HandTalk: Translating Sign language to Text

Aakash - 2021002
Parveen - 2021079
Shubham Sharma - 2021099
Pourav Surya - 2021271

INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY
DELHI

Motivation

Background:

Many disabled people worldwide rely on sign language as their primary mode of communication. However, the lack of widespread understanding of sign language among non-users creates significant challenges in education, healthcare, and workplaces.

Purpose of the Project:

HandTalk addresses these barriers by creating a system capable of detecting and translating sign language gestures into alphabets, these alphabets can be later streamed together enabling smoother communication.

Why Machine Learning?

ML, particularly CNNs, has demonstrated high efficiency in image-based tasks. This makes it an ideal approach for static gesture recognition.

Literature Review

- A New Benchmark on American Sign Language Recognition using Convolutional Neural Network: This study proposes a novel convolutional neural network (CNN) model to enhance American Sign Language (ASL) recognition accuracy. Evaluated on four publicly available ASL datasets, the model, applied to alphabet and numeral images, achieves a 9% improvement in accuracy over existing methods

- Real-Time Sign Language Detection Using CNN : To detect real-time sign language, a dataset is prepared on which a customized CNN model is trained. In the findings, it was observed that the customized CNN model can achieve the highest 95.6% accuracy.

- Deep convolutional neural networks for sign language recognition : To address the lack of mobile selfie sign language datasets, they created one with five subjects performing 200 signs from five angles and various backgrounds, capturing 60 frames per sign. Their approach achieved a 92.88% recognition rate

Relevance to HandTalk:

These studies emphasize the effectiveness of CNNs for static gesture recognition, guiding our choice to implement CNN-based models in this project.

Dataset Description

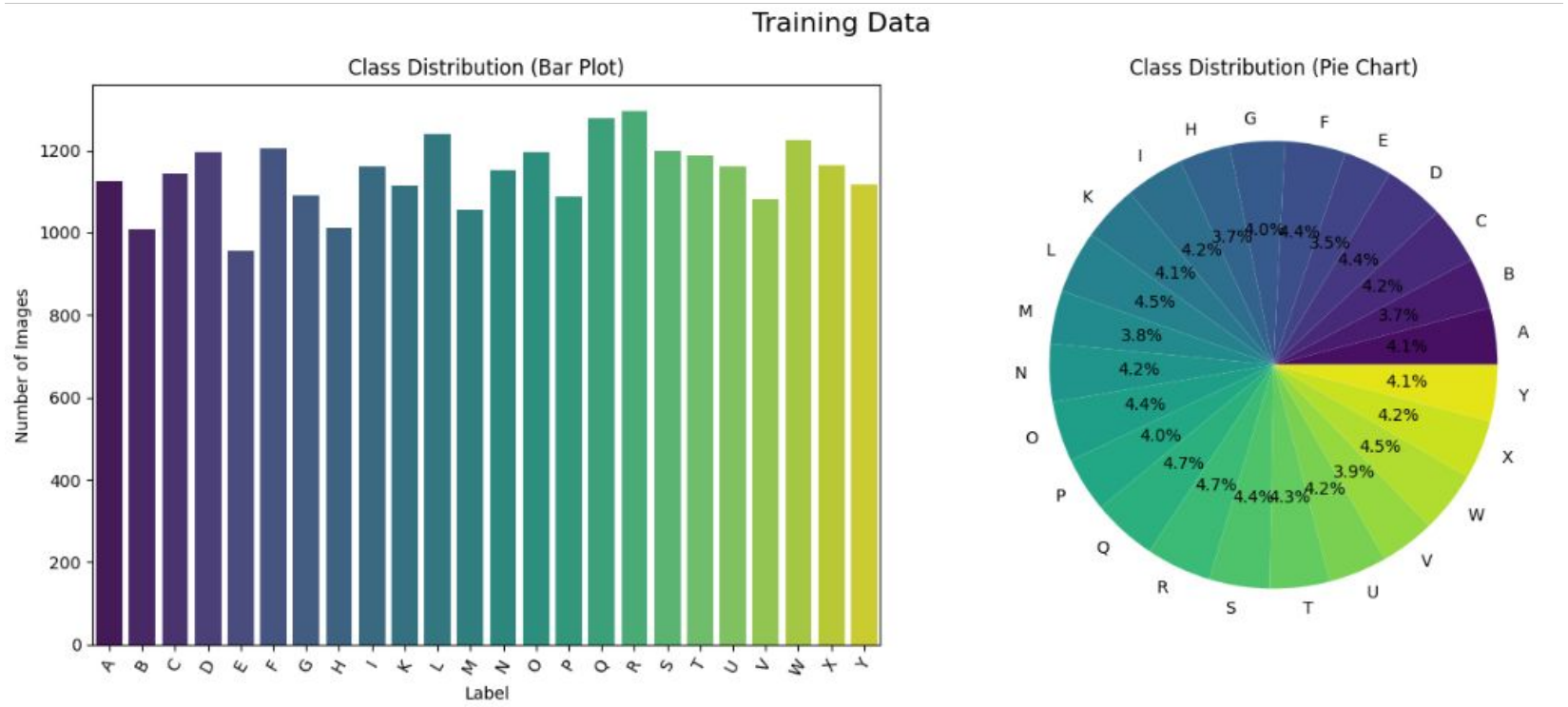
Dataset Used: Sign Language MNIST

- Contains grayscale images of 28x28 pixels representing 24 alphabets (excluding ‘J’ and ‘Z’ due to their dynamic nature).
- Total Classes: 24
- Training samples : 27,455
- Testing Samples : 7,172
- Balanced Dataset

Preprocessing Steps:

- Pixel values normalized to a range of [0, 1] for consistency.
- Dimensionality reduced using PCA from 784 features to 50, optimizing model performance and reducing training time.

Data Visualization



Methodology

Exploration of Algorithms:

The major methodology part consists of applying different model on the trained data, calculating and analyzing their accuracy, hyperparameter tuning wherever needed and finding the model that performs best in terms of accuracy or loss. The models used for classification purpose were Logistic Regression, Decision Tree, Random Forest, Perceptron, Multi Layer perceptron, Support Vector Machines, Convolution Neural Network.

Model Training and Evaluation:

Each model is trained on the best parameter found using grid search cross validation with 3 fold cross validation, and accuracy, precession, recall and F1 score is calculated and analyzed.

Feature Selection:

PCA was applied for dimensionality reduction from 784 pixels to 50 pixel value, aiding in faster and more efficient training.

Methodology

CNN Architecture Details:

Layer Type	Parameters
Conv2D (1)	Filters: 32, Kernel: (3, 3), ReLU, Input (28,28,1)
MaxPool2D (1)	Pool size: (2, 2), Strides: 2
Conv2D (2)	Filters: 128, Kernel: (3, 3), Activation: ReLU
MaxPool2D (2)	Pool size: (2, 2), Strides: 2
Conv2D (3)	Filters: 256, Kernel: (3, 3), Activation: Tanh
MaxPool2D (3)	Pool size: (2, 2), Strides: 2
Dense (1)	Units: 64, Activation: ReLU
Dense (2)	Units: 128, Activation: Tanh
Dense (3)	Units: 128, Activation: Tanh
Output Layer	Units: 25, Activation: Softmax
Optimizer	Adam, Learning Rate: 0.0005
Loss Function	Sparse Categorical Crossentropy
Metrics	Accuracy

Results

Logistic Regression

Hyperparameter	Values (Grid Search)	Best Value
penalty	{'l1', 'l2', 'elasticnet', None}	'l1'
C	{0.01, 0.1, 1, 10}	0.1
solver	{'saga'}	'saga'
max_iter	{100, 200}	200

Random Forest

Hyperparameter	Values (Grid Search)	Best Value
n_estimators	{50, 100, 200}	200
criterion	{'gini', 'entropy'}	'gini'
max_depth	{None, 10, 20}	20
min_samples_split	{2, 10}	2
min_samples_leaf	{1, 5}	1

Decision Tree

Hyperparameter	Values (Grid Search)	Best Value
criterion	{'gini', 'entropy'}	'entropy'
splitter	{'best', 'random'}	'best'
max_depth	{None, 10, 20, 30}	None
min_samples_split	{2, 10, 20}	2
min_samples_leaf	{1, 5, 10}	1

Perceptron

Hyperparameter	Values (Grid Search)	Best Value
penalty	{'l1', 'l2', 'elasticnet', None}	'l1'
alpha	{0.0001, 0.001, 0.01}	0.001
max_iter	{1000}	1000
tol	{1e-3}	0.001

Multi-Layer Perceptron

Hyperparameter	Values (Grid Search)	Best Value
hidden_layer_sizes	{(128,), (128, 64), (256, 128)}	(256, 128)
activation	{'relu', 'tanh'}	'relu'
solver	{'adam', 'sgd'}	'adam'
alpha	{0.0001, 0.001}	0.001
learning_rate	{'constant', 'adaptive'}	'constant'
max_iter	{500}	500

Support Vector Machine (SVM)

Hyperparameter	Values (Grid Search)	Best Value
C	{0.1, 1, 10}	0.1
kernel	{'linear', 'rbf'}	'linear'
gamma	{'scale', 'auto'}	'scale'

Performance Metrics

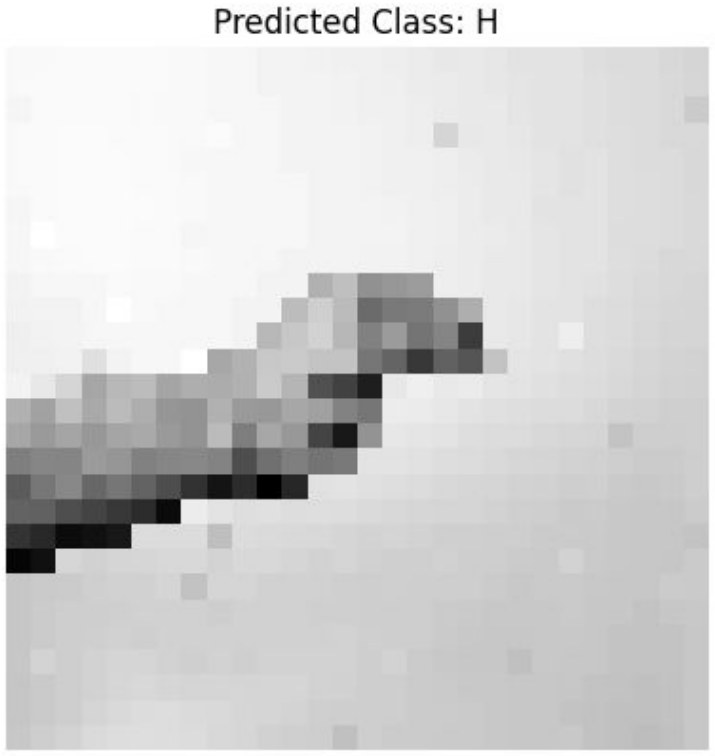
All the models with their best parameters are trained on the train dataset and then were test on the testing dataset. For all the models accuracy, precision, recall and F1-score are calculated on the test dataset and summarized in the below table.

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	0.693809	0.724626	0.693809	0.698777
Decision Tree Classifier	0.478667	0.498118	0.478667	0.483469
Random Forest Classifier	0.811907	0.835269	0.811907	0.815962
Perceptron	0.265477	0.651270	0.265477	0.298552
MLP Classifier	0.761433	0.766137	0.761433	0.758345
SVM	0.8419	0.8568	0.8419	0.8444
CNN (Sequential)	0.9352	0.9376	0.9352	0.9347

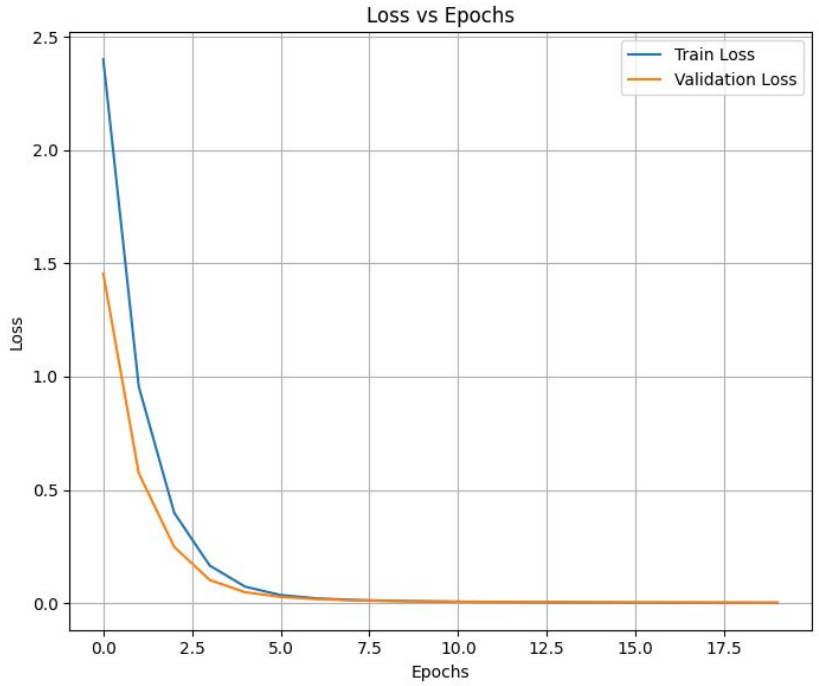
It can be inferred that CNN is the best model, outperforming other models in terms of accuracy, precision, recall and F1 score.

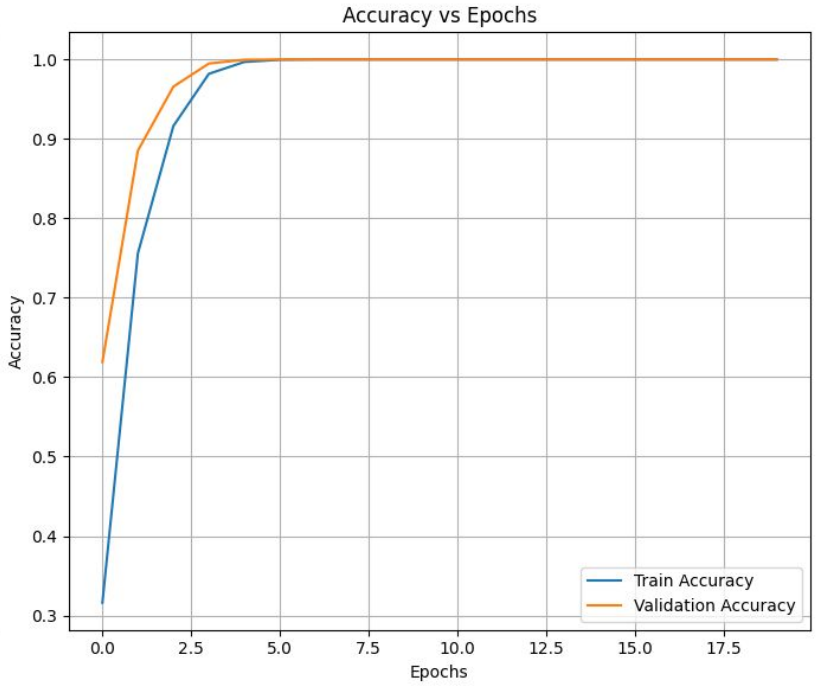
Prediction Result

Predicted image with the best model: CNN



Conclusion





Insights:

CNN outperformed other models due to its hierarchical feature extraction, enabling effective recognition of spatial patterns in images.

Accuracy vs. Epoch and Loss vs. Epoch graphs demonstrate the model's convergence and learning efficiency.

Timeline

- Timeline Adherence:

  Initial proposal and literature review completed on schedule.

  Slight delays in data preprocessing and model training due to hyperparameter tuning, However all tasks were completed by the end.

- Week 1 (28th August - 3rd September): Gather and preprocess the sign language dataset. Convert images to grayscale and normalize the pixel values. Review relevant literature on sign language recognition and machine learning techniques.
- Week 2-3 (4th September - 17th September): Conduct exploratory data analysis (EDA) to understand the dataset. Visualize the data and identify any patterns or trends in the sign language gestures.
- Week 4-7 (18th September - 22nd October): Develop initial models using Logistic Regression, Decision Trees, and Support Vector Machines (SVM). Train models on the dataset and evaluate initial performance.
- Week 8 (30th October - 5th November): Experiment with advanced machine learning algorithms, such as Random Forests and Multi-Layer Perceptron (MLP). Optimize the models through hyperparameter tuning using techniques like Grid Search or Random Search.
- Week 9 (5th November - 12th November): Evaluate model performance on the test dataset using metrics such as accuracy, precision, recall, and F1 score. Compare different models to determine the best-performing one.
- Week 10-11 (13th November - 20th November): Incorporate PCA and also explore the CNN model for better results.
- Week 11-12 (21st November - 27th November): Test the final system and finalize the project report, documenting the methodology, results, and conclusions. Prepare the project presentation and demo for submission.

Individual team members' contributions

- Aakash: Decision Tree, Hyperparameter tuning, Dataset processing, CNN, PCA
- Parveen: Logistic Regression, Perceptron, Hyperparameter tuning, Dataset processing, SVM, PCA
- Shubham Sharma: EDA, Random Forest, Hyperparameter tuning, CNN, PCA
- Pourav Surya: MLP classifier, Confusion matrix, Testing on data set, CNN

THANK YOU