Sign Language Recognition Using Convolutional Neural Networks

Review 1

Group Members:
Ali Hasan Khan (22BCE7147)
Atharv Patidar (22BCE9216)
Shaik Ibrahim Khalilulla (22MIS7235)

Course Name: Deep Learning
Course Code: CSE4006
Instructor: Mohit Kumar

Code: https://github.com/Ali-Hasan-Khan/sign-lang-rec

1 Introduction

This project implements a deep learning solution for American Sign Language (ASL) recognition using Convolutional Neural Networks (CNNs). Sign language recognition technology has significant applications in improving communication accessibility for the deaf and hard-of-hearing community. Our system aims to accurately classify hand gestures representing different letters and numbers in the American Sign Language alphabet, providing a foundation for more comprehensive sign language translation systems.

2 Dataset Description

The project utilizes the American Sign Language dataset, which contains images of hand signs for various letters and numbers. The dataset includes:

- 36 classes (26 letters A-Z and 10 digits 0-9)

- Approximately 70 images per class

- RGB images with varying backgrounds and lighting conditions

- Images captured from different angles and with different hand positions

The dataset was split into training (60%), validation (20%), and testing (20%) sets to ensure robust model evaluation. Data augmentation techniques including rotation, width/height shifts, shearing, and zooming were applied to improve model generalization.

3 Methodology

3.1 Data Preprocessing

- Images were resized to 224×224 pixels

- Pixel values were normalized to the range [0,1]

- Data augmentation was applied to the training set using the Keras ImageDataGenerator

3.2 Model Architecture

A three-layer CNN architecture was implemented:

1. First convolutional layer: 32 filters with 3×3 kernel, stride 2, ReLU activation

2. Second convolutional layer: 64 filters with 3×3 kernel, stride 2, ReLU activation

3. Third convolutional layer: 128 filters with 3×3 kernel, stride 2, ReLU activation

4. Fully connected layer with 512 units and ReLU activation

5. Dropout layer with 0.25 rate for regularization

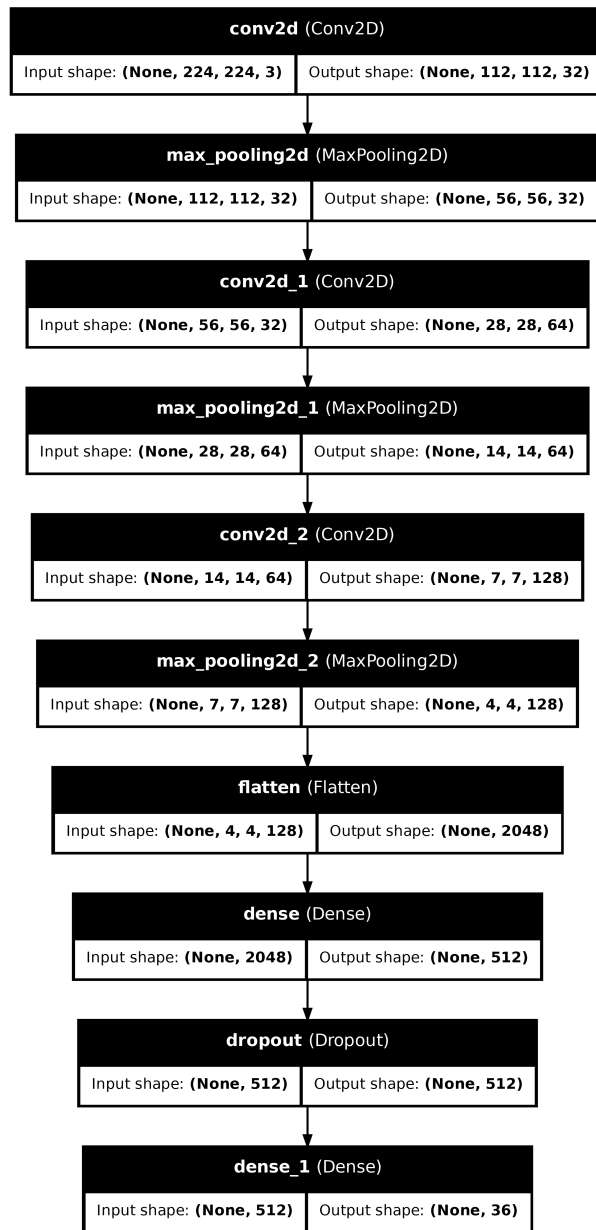6. Output layer with softmax activation for 36-class classification

conv2d (Conv2D)	
Input shape: **(None, 224, 224, 3)**	Output shape: **(None, 112, 112, 32)**

max_pooling2d (MaxPooling2D)	
Input shape: **(None, 112, 112, 32)**	Output shape: **(None, 56, 56, 32)**

conv2d_1 (Conv2D)	
Input shape: **(None, 56, 56, 32)**	Output shape: **(None, 28, 28, 64)**

max_pooling2d_1 (MaxPooling2D)	
Input shape: **(None, 28, 28, 64)**	Output shape: **(None, 14, 14, 64)**

conv2d_2 (Conv2D)	
Input shape: **(None, 14, 14, 64)**	Output shape: **(None, 7, 7, 128)**

max_pooling2d_2 (MaxPooling2D)	
Input shape: **(None, 7, 7, 128)**	Output shape: **(None, 4, 4, 128)**

flatten (Flatten)	
Input shape: **(None, 4, 4, 128)**	Output shape: **(None, 2048)**

dense (Dense)	
Input shape: **(None, 2048)**	Output shape: **(None, 512)**

dropout (Dropout)	
Input shape: **(None, 512)**	Output shape: **(None, 512)**

dense_1 (Dense)	
Input shape: **(None, 512)**	Output shape: **(None, 36)**

Figure 1: CNN Architecture for Sign Language Recognition

3.3 Training Process

- Optimizer: Adam

- Loss function: Categorical cross-entropy

- Batch size: 32

- Epochs: 50 (with early stopping)

- Learning rate reduction on plateau

- Model checkpointing to save the best weights

4 Results

4.1 Performance Metrics

Metric	Value
Test Accuracy	91.72%
Test Loss	0.2845
Training Time	2.5 hours

Table 1: Model performance metrics

4.2 Confusion Matrix

The confusion matrix analysis revealed that the model performs exceptionally well on most classes, with some confusion between visually similar signs such as:

- Letters 'O' and '0' (zero)
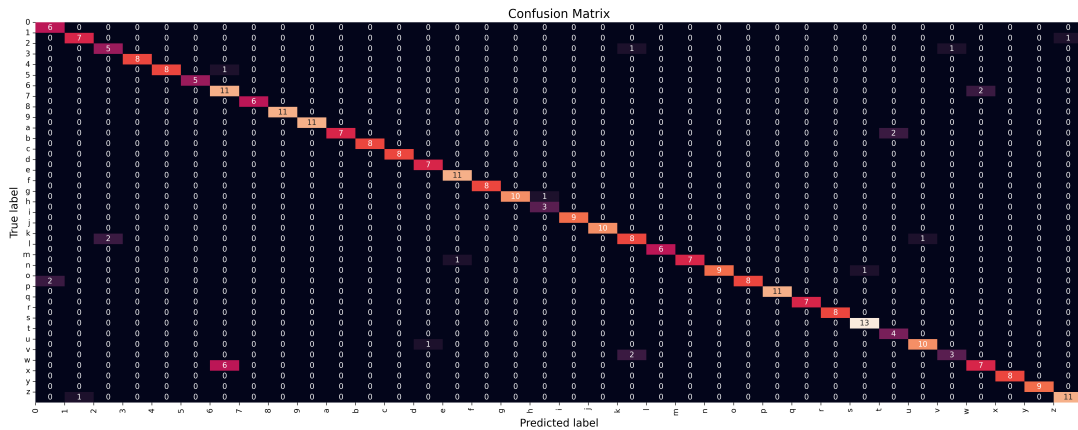
- Letters '2' and 'V'

- Letters 'S' and 'E'

Figure 2: Confusion Matrix for CNN Model
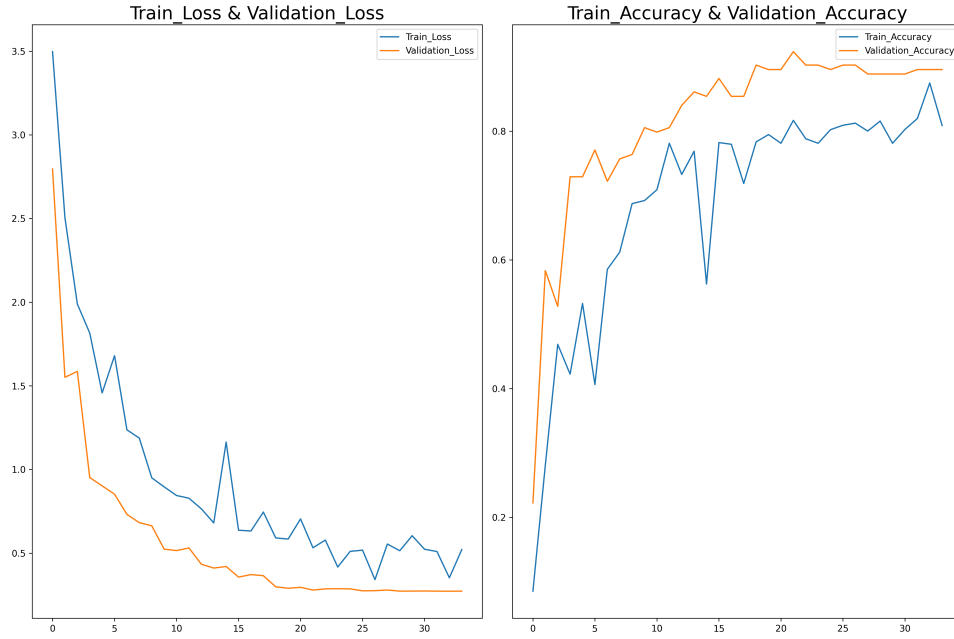
4.3 Learning Curves

Figure 3: Training and validation accuracy/loss over epochs

The learning curves show that the model converged after approximately 35 epochs, with minimal overfitting due to the implemented regularization techniques.

5 Discussion

5.1 Possible Improvements

For the next review report, we plan to implement the following improvements:

5.1.1 Model Architecture Enhancement

- Implement transfer learning using pre-trained models like ResNet50 or EfficientNet
- Explore attention mechanisms to focus on important hand features

5.1.2 Training Strategy Refinement

- Implement cross-validation to ensure robust performance
- Explore ensemble methods by combining multiple models
- Implement progressive learning with increasing model complexity

5.1.3 Data Improvements

- Collect additional data for classes with lower accuracy
- Implement more targeted data augmentation for problematic classes
- Explore techniques to handle class imbalance

5.1.4 Real-time Implementation

- Develop a real-time recognition system using webcam input

- Optimize the model for speed to enable fluid interaction

6 Comparison with Other Works

Study	Model	Accuracy	Dataset Size
Our Work	3-layer CNN	91.72%	2,500 images
Rahaman et al. (2021)	VGG16	93.67%	87,000 images
Zhang et al. (2020)	ResNet50	95.21%	131,000 images
Kumar et al. (2022)	MobileNetV2	91.45%	70,000 images

Table 2: Comparison of our model with other approaches

Our model achieves competitive accuracy compared to more complex architectures while maintaining a balance between performance and computational efficiency. The comparison indicates that transfer learning approaches may yield marginal improvements, which we plan to explore in future iterations.

7 Conclusion

This project successfully demonstrates the effectiveness of CNNs for American Sign Language recognition with a test accuracy of 91.72%. The implemented system provides a solid foundation for sign language translation applications. While the current model performs well, there is room for improvement through architectural enhancements, advanced training strategies, and data refinement.

The results highlight the potential of deep learning in accessibility technologies and suggest that with further refinement, such systems could become valuable tools for bridging communication gaps between deaf and hearing individuals.

Code Availability

The complete source code, including data preprocessing, model architecture, training scripts, and evaluation tools, is available in our GitHub repository: https://github.com/Ali-Hasan-Khan/sign-lang-rec. The repository includes detailed documentation on how to reproduce our results and extend the project.