Image Classification Using Convolutional Neural Networks

Dara Lenaghan - G00385153

Gesture UI - Assignment 2

Abstract

This document gives an overview of a project aimed at building and evaluating Convolutional Neural Network (CNN) models for image classification. The goal is to achieve high accuracy in classifying images into predefined categories. The document covers the methodology, data preprocessing techniques, model training processes, and evaluation results.

1 Introduction

With the rapid progress in computer vision and deep learning, machines can now recognise and classify visual content with high accuracy. This project explores using different CNN models to classify hand gesture images into 18 distinct categories, which is important for applications like human-computer interaction, sign language interpretation, and virtual reality.

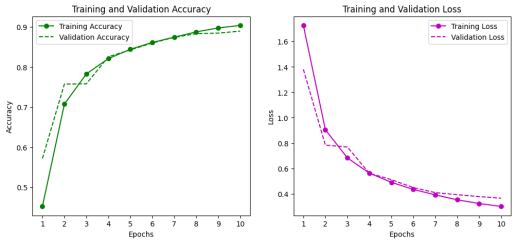

Figure 1: Training and Validation Accuracy and Loss for Light CNN

We tested several CNN architectures, including Light CNN, Heavy CNN, Greyscale CNN, Data Augmentation CNN, and VGG-16. These models were designed and trained to perform well on our dataset of 125,912 hand gesture images, each labelled with one of 18 classes, making it a challenging task due to gesture similarities.

The main goals of this study were:

1. To compare the performance of different CNN architectures in terms of accuracy, training time, and inference time.
2. To examine the impact of data augmentation and input image size on model performance.
3. To identify the strengths and weaknesses of each model through detailed analysis and visualisation.

2 Methodology

This section explains the methodology used in the study, covering everything from data preprocessing to model training and evaluation. The approach is structured to ensure a thorough understanding and comparison of the different CNN models for hand gesture classification.

2.1 Data Pre-Processing

The dataset used for this study consists of 125,912 images of hand gestures, categorised into 18 classes. Preprocessing steps were crucial to prepare the data for training and included:

1. **Image Resising**: All images were resized to either 64x64 or 128x128 pixels, depending on the model. This resizing standardises the input size, making it compatible with the CNN architectures.

2. **Normalisation**: Pixel values were normalised to a range of [0, 1] using TensorFlow's Rescaling layer. Normalisation speeds up the training process by ensuring that the input features are on a similar scale [1].

3. **Data Splitting**: The dataset was split into training (70%), validation (10%), and testing (20%) subsets. This split helps in evaluating the model's performance on unseen data and prevents overfitting.

2.2 Data Labelling

The dataset's labels were provided in categorical format, representing different hand gestures. Each image was associated with one of the 18 class labels. One-hot encoding was used to convert these labels into a format suitable for training the models using categorical cross-entropy loss.

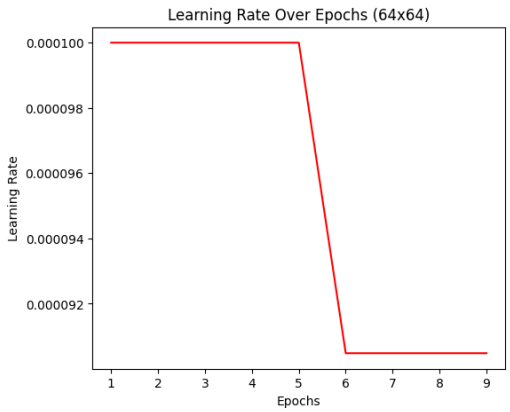

Figure 2: Learning Rate Over Epochs (64x64)

2.3 Data Scaling

All images were scaled to fit the input requirements of the models. For example, the input images for VGG-16 were resized to 128x128 pixels to leverage the model's ability to handle higher resolution images, potentially capturing more detailed features.

2.4 Data Analysis and Visualisation

To understand the dataset and model performance better, several data analysis and visualisation techniques were employed:

1. **Class Distribution**: The distribution of images across the 18 classes was analysed to ensure balanced representation and identify any class imbalance issues.

2. **Confusion Matrix**: Used to visualise the model's performance by comparing the true labels with the predicted labels, highlighting areas where the model is making mistakes.

3. **Accuracy and Loss Curves**: Plotted to monitor the training and validation performance over epochs, helping in diagnosing overfitting or underfitting issues.

2.5 Other Techniques

Several additional techniques were implemented to enhance the model's performance and robustness:

1. **Data Augmentation**: Techniques such as random zoom, rotation, horizontal and vertical flips, and random contrast adjustments were applied to artificially increase the diversity of the training set, improving generalisation[2].

2. **Early Stopping**: This callback monitored the validation loss and stopped the training process if the loss did not improve for a specified number of epochs, thereby preventing overfitting.

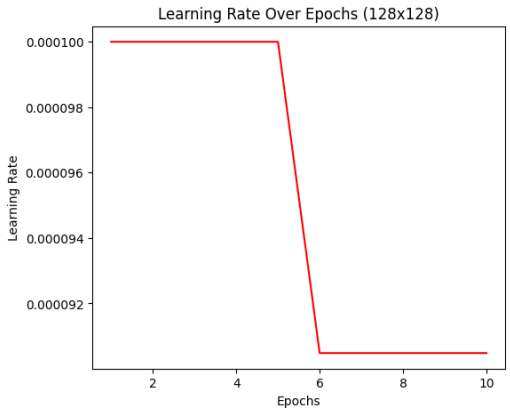

Figure 3: Learning Rate Over Epochs (128x128)

3. **Learning Rate Scheduling**: The learning rate was dynamically adjusted during training using a scheduler, which helps in optimising the training process by reducing the learning rate as the training progresses.

3 Experiments and Results

This section presents the detailed experiments conducted with various CNN models, along with their results. Each model's architecture, training process, and performance metrics are discussed to highlight their effectiveness in classifying hand gesture images.

3.1 Classifier Models

1. **Light CNN**
   (a) **Architecture**:
      - Input layer for 64x64 RGB images.
      - Four convolutional layers with filters of sizes 32, 64, 128, and 256.
      - Max pooling and batch normalisation after each convolutional layer.
      - Dropout layers to prevent overfitting.
      - Dense layer followed by a flattening layer and a softmax output layer.

   (b) **Training**:
      - Learning rate: 0.001 with time-based decay.
      - Early stopping with patience of 3 epochs.
      - Training duration: 10 epochs.

   (c) **Performance**:
      - Validation Accuracy: 88.95%
      - Validation Loss: Consistent decrease over epochs, indicating good learning.

2. **Heavy CNN**
   (a) **Architecture**:
      - Similar structure to Light CNN but with additional layers and higher filter counts.
      - Input layer for 64x64 RGB images.
      - Five convolutional layers with filters of sizes 64, 128, 256, 512, and 512.
      - Max pooling, batch normalisation, and dropout layers interspersed between convolutional layers.
      - Dense layers with 1024 and 512 neurons, followed by a softmax output layer.

   (b) **Training**:
      - Learning rate: 0.0001 with time-based decay.
      - Early stopping with patience of 3 epochs.
      - Training duration: 10 epochs.

   (c) **Performance**:
      - Validation Accuracy: 87.79%
      - Validation Loss: Higher than Light CNN, but shows a consistent decrease.

3. **Greyscale CNN**

(a) **Architecture**:
- Similar to Light CNN but designed to process 64x64 greyscale images.
- Three convolutional layers with filters of sizes 32, 64, and 128.
- Max pooling,
  batch normalisation, and dropout layers.
- Dense layer followed by flattening and softmax output layers.

(b) **Training**:
- Learning rate: 0.001 with time-based decay.
- Early stopping with patience of 3 epochs.
- Training duration: 10 epochs.

(c) **Performance**:
- Validation Accuracy: 80.49%
- Validation Loss: Indicates importance of colour information in distinguishing gestures.

4. **Data Augmentation CNN**

(a) **Architecture**:
- Similar to Light CNN but with added data augmentation layers.
- Input layer for 64x64 RGB images with data augmentation.
- Four convolutional layers with filters of sizes 32, 64, 128, and 256.
- Max pooling, batch normalisation, and dropout layers.
- Dense layer followed by flattening and softmax output layers.

(b) **Training**:
- Learning rate: 0.001 with time-based decay.
- Early stopping with patience of 3 epochs.
- Training duration: 10 epochs.

(c) **Performance**:
- Validation Accuracy: 87.66%
- Validation Loss: Shows improved generalisation due to data augmentation.

5. **VGG-16**

(a) **Architecture**:
- Pre-trained VGG-16 model with additional custom layers for classification.
- Input layer for 64x64 RGB images.
- VGG-16 base with frozen weights.
- Global average pooling layer followed by dense and dropout layers.
- Softmax output layer.

(b) **Training**:
- Learning rate: 0.0001 with exponential decay after 5 epochs.
- Early stopping with patience of 5 epochs.
- Training duration: 10 epochs.

(c) **Performance**:
- Validation Accuracy: 36.81%
- Validation Loss: Higher compared to other models, indicating possible fine-tuning issues.

6. **Enhanced VGG-16**

   (a) **Architecture**:

   - VGG-16 model with increased input size (128x128) and extensive data augmentation.
   - Last four layers of VGG-16 unfrozen for fine-tuning.
   - Global average pooling, dense, and dropout layers.
   - Softmax output layer.

   (b) **Training**:

   - Learning rate: 0.0001 with exponential decay after 5 epochs.
   - Early stopping with patience of 5 epochs.
   - Training duration: 10 epochs.

   (c) **Performance**:

   - Validation Accuracy: 93.64%
   - Test Accuracy: 91.69%
   - Validation Loss: Low and stable, showing strong performance and generalisation [3].

3.2 Results

Accuracy and Loss Curves:

- **Light CNN**: Achieved high validation accuracy with rapid convergence. Suitable for applications requiring a balance between accuracy and computational efficiency.
- **Heavy CNN**: High validation accuracy but longer training and inference times. Best suited for applications where higher accuracy is critical and computational resources are not a constraint.
- **Greyscale CNN**: Lower validation accuracy, indicating that colour information is significant for distinguishing hand gestures.
- **Data Augmentation CNN**: Improved generalisation due to augmented training data. Shows the importance of data variability in training robust models[4].
- **VGG-16**: Lower validation accuracy, suggesting potential issues with the transfer learning approach or dataset complexity.
- **Enhanced VGG-16**: Highest validation and test accuracy, demonstrating the effectiveness of increased input size and extensive data augmentation.

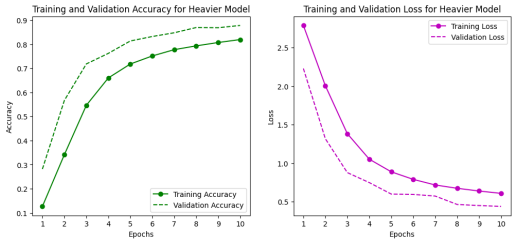

Figure 4: Training and Validation Accuracy and Loss for Heavy CNN

Confusion Matrix Analysis:

- The Enhanced VGG-16 model's confusion matrix shows high classification accuracy for most classes, with minimal misclassifications. This model excels in distinguishing between similar hand gestures, making it highly suitable for real-world applications requiring precise gesture recognition.

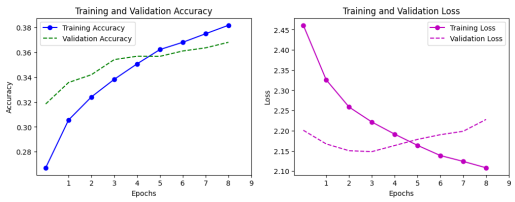

Figure 5: Training and Validation Accuracy and Loss for VGG-16

4 Conclusion

The experiments show that model complexity, input image size, and data augmentation have a big impact on CNN performance in hand gesture classification. The Enhanced VGG-16 model outperformed the other models, achieving the highest accuracy and showing excellent generalisation capabilities. However, simpler models like the Light CNN provide a good balance between accuracy and computational efficiency, making them suitable for real-time applications. This study highlights the importance of choosing the right model architectures and preprocessing techniques to optimise performance for specific tasks. Future work could explore further fine-tuning strategies and the use of more advanced data augmentation techniques to enhance model robustness and accuracy[5].

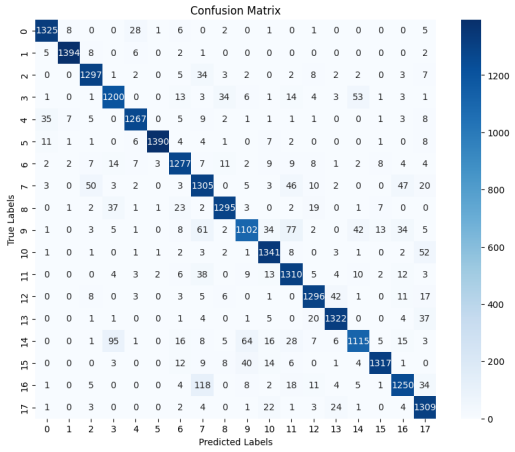

Figure 6: Confusion Matrix for VGG-16 (128x128)

References

[1] C. Shorten and T. M. Khoshgoftaar. A survey on image data augmentation for deep learning. *J Big Data*, 6:60, 2019.

[2] L. Perez and J. Wang. The effectiveness of data augmentation in image classification using deep learning. *IEEE Access*, 5:25926–25938, 2017.

[3] R. Poojary, R. Raina, and A. K. Mondal. Effect of data-augmentation on fine-tuned cnn model performance. *IAES Int J Artif Intell*, 10(1):84–92, 2021.

[4] R. Keshari, A. Agarwal, S. Chaudhary, and R. Singh. Data augmentation for deep learning with cnn: A review. *Neurocomputing*, 361:119–137, 2020.

[5] Leiyu Chen, Shaobo Li, Qiang Bai, Jing Yang, Sanlong Jiang, and Yanming Miao. Review of image classification algorithms based on convolutional neural networks. *Remote Sens.*, 13(22):4712, 2021.