Indian Sign Language Recognition using Convolutional Vision Transformers

A Project Report Submitted in Partial Fulfillment of the Requirements for the

8th Semester B.Tech Project

Submitted by

Ankan Dutta	2112087
Spandan Priyam Chetia	2112188
Mwkthangsa Daimari	2112140

Under the guidance of

Dr. Suganya Devi
Associate Professor

Department of Computer Science & Engineering
NATIONAL INSTITUTE OF TECHNOLOGY SILCHAR
Assam

May, 2025

"Character cannot be developed in ease and quiet. Only through experience of trial and suffering can the soul be strengthened, ambition inspired, and success achieved."

— Helen Keller

ABSTRACT

Indian Sign Language (ISL) serves as a vital communication medium for approximately 5 million deaf and hard-of-hearing individuals in India, yet remains under-researched and lacks technological support compared to other sign languages. The absence of standardized ISL recognition tools creates significant barriers to education, employment, and social integration for this community. To address this critical gap, this research presents a novel ISL recognition system that integrates Convolutional Vision Transformers (CvT) with landmark-based preprocessing techniques. Unlike traditional approaches that struggle with ISL's regional variations and complex spatial-temporal gestures, our system leverages CvT's hybrid architecture to capture both local feature details through convolutional operations and global contextual relationships via transformer self-attention mechanisms. Using a meticulously curated dataset of 150 distinct ISL classes with approximately 36,000 images, we implement a multi-stage preprocessing pipeline incorporating edge enhancement and MediaPipe hand landmark extraction to improve model focus on critical gesture components. Experimental results demonstrate that our CvT-based approach achieves 91.66% accuracy on ISL classification tasks, significantly outperforming conventional CNN architectures (ResNet: 81.67%, VGG16: 83.13%) and standalone Vision Transformers (90.94%). The system's real-time application interface incorporates gesture stability detection and semantic word combinations, demonstrating practical viability for bridging communication divides and fostering greater inclusion for India's deaf community.

ACKNOWLEDGEMENT

We extend our heartfelt gratitude to our esteemed supervisor, **Dr. Suganya Devi**, Department of Computer Science and Engineering, National Institute of Technology Silchar, for her invaluable guidance, encouragement, and unwavering support throughout this project. Her insightful suggestions, constructive feedback, and collaborative spirit have been instrumental in steering this endeavor toward its successful completion.

We deeply appreciate her patience and dedication, which not only facilitated the thorough investigation of our research but also provided us with the opportunity to explore and understand a multitude of new concepts. Without her steadfast mentorship, this project would not have reached its fruition with the quality and timeliness it achieved.

Ankan Dutta	2112087
Spandan Priyam Chetia	2112188
Mwkthangsa Daimari	2112140

Department of Computer Science and Engineering
National Institute of Technology Silchar, Assam

Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Overview

Sign languages serve as a vital mode of communication for individuals with hearing and speech impairments, allowing them to convey thoughts and emotions through hand gestures, facial expressions, and body movements. Among various sign languages, the Indian Sign Language (ISL) is widely used by the deaf and hard of hearing communities in India. However, despite its importance, ISL remains under-researched and lacks widespread digital support, making communication challenging for the deaf community in India [7].

This work proposes a recognition system using Convolutional Vision Transformers (CvT) [18], which leverage both global contextual modeling and local feature extraction. Our framework uses MediaPipe HandLandmarker to detect hand landmarks and a pre-trained CvT-13 model for robust gesture classification. By combining edge enhancement and skeleton-based landmark representation, we improve accuracy while maintaining computational efficiency.

1.2 Importance

Indian Sign Language is not just a linguistic medium; it is a cultural and communicative lifeline for millions of Indians with hearing or speech disabilities. Unlike more extensively studied sign languages such as American Sign Language (ASL) [13, 11], ISL faces significant challenges that increase its complexity from a computational and linguistic perspective.

ISL lacks standardization and often varies significantly across different regions of India. This results in a rich but inconsistent set of gestures, many of which are influenced by local dialects, contexts, and cultural practices. Ges-

tures for the same word can differ across regions, and the same gesture may convey different meanings based on subtle variations in speed, orientation, or accompanying facial expressions.

These intricacies make ISL recognition a more complex task than many other sign languages. Furthermore, the scarcity of large, labeled datasets for ISL further compounds these difficulties, hindering the development of accurate, generalizable recognition systems [7]. Bridging this gap through research not only pushes the boundaries of computer vision but also fosters inclusivity and digital accessibility for India's deaf and hard-of-hearing communities.

Despite its social importance, ISL has not received proportional attention in the domain of digital and assistive technologies. Enhancing ISL recognition capabilities is therefore not only a technological pursuit but also a step toward social inclusion, enabling better access to education, employment, and public resources for the deaf community in India.

1.3 Motivation

Existing ISL recognition systems struggle with:

- Variations in hand shapes, motions, lighting, and backgrounds.
- Limited ISL datasets and class imbalance.
- High computational cost vs. real-time performance trade-offs.

We explore CvT's hybrid CNN–Transformer architecture to capture both local spatial features and global context, aiming to bridge communication gaps for the deaf community.

1.4 Problem Statement

Automated ISL recognition remains inefficient and inaccurate, limiting seamless interaction between deaf and hearing users. Key challenges include:

- Complex spatial–temporal nature of gestures.
- Diverse signing styles across individuals.
- Scalability for real-world deployment.

This project addresses these by fine-tuning a pre-trained CvT model with advanced preprocessing and landmark-based features.

1.5 Objectives of the Project

- Develop an effective and precise ISL recognition system using CvT.
- Leverage CvT's capacity to capture spatial and hierarchical features.
- Improve understanding of hybrid models' strengths and limitations in SLR.
- Contribute theoretical insights to guide future ISL and assistive-tech research.

1.6 Preliminaries

1.6.1 System Configuration and Training Environment

All experiments were conducted on a workstation with the following specifications:

- **Processor:** Intel Xeon CPU @ 2.20 GHz (4 cores)
- **RAM:** 29 GB
- **GPU:** NVIDIA T4 $\times$ 2
- **Operating System:** Windows 11
- **Programming Language:** Python 3.8
- **Libraries and Frameworks:** OpenCV, PyTorch Lightning, Scikit-Learn, MediaPipe HandLandmarker

Model training and evaluation were performed in an Anaconda Jupyter Notebook environment, with random seeds fixed to ensure reproducibility.

1.6.2 Python and Key Libraries

Our pipeline is implemented in Python 3.8, leveraging:

- **PyTorch Lightning** for structured model training and callbacks.
- **OpenCV** for image I/O, preprocessing (resizing, sharpening), and geometric transforms [4].
- **MediaPipe HandLandmarker** for real-time hand landmark extraction [10].
- **scikit-learn** for evaluation metrics and auxiliary utilities.

- **Tkinter** for lightweight GUI components in data collection demos.

1.6.3 Image Handling with OpenCV

We support both JPEG (lossy) and PNG (lossless) formats. OpenCV handles:

- Reading and writing images.
- Grayscale conversion and color-space shifts.
- Sharpening via unsharp mask to enhance hand contours.
- Geometric transforms (rotations, perspective warps).

1.6.4 Hand Landmark Detection

MediaPipe's HandLandmarker provides 21 3D keypoints per hand in real time [10]. Overlaying these skeleton coordinates on frames helps our CvT model focus on motion patterns rather than raw pixel intensities.

1.6.5 Convolutional Vision Transformers (CvT)

CvT integrates convolutional token embeddings with transformer self-attention:

- *Conv Token Embedding*: replaces rigid patches with overlapping convolutions for better locality.
- *Convolutional Projection Attention*: uses depth-wise separable convolutions before self-attention for efficiency.
- *CvT-13*: chosen for its balance of accuracy and inference speed.

1.6.6 Evaluation Metrics

To account for class imbalance, we report:

- **Accuracy**: overall correct predictions.
- **Weighted Precision, Recall, F1-score**: average per-class metrics weighted by class frequency.

1.7 Advantages of Convolutional Vision Transformers (CvT)

The Convolutional Vision Transformers (CvT) combine the strengths of Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs). Below are the key advantages of CvT based on the paper:

- **Locality and Spatial Hierarchy:** Combines convolutional token embeddings and hierarchical transformers to capture local features and spatial context.
- **Robust Invariance:** Introduces shift, scale, and distortion invariance from CNNs for robust vision tasks.
- **Efficient Computation:** Uses depth-wise separable and strided convolutions to reduce FLOPs and memory usage in attention mechanisms.
- **Global Context:** Maintains Transformers' dynamic attention for modeling global relationships.
- **No Positional Encodings Needed:** Simplifies architecture and supports variable input resolutions.
- **High Performance:** Achieves state-of-the-art results on ImageNet and downstream tasks with fewer parameters and FLOPs.
- **Scalability:** Efficient multi-stage design supports larger datasets and higher resolutions.
- **Simplified and Robust Design:** Adapts well to diverse vision tasks while simplifying implementation.

Chapter 2
Literature Review

2.1 Introduction

Sign language recognition (SLR) enables better communication for the hearing and speech impaired. While CNNs excel in static gestures and Transformers in global context modeling, both have limitations in handling complex gestures. Convolutional Vision Transformers (CvTs) combine local and global feature extraction, offering a robust solution. This survey explores CvTs' potential in advancing Indian Sign Language (ISL) recognition.

2.2 Overview of Sign Language Recognition Methods

2.2.1 CNN-Based Approach

Several studies have leveraged Convolutional Neural Networks (CNNs) for sign language recognition:

- **SLRNet-8 (CNN-Based Architecture):** This model, developed for American Sign Language (ASL) recognition, utilized preprocessing techniques such as grayscale conversion and resizing, followed by CNN training with 6 convolutional layers and data augmentation. The approach achieved **99.9% accuracy** on four ASL datasets [13].
- **SURF and CNN Hybrid Model:** The integration of Speeded Up Robust Features (SURF) for feature extraction with CNN for classification demonstrated over **99% accuracy** for Indian Sign Language (ISL) recognition [7].

- **Real-time CNN Recognition:** A framework implementing CNNs for real-time sign language gesture recognition from video sequences achieved **92.88% accuracy** with significant improvements in processing speed suitable for practical applications [11].

2.2.2 Hybrid CNN-HMM Approach

The combination of CNN for feature extraction and Hidden Markov Models (HMMs) for sequence modeling has proven effective for continuous sign language recognition:

- **Deep Sign Framework:** A hybrid CNN-HMM approach, pretrained on ImageNet and fine-tuned on sign language datasets, significantly reduced Word Error Rate (WER) across multiple datasets. GoogLeNet, a CNN architecture used in this framework, outperformed others, achieving a WER of **30.0%** compared to previous state-of-the-art results of 38.3% on the RWTH-PHOENIX-Weather corpus [8].

2.2.3 CNN-RNN-Based Approach

Combining CNNs for feature extraction with Recurrent Neural Networks (RNNs) for temporal modeling has shown strong performance:

- **CNN and RNN for Temporal-Spatial Features:** A model developed for sign language recognition analyzed both temporal and spatial features using a combination of CNN and RNN architectures. This approach effectively captured both the spatial characteristics of individual frames and the temporal relationships between sequences of frames [11].

- **Pose and Multimodal Transformers:** The integration of OpenPose for human keypoint estimation with CNNs and RNNs resulted in a test accuracy of **74.7%** for isolated Flemish Sign Language recognition [3].

2.2.4 Transformer-Based Approach

Transformer architectures have gained prominence for their efficiency in modeling spatial-temporal relationships:

- **Sign Language Transformers:** A unified transformer-based model for Continuous Sign Language Recognition (CSLR) and Translation (SLT) achieved state-of-the-art performance, doubling BLEU scores

in some translation tasks on the RWTH-PHOENIX-Weather-2014T dataset [2]. This approach addressed both recognition and translation in a joint end-to-end framework.

- **SIGNFORMER:** Designed for static Indian sign language recognition, this Vision Transformer model achieved **99.29% accuracy** within just five training epochs, demonstrating the efficiency of attention mechanisms for capturing spatial relationships in sign gestures [9].
- **Korean Sign Language Recognition:** A transformer-based deep neural network approach for Korean Sign Language achieved **98.1% accuracy**, outperforming traditional CNN approaches by leveraging the transformer's ability to capture global dependencies [14].
- **Arabic Sign Language Recognition:** Vision Transformers with transfer learning approaches demonstrated excellent performance on Arabic Sign Language, with a fine-tuned ViT-Base model achieving **97.84% accuracy** [1].

2.3 Key papers Summary

SL.NO.	Paper	Methodology	Findings and Takeaways
1	Vision Transformers and Transfer Learning Approaches for Arabic Sign Language Recognition. [1]	The paper investigates two primary approaches for Arabic Sign Language (ArSL) recognition: • Transfer learning using pretrained deep learning models (e.g., VGG, ResNet, MobileNet, Xception, Inception, DenseNet, BiT, ViT, Swin). • Deep learning approaches using CNN architectures developed from scratch. The study evaluates these models on a dataset containing 54,049 images of 32 Arabic alphabets.	The study highlights the effectiveness of transfer learning combined with vision transformers for Arabic Sign Language recognition, demonstrating superior performance over CNN-based models. Transfer learning approaches, particularly those utilizing pretrained models, are more robust and efficient for sign language classification tasks.

2	A New Benchmark on American Sign Language Recognition using CNN [13]	The methodology consists of: • **Preprocessing:** Grayscale, normalization, resized to 64×64. • **Model:** SLRNet-8 (6 conv, 3 pool layers), ReLU, batch norm, max-pooling (2×2), GAP, dropout (0.5). • **Training:** Augmentation (rotation, zoom, shift), Adam (lr=0.001), 10-fold CV, early stopping (30 epochs). • **Testing:** 99.9% accuracy on 4 ASL datasets.	Key findings and takeaways: • **Accuracy:** SLRNet-8 reached 99.9%. • **Preprocessing:** Efficient grayscale/resizing aided fast training. • **Generalization:** Use of dropout, ReLU, and pooling layers helped minimize overfitting and improved model adaptability. • **Augmentation:** Boosted robustness to input variations.

3	Indian Sign Language Recognition System using SURF with SVM and CNN. [7]	The proposed ISL recognition system includes: • **Feature Extraction:** SURF (Speeded Up Robust Features). • **Mapping Features:** Bag of Visual Words (BOVW) model. • **Classification:** SVM and CNN. • **Custom Dataset:** 36,000 images created from videos captured via webcam. Outputs labels in text and speech formats.	Key findings and takeaways: • **Hybrid Approach:** Combining SURF with SVM and CNN offers a robust and efficient ISL recognition method. • **High Accuracy:** The system achieves over 99 • **Custom Dataset:** Developed a custom dataset due to the lack of a standardized ISL dataset.

4	Attention is All You Sign: Sign Language Translation with Transformers. [19]	The methodology includes: • Transformer-based approach combined with the Spatial-Temporal Multi-Cue (STMC) network for end-to-end Sign Language Translation (SLT). • Explores Gloss2Text and Sign2Gloss2Text tasks. • Implements weight tying, transfer learning, and ensemble learning. • Experiments conducted on PHOENIX-Weather 2014T and ASLG-PC12 datasets.	Key findings and takeaways: • **State-of-the-Art Results:** Achieved BLEU score improvements of +5 on ground truth glosses and +7 on predicted glosses for the PHOENIX dataset. • **End-to-End Translation:** Using predicted glosses outperforms ground truth glosses. • **Novel Architecture:** Introduced STMC-Transformer, with potential for further improvements via joint training of SLR and translation systems.

2.4 Addressable Research Gaps

2.4.1 Improved Scalability

- **Gap:** In CNN models, performance saturates beyond a certain model size
- **Solution:** CvTs benefit from the scalability of Transformers, enabling the handling of extremely large models and datasets (e.g., JFT-300M).

2.4.2 Local Spatial Structures and Hierarchical Feature Extraction

- **Gap:** ViTs struggle to model local spatial structures due to global self-attention.
- **Solution:** CvTs use convolutional token embedding and projections for local context and hierarchical feature extraction.
- **Gap:** ViTs underperform on small datasets due to lack of inductive biases.
- **Solution:** CvTs incorporate CNN-like biases (locality and hierarchy), achieving competitive results with less data.

2.4.3 Positional Encoding Inefficiency

- **Gap:** ViTs rely on positional encodings, which are inefficient for variable resolutions.
- **Solution:** CvTs embed spatial relationships directly via convolutions, eliminating the need for positional encodings.

Chapter 3
Dataset Preparation

3.1 Overview of the Dataset

To develop a comprehensive recognition system, we curated a dataset comprising 150 distinct word classes. Each class represents a unique ISL gesture and was carefully selected to cover a wide range of commonly used signs in daily communication. The selection of these 150 classes was based on gestures referenced from the official Indian Sign Language portal indiansignlanguage.org, ensuring relevance and practicality in real-world usage.

The dataset encompasses a diverse range of hand gestures, accounting for variations in hand shapes, orientations, lighting conditions, and gesture articulations. This diversity contributes to the robustness and generalizability of our recognition system.

3.2 Data Collection

The dataset was constructed using a combination of sources:

- **Reference Material:** The primary reference for ISL gestures was the Indian Sign Language portal (`https://indiansignlanguage.org/`), which provides visual representations of various signs.
- **Webcam Captures:** To augment the data set and introduce real-world variability, additional samples were captured using a laptop webcam (MSI GF65 Thin). This approach ensured the inclusion of diverse backgrounds, lighting conditions, and hand orientations.

Each word class contains approximately 30–50 images, resulting in a balanced and comprehensive data set suitable for training and evaluation purposes.

3.3 Dataset Tables

Table I: File Count for Letters and Digits

Label	Count	Label	Count	Label	Count
1	220	D	220	P	220
2	225	E	240	Q	225
3	220	F	210	R	220
4	215	G	235	S	230
5	210	H	225	T	240
6	240	I	220	U	220
7	230	J	215	V	220
8	200	K	215	W	225
9	200	L	240	X	220
A	210	M	210	Y	230
B	230	N	225	Z	220
C	225	O	230		

Table 3.1: File Count for ISL Letters and Digits

This table includes the number of image/video files available for ISL hand signs representing digits (1–9) and letters (A–Z).

Table II: File Count for Word-Level Signs

Word	Count	Word	Count	Word	Count	Word	Count
afraid	220	pray	220	pain	200	about	210
agree	225	secondary	230	up	220	always	205
assistance	210	skin	215	present	200	again	215
bad	200	small	205	warn	200	better	225
become	220	specific	210	spouse	205	bring	210
college	200	stand	240	wear	210	between	220
doctor	200	teach	230	welcome	215	before	200
from	200	there	240	wish	225	beyond	205
ghost	215	group	220	with	220	behind	210
mother	200	today	200	without	230	beside	200
old	200	toward	220	work	220	across	215
you	200	above	210	arrive	205	adapt	200
again	210	afford	205	allow	215	borrow	200
catch	220	choose	210	doubt	205	enjoy	200
enter	200	excuse	205	follow	210	guide	200
happen	215	include	210	laugh	200	manage	205
offer	200	measure	215	notice	200	occupy	210
participate	220	permit	205	prefer	200	recall	205
reduce	210	remove	200	reply	205	remain	200
share	215	steal	200	suppose	205	survive	210
treat	200	unite	205	upload	200	chase	205
divide	200	dream	210	escape	205	invent	200
isolate	210	justify	205	locate	200	oppose	205
repair	200	revive	210	salute	205	survive	210
travel	200	erode	205	evolve	210	rejoice	200
venture	205	wonder	200	witness	205	yield	200

Table 3.2: File Count for ISL Word-Level Signs

This table lists the file counts for commonly used word-level signs in Indian Sign Language.

3.4 Dataset Structure and Format

The dataset is organized into two primary subsets:

- **Training Set (80%):** Utilized for model training and feature extraction.

- **Testing Set (20%):** Reserved to evaluate the performance of the model and generalizability.

All images are standardized to a resolution of 224×224 pixels and stored in JPEG format to maintain consistency and facilitate efficient processing.

3.5 Preprocessing Pipeline

To enhance the quality and robustness of the dataset, a two-stage preprocessing pipeline was implemented:

3.5.1 Basic Preprocessing

Initial preprocessing steps included:

- **Resizing:** All images were resized to 224×224 pixels to match the input requirements of the recognition model.
- **Normalization:** Pixel values were normalized to a standard range to facilitate faster convergence during training.

3.5.2 Data Augmentation

To improve the generalization and robustness of the model, various enhancement techniques were applied:

- **Color Transformations:**
  - Random adjustments in brightness and contrast.
  - Variations in hue, saturation, and value.
  - Application of Contrast Limited Adaptive Histogram Equalization (CLAHE).
  - Occasional conversion to grayscale.
  - RGB channel shifting and random channel switching.
  - Leaving a random channel to simulate sensor anomalies.
- **Noise and Blur Effects:**
  - Addition of Gaussian noise.
  - Application of Gaussian and motion blur.
  - Simulation of ISO noise characteristics.
  - Random removal of image pixels.

- **Geometric Transformations:**
  - Random shifts, scaling, and rotations.
  - Perspective transformations.
- **Lens Distortion Simulation:**
  - Optical distortion effects to mimic lens anomalies.

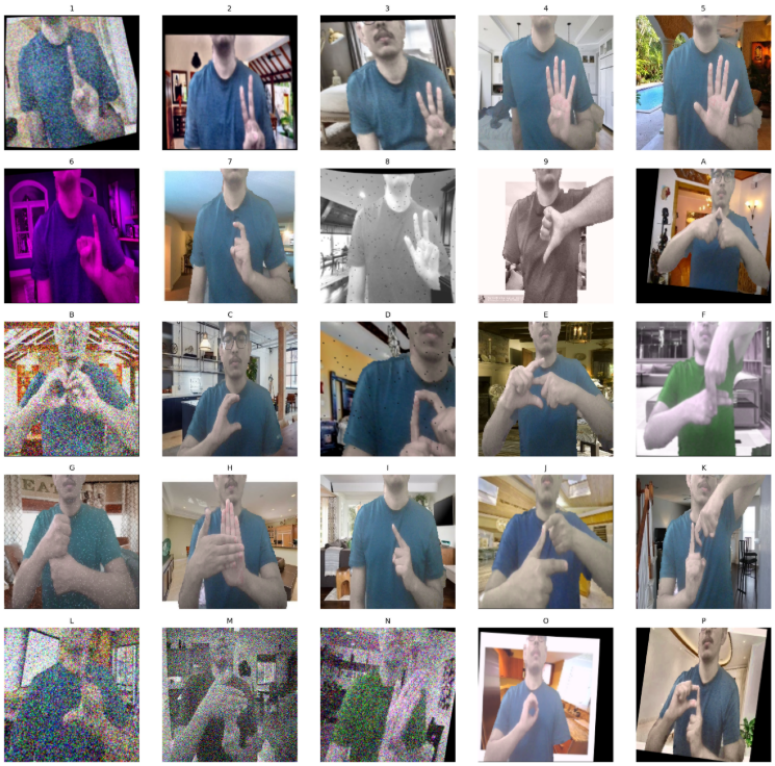

Figure 3.1: Grid of random ISL hand gesture images

Chapter 4
Methodology

4.1 Overview

This chapter elaborates on the internal design of the Convolutional Vision Transformer (CvT) used in our ISL recognition model, the training strategy adopted, and the evaluation metrics applied. Components like convolutional embeddings, attention mechanisms, and transformer stages are discussed in detail to demonstrate how CvT was leveraged for gesture classification.

4.2 CvT Architecture

CvT integrates convolutional operations with Vision Transformer (ViT) blocks to combine the locality strengths of convolutional neural networks (CNNs) with the global context modeling capabilities of transformers. The CvT-13 variant used in our model comprises the following components:

I. **Convolutional Token Embedding:**

- Instead of the standard ViT's non-overlapping fixed-size patch embedding, CvT uses overlapping convolutional tokenization with multiple convolutional layers.
- This embedding uses a stack of convolutional layers with kernel sizes of 3×3 and stride less than the kernel size (typically stride 1 or 2) to generate tokens with richer local features and spatial continuity.
- The convolutional embedding reduces the input image to a lower spatial resolution feature map of shape $H' \times W' \times C$, where H' and W' are smaller than the original image height and width due to downsampling, and C is the token dimension.

- This convolutional embedding acts as a learned, flexible feature extractor that retains spatial relationships better than rigid patch splitting.

II. **Hierarchical Transformer Blocks:**

- Each transformer block begins with a depth-wise separable convolution (DWConv) layer applied on the token embeddings to inject inductive biases of locality and spatial correlation.
- The DWConv operates with a kernel size of 3×3, padding 1, maintaining spatial resolution, but significantly enhancing local context modeling before the attention mechanism.
- Following the DWConv, multi-head self-attention (MHSA) is applied, with typical configurations of 8 or 12 attention heads and a token embedding dimension d (e.g., 384 or 512).
- MHSA uses scaled dot-product attention, computing:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V,$$

  where Q, K, V are the query, key, and value matrices, and d_k is the dimension per head.
- Layer normalization (LayerNorm) is applied before MHSA and the feed-forward network (FFN) for stable training.
- The FFN consists of a two-layer multi-layer perceptron (MLP) with an intermediate hidden dimension usually 4 times larger than the token embedding size, activated by the GELU function:

$$\text{MLP}(x) = \text{GELU}(xW_1 + b_1)W_2 + b_2,$$

  with residual skip connections around both MHSA and MLP blocks for improved gradient flow.

III. **Multi-Stage Feature Extraction:**

- The model is organized into multiple stages (e.g., 3 stages for CvT-13), each stage operating at different spatial resolutions.
- At the end of each stage, the spatial resolution is reduced by convolutional layers with stride 2, while the token embedding dimension is increased, allowing hierarchical feature extraction.

- This progressive reduction in spatial size and increase in channel dimension allows CvT to capture both fine-grained local features in early stages and more abstract, global representations in later stages.

IV. **Classification Head:**

- After the last transformer stage, a Global Average Pooling (GAP) layer aggregates spatial features into a single vector representation.
- This vector is passed through a fully connected MLP layer that maps it to the final class logits.
- Dropout may be applied before the classification layer to regularize the model.

Additional Notes:

- The convolutional token embedding and depth-wise convolution layers inject strong inductive biases of locality and translation equivariance into the model, improving data efficiency and performance on vision tasks.
- Compared to vanilla ViT, CvT achieves better accuracy with fewer parameters and less computational overhead due to the efficient convolutional operations combined with transformers.
- The hierarchical design allows CvT to handle higher-resolution inputs more effectively than traditional ViTs.

Figure 4.1: Architecture of the Convolutional Vision Transformer (CvT-13)

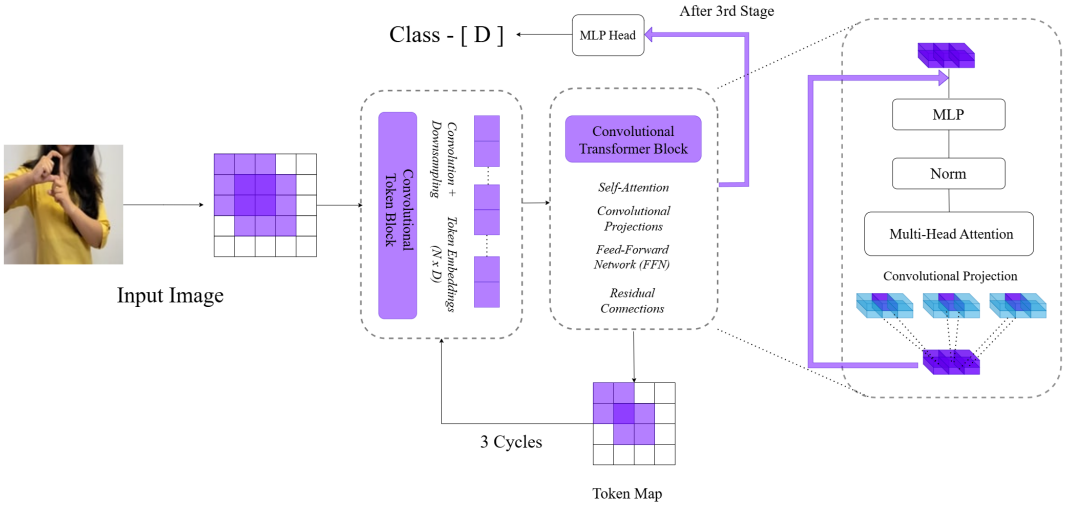

Figure 4.2: Architecture of the Convolutional Vision Transformer (CvT-13)

4.3 Landmark Representation with MediaPipe

In addition to RGB input, MediaPipe HandLandmarker was used to generate 21 hand keypoints per frame. These keypoints were rendered as skeletonized representations using OpenCV to guide the CvT model in capturing structural gesture details.

To enhance landmark detection, we applied sharpening without a mask using an unsharp filter, which helped highlight the hand edges and contours more effectively. This preprocessing step improves the model's ability to localize critical points in the hand.

Figure 4.3: Sample image before (left) and after (right) sharpening

After sharpening, MediaPipe extracted the hand landmarks and generated a corresponding skeleton image. This processed output was used as auxiliary input, enabling the CvT model to focus more on shape and articulation rather than raw pixels.

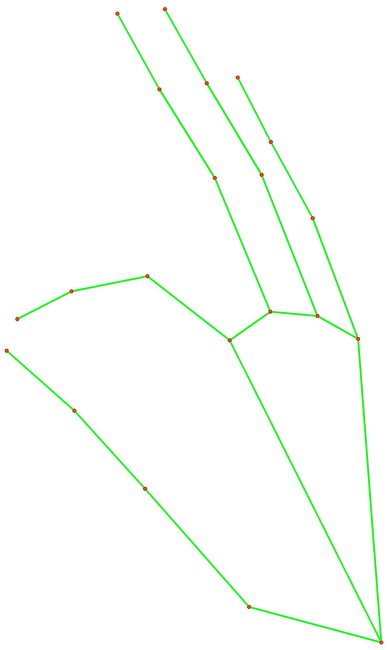

Figure 4.4: Sample image after landmark transformation using MediaPipe

4.4 Training Strategy

To train the ISL recognition model effectively, we adopted a careful fine-tuning strategy using the pre-trained `microsoft/cvt-13` model from the Hugging Face Model Hub. Initially, the lower (early) layers of the CvT-13 model were frozen to focus learning on the new classification head, allowing the model to adapt high-level features without disrupting the pre-trained representations.

After the first 5 epochs, we progressively unfroze the base model's layers using a custom PyTorch Lightning callback. This staged unfreezing technique helped the model gradually adjust to the new dataset and stabilize training, reducing the risk of catastrophic forgetting.

The dataset was split into 70% training, 10% validation, and 20% testing. We trained the model for 100 epochs with a batch size of 64. Optimization was performed using the AdamW optimizer, initialized with a learning rate of 1×10^{-6}. To ensure steady convergence, a ReduceLROnPlateau scheduler was used, which reduced the learning rate by a factor of 0.1 when the validation loss plateaued, down to a minimum of 1×10^{-9}. The loss function employed was binary cross-entropy, which worked well in this multiclass setting due to its stability and smooth gradients.

Summary of Training Parameters:

- Pre-trained model: `microsoft/cvt-13`
- Staged unfreezing after 5 epochs
- Train/Validation/Test split: 70% / 10% / 20%
- Optimizer: AdamW with ReduceLROnPlateau scheduler
- Initial Learning Rate: 1×10^{-6}; Min LR: 1×10^{-9}
- Batch size: 64; Total epochs: 100
- Loss function: Binary Cross-Entropy

4.5 Evaluation Metrics

To ensure reliable and fair performance comparison, we use:

- **Accuracy (Multiclass)**:
  Measures the proportion of correctly predicted instances among all predictions.

$$\text{Accuracy} = \frac{\sum_{c=1}^{C} TP_c}{\sum_{c=1}^{C} (TP_c + FP_c + FN_c)} \tag{1}$$

  Where:

  TP_c: True Positives for class c (correct predictions of class c)

  FP_c: False Positives for class c (instances incorrectly predicted as class c)

  FN_c: False Negatives for class c (instances of class c incorrectly predicted as another class)

  C: Total number of classes

- **Weighted Precision**:
  Calculates the average precision across all classes, weighted by the number of true instances (support) for each class.

$$\text{Weighted Precision} = \frac{\sum_{c=1}^{C} N_c \cdot \text{Precision}_c}{\sum_{c=1}^{C} N_c} \tag{2}$$

  Where:

  N_c: Number of true instances (support) for class c

  Precision_c: Precision for class c, defined as:

$$\text{Precision}_c = \frac{TP_c}{TP_c + FP_c} \tag{2a}$$

- **Weighted Recall**:
  Computes the average recall across all classes, weighted by the number of true instances for each class.

$$\text{Weighted Recall} = \frac{\sum_{c=1}^{C} N_c \cdot \text{Recall}_c}{\sum_{c=1}^{C} N_c} \tag{3}$$

  Where:

Recall_c: Recall for class c, defined as:

$$\text{Recall}_c = \frac{TP_c}{TP_c + FN_c} \tag{3a}$$

- **Weighted F1 Score**:
  The average of the F1 scores for each class, weighted by the number of true instances for each class.

$$\text{Weighted F1 Score} = \frac{\sum_{c=1}^{C} N_c \cdot F1_c}{\sum_{c=1}^{C} N_c} \tag{4}$$

Where:

$F1_c$: F1 Score for class c, defined as:

$$F1_c = 2 \cdot \frac{\text{Precision}_c \cdot \text{Recall}_c}{\text{Precision}_c + \text{Recall}_c} \tag{4a}$$

4.6 Description of SOTA Models

1. **ResNet (Residual Network)** [6] introduces skip connections to facilitate the training of very deep networks by addressing vanishing gradients. ResNet-50, a common variant, has 50 layers and approximately 25.6 million parameters. It employs bottleneck residual blocks and is typically trained on 224×224 input images.[6]

2. **Vision Transformer (ViT)**[5] applies the Transformer architecture to vision tasks by splitting an image into 16×16 patches, flattening them, and feeding them as tokens into a standard Transformer encoder. ViT-Base has 86 million parameters, 12 attention heads, and 12 layers, with input resolution 224×224.

3. **InceptionV3** [16] enhances the inception module with factorized convolutions, label smoothing, and auxiliary classifiers. It contains approximately 23.8 million parameters and accepts 299×299 input images. The architecture uses parallel convolutional paths and dimensionality reduction to optimize computational efficiency.

4. **VGG16** [15] is a 16-layer convolutional network with about 138 million parameters. It consists of sequential 3×3 convolutions and 2×2 max-pooling layers, followed by three fully connected layers. It uses fixed-size 224×224 input images and is known for its depth and simplicity.

Chapter 5
Results and Discussion

5.1 Overview

This chapter presents the evaluation of the Convolutional Vision Transformer (CvT) model for Indian Sign Language (ISL) recognition. The CvT model's performance is compared against state-of-the-art deep learning models such as ResNet, ViT, InceptionV3, and VGG16. Metrics used include Accuracy, Precision, Recall, and F1-Score.

5.2 Comparison with State-of-the-Art Models

Table 5.1: Performance Comparison of CvT and Other SOTA Models

Metric	**ResNet**	**ViT**	**InceptionV3**	**VGG16**	**CvT (Ours)**
Accuracy	81.67%	90.94%	84.90%	83.13%	**91.66%**
Precision	81.92%	91.85%	85.62%	83.44%	**92.17%**
Recall	81.33%	90.00%	84.10%	82.91%	**91.14%**
F1-Score	80.50%	89.45%	83.85%	81.68%	**90.92%**

5.3 Ablation Study

An ablation study was conducted to evaluate the impact of advanced preprocessing steps, including sharpening and landmark-based skeletonization. The second phase of the dataset included additional word-level signs (e.g., "about", "today", "before"). The following results compare model performance using raw images versus processed images.

Table 5.2: Ablation Study: Raw vs Processed Dataset Performance

Metric	Base Dataset (Raw Images)	Processed Dataset (Landmark + Filter)
Accuracy	86.00%	**91.05%**
Precision	85.87%	**90.98%**
Recall	86.21%	**91.13%**
F1-Score	85.54%	**90.91%**

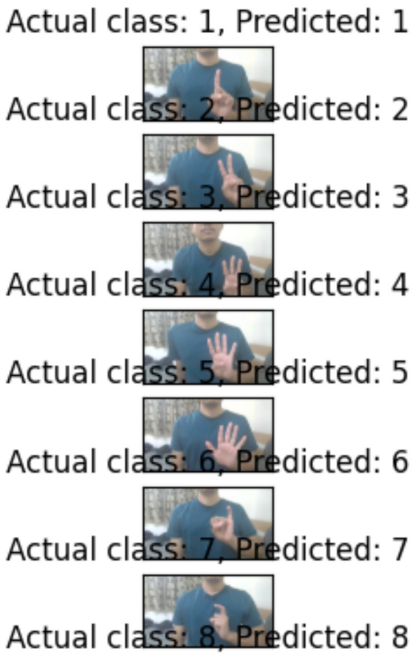

Figure 5.1: Sample Prediction Output on ISL Numbers using CvT

5.4 Training and Validation Performance

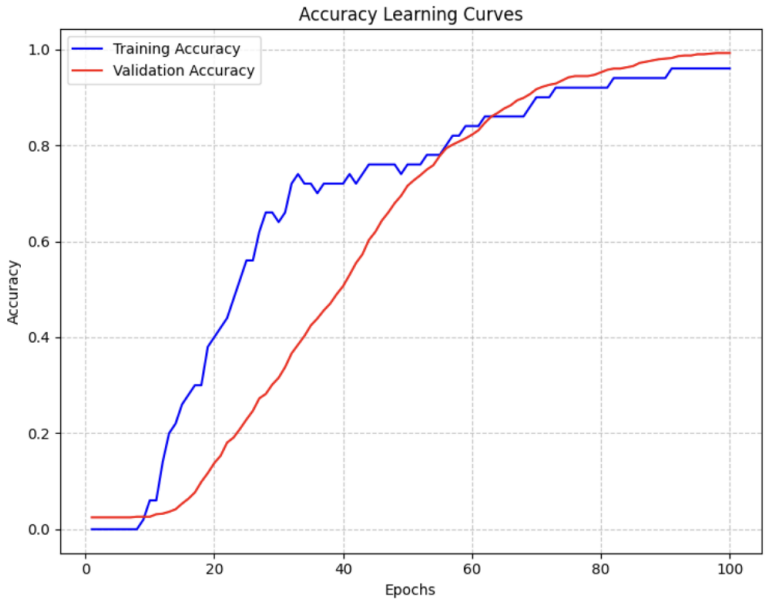

Figure 5.2: Training vs Validation Accuracy Graph

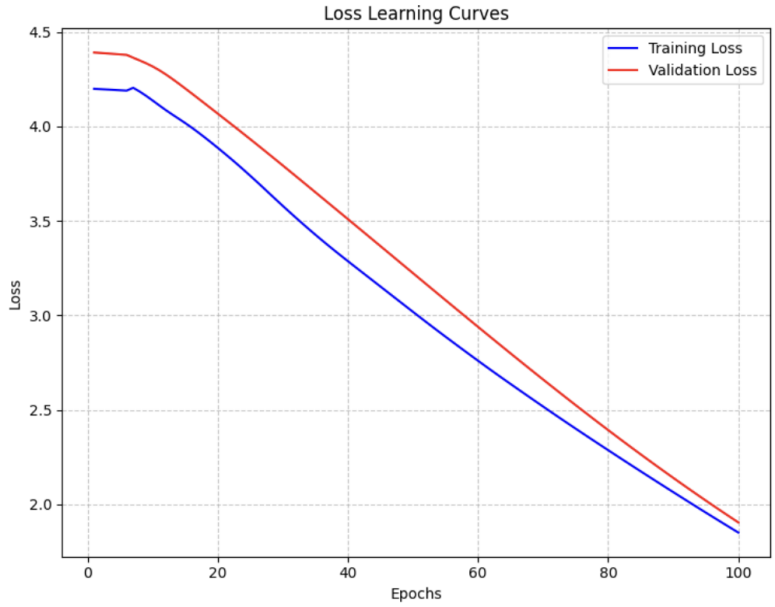

Figure 5.3: Training vs Validation Loss Graph

5.5 Analysis: CvT Performance and Model Limitations

5.5.1 Why CvT Outperforms Other Models

The experimental results demonstrate that the Convolutional Vision Transformer (CvT) model achieved superior performance (91.66% accuracy) compared to other state-of-the-art approaches: ResNet (81.67%), Vision Transformer (90.94%), InceptionV3 (84.90%), and VGG16 (83.13%). This performance advantage can be attributed to several key factors:

1. **Hybrid Architecture Integration**: CvT effectively combines the local feature extraction capabilities of CNNs with the global context modeling of Transformers, creating a synergistic architecture that captures both detailed hand features and their contextual relationships.

2. **Hierarchical Feature Learning**: The multi-stage design enables progressive feature extraction where:

   - Early stages capture local, fine-grained details
   - Later stages aggregate these into higher-level semantic representations
   - Spatial dimension reduction with increasing channel depth optimizes information flow

3. **Overlapping Convolutional Tokenization**: Unlike standard ViT's rigid patch splitting, CvT employs overlapping convolutions with stride less than kernel size, resulting in tokens with enhanced local features and spatial continuity.

4. **Architectural Efficiency**: CvT achieves its performance with several efficiency advantages:

   - No positional encodings required
   - Support for variable input resolutions
   - Fewer parameters and computational requirements

5.5.2 Limitations of Other Models

Model	Key Limitations
ResNet	• Limited global context awareness despite deep architecture • Fixed hierarchical structure with predetermined residual connections • Information bottleneck from excessive downsampling operations
Vision Transformer	• Struggles with local spatial structures due to global self-attention • Inefficient positional encoding mechanism for variable resolutions • Lacks multi-scale feature hierarchy for detailed hand gesture recognition
InceptionV3	• Fixed receptive fields despite multi-scale convolutions • Limited capacity to model long-range dependencies • Predefined parallel paths lack adaptability to focus on key regions
VGG16	• Simplistic stacked architecture without sophisticated mechanisms • Insufficient feature hierarchy for complex gesture representation • Computationally inefficient parameter count with diminishing returns

Table 5.3: Limitations of State-of-the-Art Models Compared to CvT

The empirical results demonstrate that CvT successfully addresses the limitations of both pure CNN architectures and standard Transformer models by effectively combining their strengths while mitigating their individual weaknesses in the context of Indian Sign Language recognition.

Chapter 6
Application Deployment and Inference Interface

6.1 Overview

To facilitate real-time interaction and usability, a graphical user interface (GUI)-based desktop application was developed using `Tkinter`. This application integrates the trained CvT model with live webcam feed to recognize ISL gestures in real-time. The system also provides auxiliary features such as landmark rendering, speech synthesis, gesture stability detection, and semantic word combinations for improved communication flow.

6.2 System Architecture

The overall system is divided into five functional blocks:

I. **Live Video Capture:** Captures webcam frames at runtime.

II. **Hand Detection Module:** Uses MediaPipe for robust landmark extraction or a fallback Haar cascade and skin color segmentation if MediaPipe fails.

III. **Gesture Prediction Pipeline:** Processes the region of interest and predicts the gesture using a TensorRT-optimized CvT model.

IV. **GUI & Text Aggregator:** Displays live feed, prediction status, captioning, and accumulated text in a user-friendly interface.

V. **Speech and Semantics Engine:** Converts recognized words to speech and combines word pairs into compound expressions when applicable.

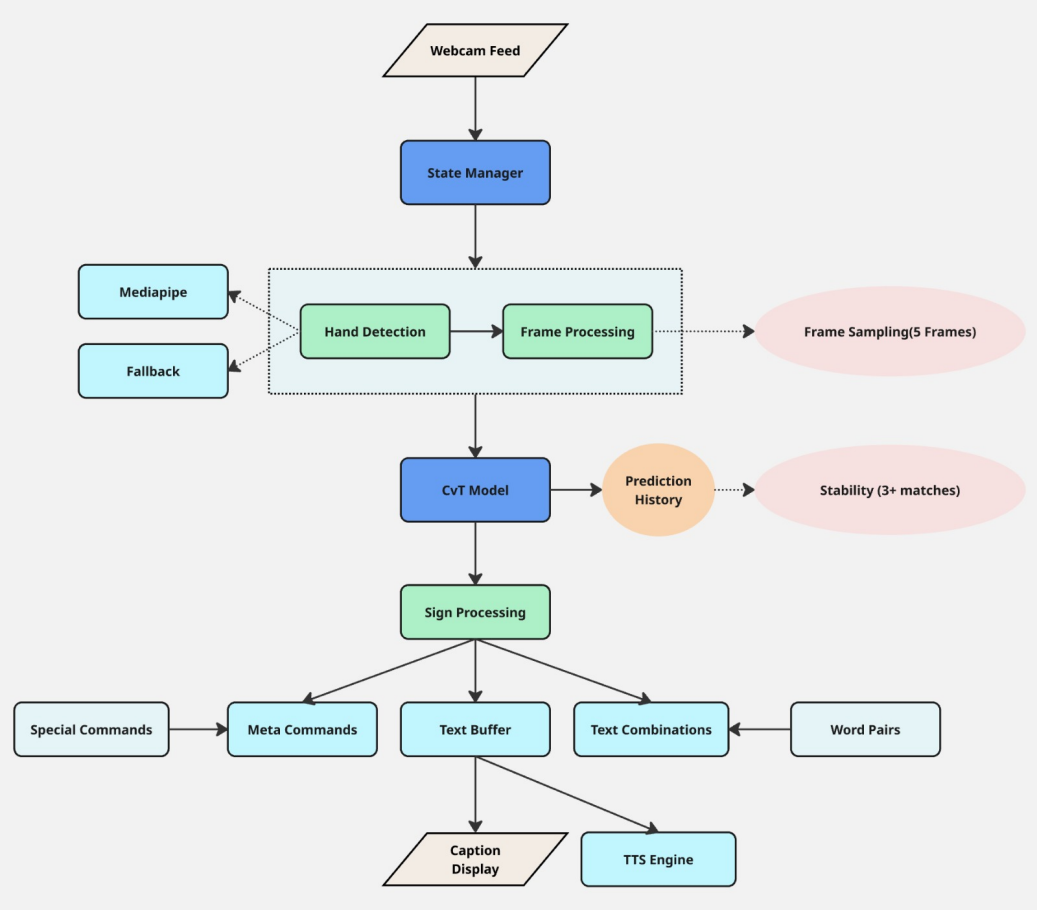

Figure 6.1: Architecture Diagram of the ISL Inference Application

6.3 Application Workflow: Algorithm

The high-level logic of the application is summarized below:

Algorithm 1 ISL Gesture Recognition Application

```
Initialize webcam and GUI elements
Load CvT model and convert with TensorRT (if available)
Initialize hand detector (prefer MediaPipe, fallback to color segmentation)
while application is running do
    Capture frame from webcam
    Detect hand region in the frame
    if hand is detected then
        Preprocess frame
        Predict gesture using CvT model
        Append to prediction history
        if prediction is stable over several frames then
            Update caption display
            Add prediction to text area
            Check for known word pair combinations
            if combination exists then
                Prompt user to replace with compound word
            end if
        end if
    else
        Clear prediction buffer
    end if
    Update GUI elements (FPS, status, etc.)
end while
On exit: release webcam and close hand detector
```

6.4 Gesture Stability and Prediction Smoothing

To minimize flickering and erratic predictions, a temporal buffer stores the last 10 predictions. A new gesture is accepted only if it occurs at least three times within the buffer, ensuring robustness against transient misclassifications. This mechanism greatly improves the user experience in dynamic environments.

6.5 Word Combination and Speech Integration

The application supports intelligent word combinations such as:

- `old + mother` → `grandmother`
- `child + doctor` → `pediatrician`
- `work + college` → `internship`

These combinations are prompted interactively to the user through dialog boxes. A speech engine (`pyttsx3`) is also integrated, allowing the user to convert accumulated text into spoken sentences using a trigger gesture (e.g., `secondary`).

6.6 Deployment Optimizations

The inference pipeline is optimized using TensorRT (via `torch_tensorrt`) for faster execution on compatible hardware. The model operates in FP16 precision and `channels_last` memory format to maximize throughput during prediction.

Key Features:

- Real-time webcam-based gesture recognition.
- Robust hand detection using MediaPipe with fallback.
- GUI with captioning, text aggregation, and word combination support.
- Text-to-speech using `pyttsx3`.
- TensorRT acceleration for inference speed.

Chapter 7
Conclusion

7.1 Summary of Findings

The CvT-based Sign Language Recognition model successfully classifies Indian Sign Language (ISL) alphabets with high accuracy. The model benefits from effective preprocessing, data augmentation, and fine-tuning, ensuring robust performance across varied input conditions. Early stopping and model checkpointing optimized the training process and prevented overfitting. Evaluation metrics — accuracy, precision, recall, and F1 score — demonstrated balanced performance, even in the presence of class imbalances.

7.2 Key Insights

- **Effective Use of CvT Architecture**: The Convolutional Vision Transformer (CvT) model excels at sign language recognition by combining convolutional and transformer features to capture both local spatial details and global contextual patterns in ISL images.
- **Image Preprocessing & Augmentation**: Techniques such as resizing, normalization, horizontal/vertical flipping, and color jittering significantly improve generalization, reduce overfitting, and enhance robustness to different lighting and background conditions.
- **Optimized Training with Early Stopping**: Incorporating early stopping and model checkpointing streamlined training, retained the best model checkpoints, and prevented unnecessary epochs that could lead to overfitting.

7.3 Benefits

The development of a real-time ISL recognition application offers several benefits:

- **Enhanced Communication:** Provides a practical solution for seamless and efficient communication between individuals using Indian Sign Language and those unfamiliar with it, fostering inclusivity and understanding.
- **Real-Time Processing:** The ability to translate gestures on-the-go reduces delays, making it suitable for dynamic and interactive use cases such as live conversations and educational tools.
- **Scalability:** By transitioning from a static image-based model to a live video-based system, the application can address a broader range of real-world scenarios, including varying lighting conditions and diverse user profiles.
- **Foundation for Future Work:** Establishes a robust framework for integrating additional features, such as recognition of dynamic gestures and multi-language support, enhancing the system's capabilities over time.

References

[1] N. M. Alharthi and S. M. Alzahrani. Vision transformers and transfer learning approaches for arabic sign language recognition. *Applied Sciences*, 13(21):11625, 2023.

[2] N. C. Camgöz, O. Koller, S. Hadfield, and R. Bowden. Sign language transformers: Joint end-to-end sign language recognition and translation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10023–10033, June 2020.

[3] M. D. Coster, M. V. Herreweghe, and J. Dambre. Sign language recognition with transformer networks. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 6018–6024, Marseille, France, May 2020. European Language Resources Association.

[4] I. Culjak, D. Abram, T. Pribanic, H. Dzapo, and M. Cifrek. A brief introduction to opencv. In *2012 Proceedings of the 35th International Convention MIPRO*, pages 1725–1730, 2012.

[5] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

[7] S. Katoch, V. Singh, and U. S. Tiwary. Indian sign language recognition system using surf with svm and cnn. *Array*, 14:100141, Apr. 2022.

[8] O. Koller, S. Zargaran, H. Ney, and R. Bowden. Deep sign: Enabling robust statistical continuous sign language recognition via hybrid cnn–hmms. *International Journal of Computer Vision*, 126:1081–1098, Dec. 2018.

[9] D. Kothadiya, C. Bhatt, T. Saba, and A. Rehman. Signformer: Deepvision transformer for sign language recognition. *IEEE Access*, 11:1–12, 2023.

[10] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. G. Yong, J. Lee, W.-T. Chang, W. Hua, M. Georg, and M. Grundmann. Mediapipe: A framework for building perception pipelines, 2019.

[11] S. Masood, A. Srivastava, H. Thuwal, and M. Ahmad. Real-time sign language gesture (word) recognition from video sequences using cnn and rnn. In *Proceedings of the 2018 International Conference on Signal Processing and Communication (SPCOM)*, pages 623–632, Jan. 2018.

[12] K. O'Shea and R. Nash. An introduction to convolutional neural networks, 2015.

[13] M. M. Rahman, M. Islam, M. H. Rahman, R. Sassi, M. Rivolta, and M. Aktaruzzaman. A new benchmark on american sign language recognition using convolutional neural network. In *Proceedings of the IEEE International Conference on Smart Technologies for Intelligent Systems (STI)*, pages 1–6, Apr. 2020.

[14] J. Shin, A. S. M. Miah, M. A. M. Hasan, K. Hirooka, K. Suzuki, H.-S. Lee, and S.-W. Jang. Korean sign language recognition using transformer-based deep neural network. *Applied Sciences*, 13(5):3029, 2023.

[15] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision, 2015.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, 2023.

[18] H. Wu, B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, and L. Zhang. Cvt: Introducing convolutions to vision transformers, 2021.

[19] K. Yin and J. Read. Attention is all you sign: Sign language translation with transformers, 2020. arXiv:2012.16047.