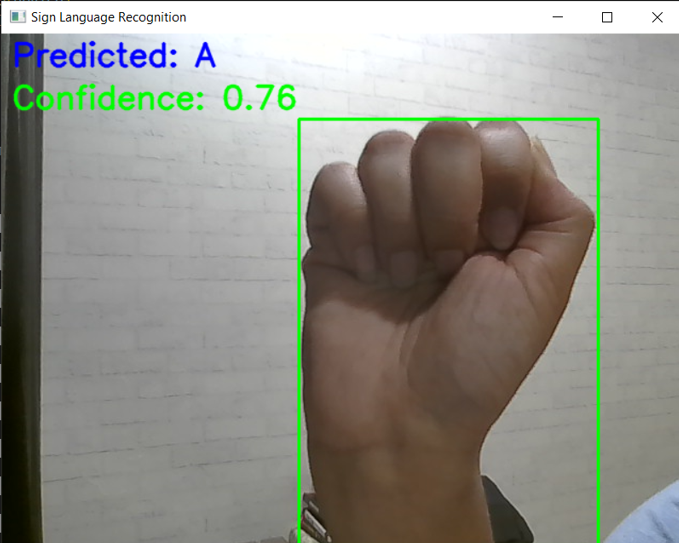
**SIGN LANGUAGE RECOGNITION**

PROBLEM STATEMENT

Sign language is a vital mode of communication for the hearing impaired, enabling them to interact with the world around them.

This project involves building a machine learning model to recognize American Sign Language (ASL) alphabets. The model is trained using a dataset from Kaggle – Sign Language MNIST, which contains images of hand gestures representing the ASL alphabets.

The primary goal is to develop a system that can accurately identify these hand gestures in real-time using a webcam and translate them into corresponding text along with the accuracy score of prediction to facilitate better communication and inclusivity.



DATA

Dataset used:

Kaggle’s sign language MNIST which includes hand gestures for recognition tasks.



Data Loading and Preprocessing:

1. Loading data from CSV files of dataset into numpy dataframes.

2. Separating features and labels and reshaping and normalizing the data.

3. Converting data to Tensor for training model.

4. Applying random transformations for data augmentation.

MODEL DETAILS

Convolutional neural network (CNN) consisting of:

1. two convolutional layers

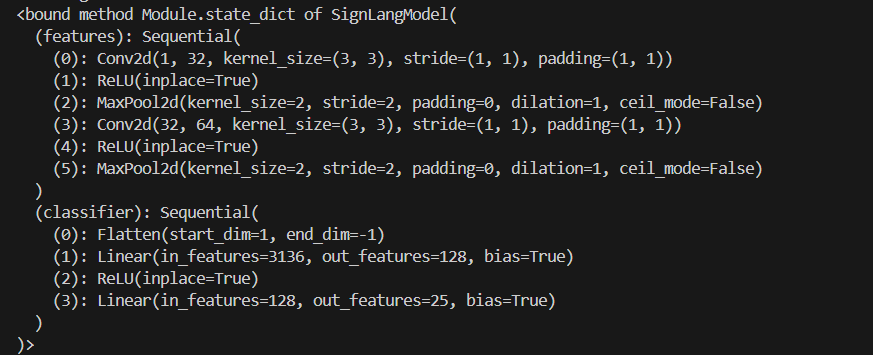
2. two max pooling layers

3. flatten layer

4. two fully connected layers

Summary of layers:

1. **Input Layer**: Accepts grayscale images of size 28x28.
2. **Conv Layer 1**: 32 filters of size 3x3, stride 1, padding 1, followed by ReLU activation.
3. **Max-Pooling Layer 1**: Pool size 2x2, stride 2.
4. **Conv Layer 2**: 64 filters of size 3x3, stride 1, padding 1, followed by ReLU activation.
5. **Max-Pooling Layer 2**: Pool size 2x2, stride 2.
6. **Flatten Layer**: Converts the 2D feature maps into a 1D feature vector.
7. **Fully Connected Layer 1**: 64 \* 7 \* 7 input features, 128 output features, followed by ReLU activation.
8. **Fully Connected Output Layer**: 128 input features, num\_classes (26 for ASL alphabets) output features.

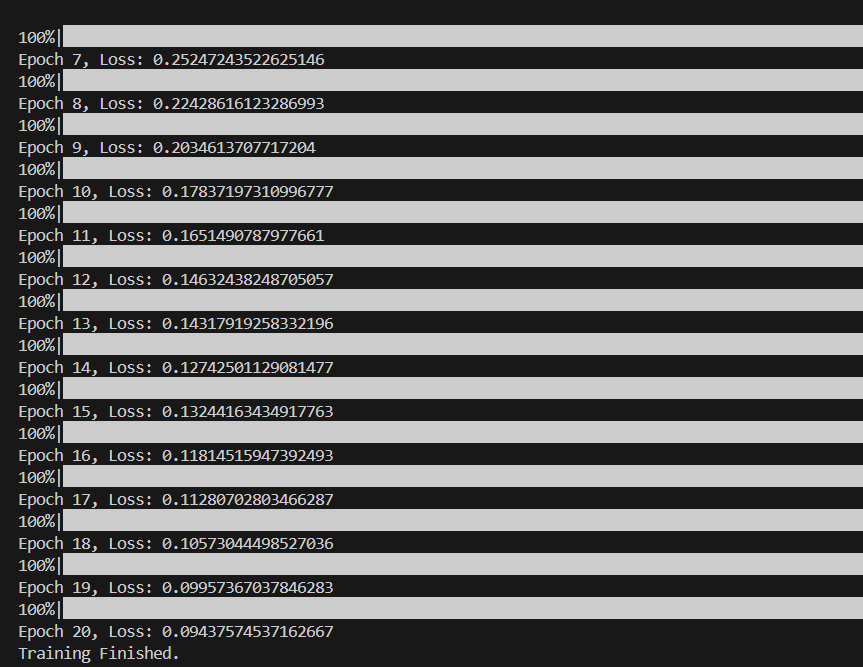


TRAINING MODEL

Using cross entropy loss function and Adam optimizer with the learning rate of 0.001 model was trained for 20 epochs.

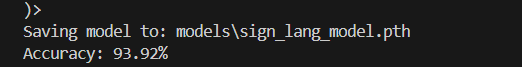
Loss at each epoch is calculated for understanding areas of improvement in model.

Saving model’s state dictionary to use it in real time prediction.



EVALUATION

Accuracy: 93.92%



INFERENCE WITH LOADED MODEL

The pre-trained model is leveraged to make real-time predictions on a live video feed using the OpenCV library.

1. Live Video Feed Acquisition:

The OpenCV library captures the live video feed from the webcam, providing a continuous stream of frames for real-time processing.

2. Region of Interest (ROI) Detection:

A fixed region of interest (ROI) is defined within each frame. This region is highlighted with a rectangle, guiding the user to place their hand within this area for sign language recognition.

3. Image Processing:

The ROI is extracted from the frame and converted from BGR to RGB format.

The extracted image is then resized and converted to a grayscale image to match the input requirements of the model.

Further preprocessing involves normalizing the image to ensure consistency with the training data.

4. Model Inference:

The preprocessed image is fed into the loaded model. The model processes the input and generates a set of class scores (logits) for each alphabet.

These logits are transformed into probabilities using the softmax function, and the class with the highest probability is selected as the predicted label.

5. Prediction Queue:

A queue is maintained to store the recent predictions.

The most frequently occurring prediction in the queue is considered the final output.

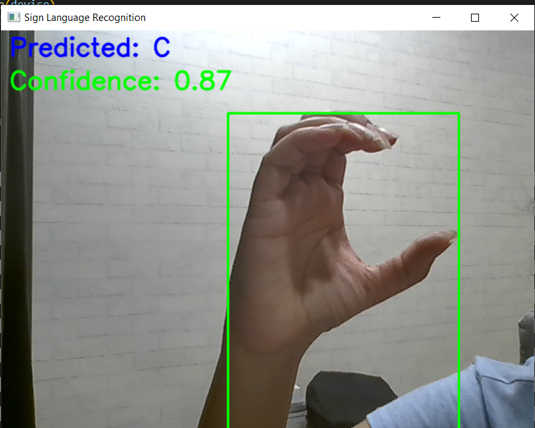
6. Displaying Results:

The predicted alphabet and its associated accuracy score are displayed on the video feed, providing immediate feedback to the user.

The accuracy of the predictions is continuously updated and shown on the screen.

7. User Interaction:

The system allows for seamless interaction, enabling the user to stop the inference process by pressing a designated key (e.g., 'b').



FUTURE DEVELOPMENT

1. Improving model’s real time prediction accuracy.
2. Including more hand gestures for sign language.
3. Extending the project to other languages.
4. Developing an app to provide user interface.