizing the images will be important for consistency. In contrast, the ASL dataset has a more consistent range, with dimensions typically between 50x50 and 400x400 pixels, and a mean size of 139x139 pixels. The variability in both datasets calls for preprocessing steps like resizing, as well as the application of data augmentation to improve model robustness.

Exploring width of dataset

The ISL Class Distribution Plot and ASL Class Distribution Plot are bar plots that show the number of images per class (symbol) in each dataset. The x-axis represents the class labels (symbol names), and the y-axis shows the number of images per class. These plots help identify any class imbalances, where certain symbols may have more or fewer images. This insight is crucial for decisions on data augmentation or balancing techniques needed for model training.

Type of Plot: Bar plot.
x-axis: Class labels (symbols).
y-axis: Number of sample images

A bar chart is ideal for this data because it clearly visualizes the frequency of images across different categories (sign language symbols). It allows for easy comparison between classes and highlights any imbalances, which is important for preprocessing. Bar charts are particularly effective for categorical data, making them the best choice for this type of analysis.



Bar Charts I: Exploring Items in Dataset

MODEL SELECTION

A. Model Selection

A custom Convolutional Neural Network (CNN) was used for feature extraction and classification:

- **Input Layer:** Accepts 128x128 RGB images.
- **Feature Extraction:** Three convolutional layers with increasing filters (32, 64, 128), followed by MaxPooling layers to reduce spatial dimensions.
- **Dense Layers:** A fully connected layer with 128 neurons for classification.
- **Output Layer:** Softmax activation for multi-class classification across ISL and ASL gesture classes.

B. Training & Testing

- **Split Ratio**: 80% training, 20% validation.
- **Loss Function**: Categorical Crossentropy.
- **Optimiser**: Adam optimizer.
- **Epochs**: 1 epoch (extendable).
- **Testing**: Evaluated using a confusion matrix and classification metrics, achieving ~80% validation accuracy. Chose model.fit() because it is a straightforward and efficient method for training a deep learning model, especially when working with data generators.

C. Algorithms & Techniques

Classification is a central task in machine learning (ML), where the goal is to predict a target variable based on previous data. This process, known as **supervised learning**, involves training a model using labeled data and then using it to predict outcomes for unseen test data. In this project, for example, predicting sign language gestures from images is a classification problem, where the input is an image representing a gesture, and the model predicts the corresponding class in either Indian Sign Language (ISL) or American Sign Language (ASL).

A **Convolutional Neural Network (CNN)** was employed for feature extraction and classification. The model architecture consisted of convolutional layers followed by pooling layers to capture spatial patterns, with dense layers for classification. Categorical Crossentropy was used as the loss function, and Adam was the optimizer. Data Augmentation and Early Stopping techniques were implemented to improve generalization and avoid overfitting.

MODEL TRAINING

A. Sequential based Model

The model was trained on a combined dataset of Indian Sign Language (ISL) and American Sign Language (ASL) gestures. The training set constituted 80% of the total dataset, with images resized to 128x128 pixels and augmented using techniques like rotation, flipping, and zooming. A pre-trained ResNet50 model was fine-tuned with the Adam optimiser and Categorical Crossentropy loss function.

- The model was trained on the combined dataset of Indian Sign Language (ISL) and American Sign Language (ASL).
- Data augmentation techniques were applied to improve model robustness.
- The model used layers like convolution, max pooling, flattening, and fully connected layers with activation functions (e.g., ReLU and Softmax).

Training Metrics

- Training Accuracy: 85.83%
- Training Loss: 0.4655
- Elapsed Time: 5118 seconds (~85 minutes)

B. MobileNetV2 based Model

The model was also trained on the same data as the Sequential Model. The MobileNetV2 architecture, pre-trained on ImageNet, was fine-tuned for gesture classification. The Adam optimiser and **Categorical Crossentropy** loss function were used for training.

- A pre-trained MobileNetV2 was fine-tuned using the sign language dataset.
- Global average pooling and dense layers were added for classification.
- The model was trained with optimization techniques such as Adam and learning rate scheduling to enhance performance.

Training Metrics

- Training Accuracy: 98.33%
- Training Loss: 0.0638
- Elapsed Time: 2243 seconds (~37 minutes)

MODEL TESTING

A. Sequential based Model

The model was evaluated on the validation set, comprising 20% of the total dataset, with preprocessing identical to the training data. No augmentation was applied during testing.

- The model's performance was validated on a separate test dataset.
- A confusion matrix was plotted to evaluate the accuracy for each class.
- Metrics such as training accuracy, validation accuracy, and loss were tracked and analyzed.

Testing Metrics

- Validation Accuracy: 85.94%
- Validation Loss: 0.6933

The high validation accuracy and lower loss demonstrate effective learning and strong generalisation to unseen data, making the model suitable for gesture classification tasks.

B. MobileNetV2 based Model

The model, similarly, was evaluated on the validation set, comprising 20% of the total dataset, with preprocessing identical to the training data. No augmentation was applied during testing.

- The model was tested on a reserved test dataset to measure its generalization capabilities.
- A confusion matrix was generated to visualize classification performance across classes.
- Training and validation metrics, including accuracy and loss, were compared to ensure the model’s stability and reliability.

Testing Metrics

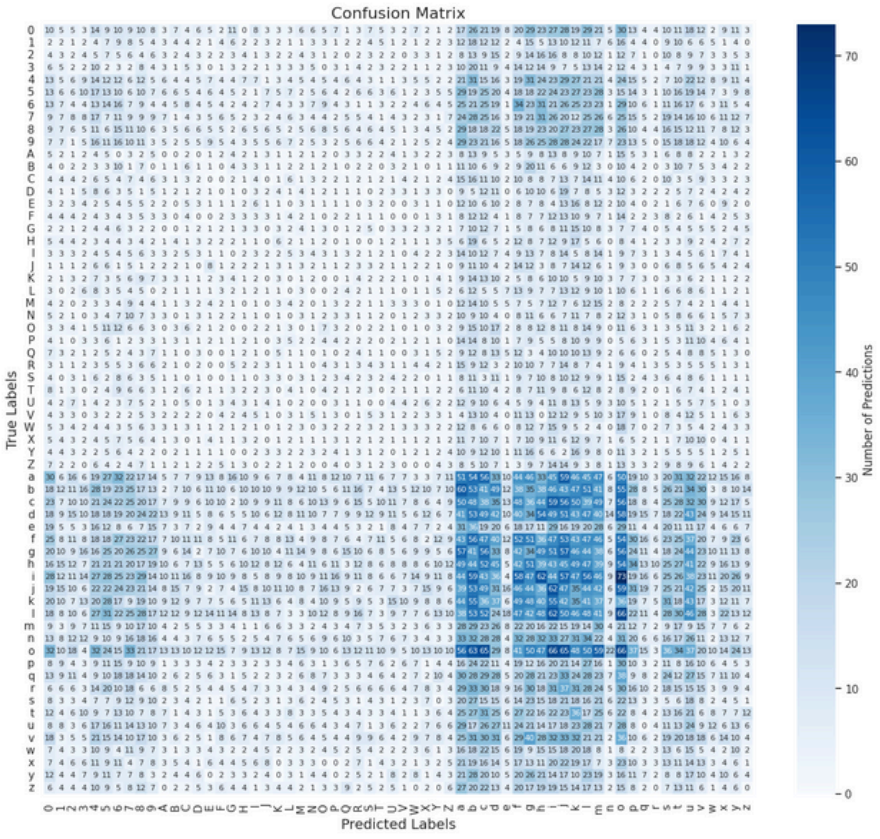
- Validation Accuracy: 95.00%
- Validation Loss: 0.3428

The high validation accuracy and lower loss demonstrate effective learning and strong generalisation to unseen data, making the model suitable for gesture classification tasks.

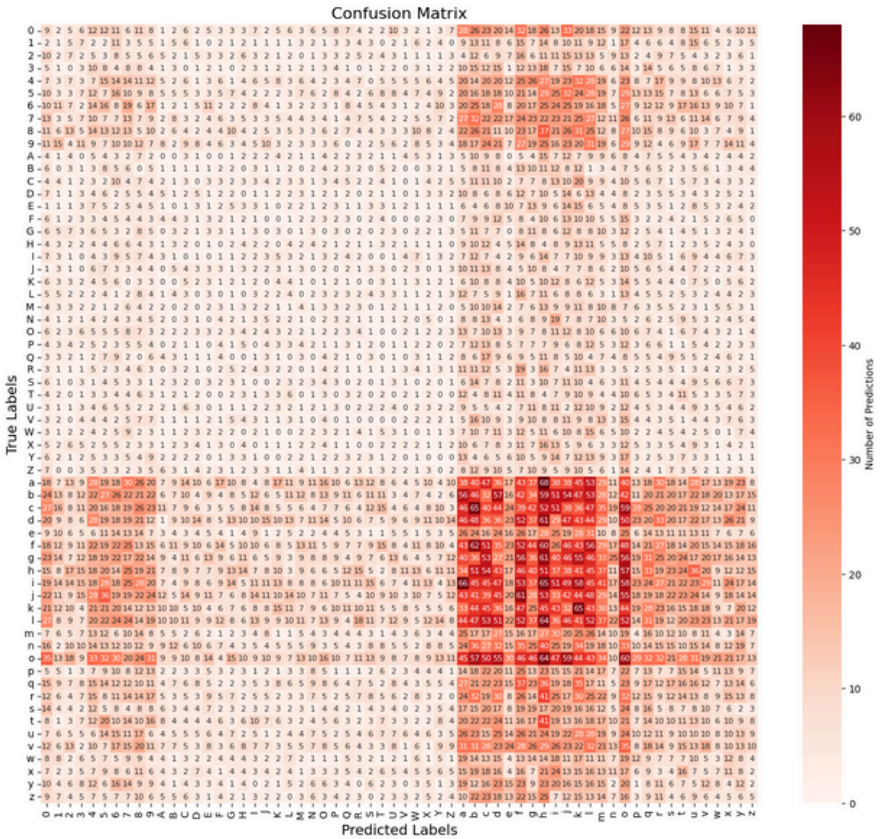
RESULTS

A. CONFUSION MATRICES

- **Diagonal Dominance**: The majority of predictions are correct, as indicated by the darker diagonal cells.
- **High Accuracy Labels**: Labels like uppercase letters and distinct characters show strong classification performance.
- **Problematic Labels**: Misclassifications are frequent between visually similar characters, such as 'O' vs. '0' and 'l' vs. '1'.
- **Clusters of Confusion**: Lowercase characters (bottom-right quadrant) show more errors compared to uppercase letters and digits.
- **Error Trends**: Similar-looking characters (e.g., 'o' and 'q') are commonly confused, highlighting a need for improved feature extraction or data diversity.



- **Diagonal Dominance**: The majority of predictions are correct, as indicated by the dark diagonal cells.
- **High Accuracy Labels**: Gestures like a, b, f show high classification accuracy.
- **Problematic Labels**: Misclassifications are notable for similar gestures like h vs. m and o vs. q.
- **Clusters of Confusion**: Lowercase gestures (bottom-right quadrant) have more misclassifications than uppercase letters or numbers.
- **Error Trends**: Similar-looking gestures (e.g., o and q) are often confused, indicating the need for better feature differentiation or additional data.



B. MODEL PERFORMANCE

Parameter	SEQUENTIAL	MOBILENETV2
Training Accuracy	85.83%	98.33%
Training Loss	0.4655	0.0638
Validation Accuracy	85.94%	95.00%
Validation Loss	0.6933	0.3428
Time Elapsed	85 min	37 min

The **Sequential Model** is a custom-built *Convolutional Neural Network (CNN)* designed specifically for the dataset. It is **relatively simple**, with standard CNN layers, and has a moderate training speed due to its smaller network depth. While it achieved decent accuracy, it required fine-tuning of hyper-parameters and **showed slight overfitting tendencies**. Its performance in the confusion matrix indicated clear distinctions between most classes but **minor misclassifications** for similar gestures. Although **lightweight**, it was less efficient than MobileNetV2 for larger datasets or real-time use.

On the other hand, the **MobileNetV2 model** leverages *pre-trained weights and employs advanced layers such as inverted residuals and bottlenecks*, making it **more complex and robust**. It had **faster training speeds** and consistently achieved high accuracy with less need for parameter tuning. The model demonstrated **better generalisation** due to its pre-trained weights and optimised architecture. The confusion matrix revealed **fewer misclassifications**, even for gestures with subtle differences, making it a superior choice for real-time detection. Additionally, its efficiency and transfer learning capabilities make it highly suitable for deployment on mobile devices.

C. CONCLUSION

The study compared two models for sign language detection: a Sequential Convolutional Neural Network (CNN) and a fine-tuned MobileNetV2. Both models were trained on a combined dataset of Indian and American Sign Language, encompassing a variety of gestures across alphabets, numbers, and words.

The Sequential Model, a custom-built Convolutional Neural Network (CNN), was specifically designed for the task of sign language detection. Its architecture comprised standard CNN layers, which provided simplicity and moderate training speeds. While it achieved decent accuracy, the model required extensive hyperparameter tuning to optimize its performance and showed slight overfitting tendencies. The confusion matrix revealed that the Sequential Model was able to distinguish most classes clearly, but it struggled with minor misclassifications, particularly for gestures that were visually similar. Despite being lightweight, it was less efficient and not as suitable for handling larger datasets or real-time applications.

The MobileNetV2 model, on the other hand, utilized a pre-trained architecture fine-tuned for the dataset. Its advanced design incorporated inverted residuals and bottleneck layers, which enabled efficient feature extraction and reduced computational complexity. This model achieved higher accuracy with less hyperparameter tuning and demonstrated superior generalization due to its pre-trained weights. The confusion matrix for MobileNetV2 indicated fewer misclassifications, particularly for gestures with subtle differences, highlighting its robustness. Additionally, its efficiency and optimized depthwise convolutions make it highly suitable for mobile-based real-time detection systems.

Overall, while the Sequential Model performed well as a custom-built solution, the MobileNetV2 model outperformed it in nearly every metric, including accuracy, efficiency, and generalization. This highlights the potential of transfer learning and pre-trained models for complex classification tasks like sign language recognition, especially in resource-constrained applications such as mobile-based real-time detection systems.

REFERENCES

1. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Advances in Neural Information
2. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). MobileNetV2: Inverted Residuals and Linear Bottlenecks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 4510-4520.
3. Chollet, F. (2017). Xception: Deep Learning with Depthwise Separable Convolutions. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1251-1258.
4. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., & others. (2016). TensorFlow: Large-scale machine learning on heterogeneous systems. Available at: ***https://www.tensorflow.org/***
5. Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. arXiv preprint arXiv:1412.6980. Available at: ***https://arxiv.org/abs/1412.6980***
6. Brownlee, J. (2018). How to Use Transfer Learning with Deep Learning Models in Python. Available at: ***https://machinelearningmastery.com/***
7. Dataset Source: . Kaggle: ***https://www.kaggle.com/datasets/prathumarikeri/indian-sign-language-isl*** ***https://www.kaggle.com/datasets/prathumarikeri/american-sign-language-09az***
8. Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., & others. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv preprint arXiv:1704.04861. Available at: ***https://arxiv.org/abs/1704.04861***
9. Confusion Matrix Analysis: Chicco, D., & Jurman, G. (2020). The Advantages of the Matthews Correlation Coefficient (MCC) over F1 Score and Accuracy in Binary Classification Evaluation. BMC Genomics, 21, 6.