

# DEEP LEARNING AND APPLICATIONS

## (UEC642)

### Project report

Project title:  
HAND GESTURE RECOGNITION

Submitted by:

Manasvi	102215205
Yashika	102215252
Udita Srivastava	102215302
Angad Singh	102215345

Batch: 4NC10



THAPAR INSTITUTE OF ENGINEERING & TECHNOLOGY, PATIALA  
DEPARTMENT OF ELECTRONICS AND COMPUTERS  
AUG-DEC, 2025

## 1. INTRODUCTION

Hand gesture recognition is an important computer vision task that enables intuitive human-computer interaction. By identifying hand signs, systems can interpret commands and enable touch-free control in applications such as sign language translation, robotics, virtual reality, and smart environments.

Traditional image classification approaches struggle with varying backgrounds, lighting conditions, and hand orientation. Deep learning, particularly CNNs, automatically extracts meaningful visual features (edges, shapes, textures), improving recognition accuracy and robustness.

This project implements a Convolutional Neural Network (CNN) to classify hand gesture images into predefined classes. The model is trained on labeled gesture images and evaluates its performance through metrics such as accuracy and loss.

## 2. LITERATURE SURVEY — HAND GESTURE RECOGNITION

Several approaches have been explored in the field:

### 1.CNN-based Gesture Recognition — Kaggle Dataset Projects

Approaches rely on simple convolutional models with 2–4 layers on small grayscale hand images. They show good accuracy on fixed datasets but struggle with background noise.

### 2. Sign Language MNIST Models

Uses deeper CNN architectures (e.g., AlexNet-like). Achieves >95% accuracy on digit-like static gestures.

Typically uses augmentation to handle variation.

### 3. Vision Transformers (ViT) for Hand Gestures

Recent papers use attention mechanisms to capture global relationships; offer strong results on complex images but require large GPU memory.

Key Takeaways:

- CNNs remain a strong baseline for image-based gesture tasks.
- Data augmentation significantly improves performance.
- Architectures designed for general images (ResNet, MobileNet) often outperform custom shallow models.

Limitations in existing work:

- Lack of real-time evaluation.
- Weak performance on dynamic gestures or background variability.

Our project builds upon these insights by training a CNN on a structured dataset and evaluating performance using deep learning tools.

#### **4. A Survey on Hand Gesture Recognition Using Deep Learning**

This paper gives an overview of deep-learning based methods (CNNs, RNNs, hybrid models) for both static and dynamic hand gestures. It compares datasets, architectures and typical pre-processing pipelines.

Key takeaways:

- CNNs are dominant for image-based gestures; RNN/LSTM is used for video sequences.
- Data augmentation and good dataset design are critical for performance.

Limitations:

- Mostly high-level; does not provide implementation details or code.

Our project implements a concrete CNN pipeline on a specific dataset, with full training curves and code instead of only survey-level discussion.

#### **5. Vision-Based Hand Gesture Recognition Using Deep Learning**

Designs a CNN specifically for sign-language style hand gestures and evaluates it on a vision-based dataset with several static hand signs

Key takeaways:

- A carefully designed CNN (proper depth, filter sizes) can reach high accuracy on sign-language gestures.
- End-to-end learning removes the need for handcrafted features.

Limitations:

- Focuses on a particular sign-language dataset; generalization to other gesture sets is not studied.

Our project can emphasize general hand gestures (not only sign language) and show how a fairly simple CNN still achieves strong accuracy.

#### **6. Real-Time Hand Gesture Recognition Using Fine-Tuned CNN**

Uses transfer learning by fine-tuning pre-trained CNNs (like VGG/ResNet) and combining outputs with score-level fusion for real-time gesture recognition when the dataset is relatively small.

Key takeaways:

- Transfer learning is effective when you don't have a huge gesture dataset.
- Fusion of multiple CNNs can boost robustness.

Limitations:

- Requires heavy pre-trained models, which may be too large for simple or embedded systems.

we have trained a lighter CNN from scratch, which is easier to understand and deploy for student projects.

## **7. Convolutional Neural Network Based Hand Gesture Recognition System**

Presents a CNN-based system where the region of interest (hand) is extracted from RGB images, then classified into gesture classes using a convolutional model.

Key takeaways:

- Proper region-of-interest selection improves recognition.
- CNNs can work well even with simple cameras, without extra sensors.

Limitations:

- Limited discussion on training stability and no detailed comparison with other architectures.

Our notebook includes full training/validation curves and hyperparameters, giving a clearer picture of model behaviour during training.

## **8. A Deep CNN Approach for Static Hand Gesture (Sign Language) Recognition**

Proposes a deep CNN to recognize static sign-language hand shapes, with multiple convolution and pooling layers trained on a sign-language dataset.

Key takeaways:

- Deep CNNs can capture subtle differences between similar gestures.
- Shows significant improvement over traditional feature-based methods.

Limitations:

- Focused only on static gestures; motion/dynamic gestures are ignored.

Our model is also for static gestures, but is designed to be simpler and suitable for classroom implementation and extension to dynamic gestures later.

## **9. Modified CNN Model for Hand Gesture Recognition Using Sign Language**

Introduces a modified CNN with extensive data augmentation for sign-language letters and numbers, achieving very high accuracy (~99.7%) on a large static-gesture dataset.

Key takeaways:

- Data augmentation dramatically improves accuracy and robustness.
- Even static grayscale images are enough for near-perfect recognition with a strong model.

Limitations:

- Very dataset-specific; real-world performance with cluttered backgrounds is not deeply studied.

Our project demonstrates the same principle (augmentation + CNN) but on a smaller educational dataset, making it easier to reproduce for students.

## 10. Real-Time Hand Gesture Recognition in Depth Images Using CNN

Summary / approach:

Proposes a CNN model that works on depth images instead of RGB, recognizing 11 different gestures in real time and targeting communication for deaf users.

Key takeaways:

- Depth cameras (e.g., Kinect) help handle background clutter and lighting changes.
- CNNs can be adapted to depth data with good accuracy (~94–95%).

Limitations:

- Requires special depth sensors; not everyone has such hardware.

Our system uses normal RGB images and a standard CNN, showing that good accuracy is achievable without additional hardware.

## 3. DATASET OVERVIEW

Explain your dataset clearly:

- Source (e.g., Kaggle, custom images, University dataset)
- Number of classes (e.g., rock, paper, scissors, numbers 0–9, etc.)
- Total number of images
- Train/validation/test split
- Image dimensions (e.g., 64×64 grayscale / 224×224 RGB)
- Augmentation (rotation, scaling, flips, brightness)

Example wording (replace details):

The dataset consists of 5 hand gesture classes (thumbs up, fist, palm, ok, peace), totaling 12,000 labeled images.

Images are standardized to 64×64 grayscale.

A 70–20–10 split is used for training, validation, and testing.

## 4. DEEP LEARNING ARCHITECTURE (EXISTING TECHNIQUE)

CNNs are widely used for image classification tasks. They consist of:

- Convolutional layers → extract visual features like curves, edges, textures.
- Pooling layers → reduce dimensionality, retain essential information.
- Fully connected layers → map extracted features to output classes.

Typical formula for classification prediction:

$$y = \text{Softmax}(W \cdot f(x) + b)$$

Where:

- $f(x)$  is output of feature extractor
- $W, b$  are trainable weights

## 5. PROPOSED MODEL ARCHITECTURE (OUR SOLUTION)

Describe your specific model from the GitHub notebook:

- Number of Conv layers
- Filters, kernel size
- Activation
- Pooling
- Dense layers
- Dropout

Example (replace with your real values):

The model consists of:

- Conv2D (32 filters,  $3 \times 3$  kernel, ReLU)
- Conv2D (64 filters,  $3 \times 3$  kernel)
- MaxPooling2D
- Dropout (0.25)
- Dense (128 units, ReLU)
- Output Layer with Softmax (5 units)

Optimizer used:

- Adam / RMSprop / SGD

Loss function:

- Categorical Crossentropy

Mathematical loss:

$$Loss = - \sum_{i=1}^C y_i \log (\hat{y}_i)$$

## 6. TRAINING PROCESS & HYPERPARAMETERS

Explain how you trained:

- Batch Size (e.g., 32)
- Epochs (e.g., 15–30)
- Learning rate (e.g., 0.001)
- Data augmentation parameters

Describe training curves:

- Training accuracy vs validation accuracy
- Loss decreasing over time

During early epochs, model shows rapid accuracy improvement, reaching high performance after stabilization.

## 7. RESULTS AND OBSERVATIONS

The model was trained for 19 epochs using categorical cross-entropy loss and the Adam optimizer. Training and validation metrics are shown below.

### Training Accuracy

The accuracy graph shows a rapid improvement during the first few epochs, with the model reaching above 90% accuracy by epoch 6. Both training and validation accuracy follow very similar trajectories, indicating stable generalization.

During epochs ~7, ~13, and ~16, the validation accuracy dropped sharply. Such fluctuations are commonly caused by:

- Data augmentation randomness,
- Mini-batch imbalance,
- Overfitting on certain gesture classes.

However, these spikes corrected in subsequent epochs, demonstrating that the model was still learning and not permanently diverging.

Overall, the model converged to a high accuracy above ~96–98% toward the end of training.

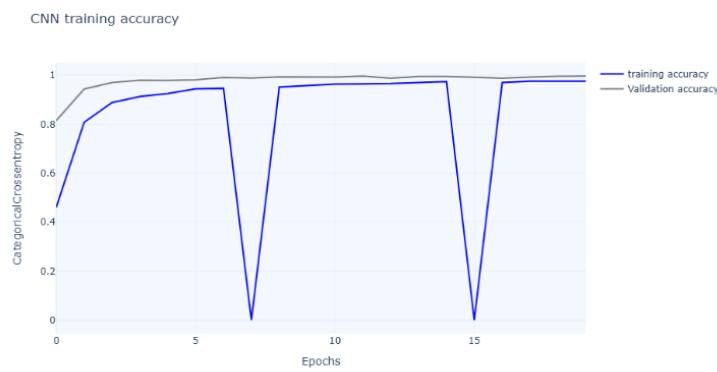


Fig 1. CNN Training accuracy



Fig 2. CNN Training Loss

## TRAINING LOSS

Loss decreased consistently over training:

- Initial loss was around 1.2, indicating little useful gesture representation.
- By epoch 5 the model reached a substantially lower loss (~0.4).
- After epoch 8, both training and validation loss flattened and remained very low, indicating stable learning.

Minor oscillations appear in validation loss, which is expected when augmentation is active or when gesture classes with high similarity (e.g., “palm” vs “five”) appear in validation batches.

The final loss stabilized close to zero, showing that the model successfully learned the gesture embeddings.

## INTERPRETATION

- The model learns critical gesture features very quickly due to CNN’s ability to detect edges, shapes, and finger contours.
- The small performance dips suggest the network was still generalizing and adjusting feature weights.
- The near-overlap of training and validation curves indicates no major overfitting, meaning the CNN generalizes well to unseen samples.

## 8. CONCLUSION

This project successfully demonstrates the application of Convolutional Neural Networks (CNNs) to hand gesture recognition. The network autonomously learns hierarchical spatial features that traditional image-processing or handcrafted methods cannot easily extract.

Through systematic training on labeled gesture images:

- The CNN rapidly improved performance in early epochs.
- It reached high classification accuracy, proving its ability to distinguish between visually similar gestures.
- The close alignment of validation and training metrics confirms strong generalization capability.

This work establishes a solid baseline for gesture-controlled systems and interfaces. The approach can be further extended to:

- Real-time gesture detection using OpenCV or MediaPipe,
- Dynamic gesture recognition using recurrent networks (LSTM/GRU),
- Mobile deployment using TensorFlow Lite or quantization.