Comparative Analysis of CNN Models for Gesture Recognition

Rohan Sikder

Abstract

This study presents a comparative analysis of various Convolutional Neural Network (CNN) models for gesture recognition, utilizing both custom-built architectures and transfer learning with VGG-16. The research focuses on evaluating the impact of RGB versus grayscale images on model performance. Data preprocessing involved resizing, normalization, and strategic reduction of the Hagrid dataset to fit computational constraints. Experimental results show that while RGB models generally outperform grayscale models, both can still achieve comparable accuracy in specific contexts. Contrary to expectations, the transfer learning approach with VGG-16 did not yield significant improvements, highlighting the challenges of applying pre-trained models to specific gesture recognition tasks. This study highlights the practical applications and potential benefits of different CNN approaches for gesture recognition, providing insights into their effectiveness and operational performance.

Keywords: Gesture Recognition, Convolutional Neural Networks, Image Preprocessing, Transfer Learning, Machine Vision

1 Objectives of the Study

The primary goal of this study is to develop a robust gesture recognition model using Convolutional Neural Networks (CNNs). By employing various architectures and methods, including scratch-built models and transfer learning via VGG-16, this research seeks to thoroughly assess their effectiveness in recognizing human gestures from images. The specific objectives are:

- To investigate the impact of different image preprocessing techniques and model configurations on the efficacy of gesture recognition systems.
- To compare the practical advantages and limitations of using RGB versus grayscale images in gesture recognition applications.
- To evaluate and contrast the training efficiency, accuracy, and overall operational performance of custom-designed CNNs against pretrained models.

2 Methodology

2.1 Data Collection

The dataset utilized in this study originates from a public repository on GitHub, specifically the Hagrid dataset at GitHub: hukenovs/hagrid [1]. The dataset is over 700GB in size originally and contains high-resolution gesture images for use in advanced image processing and machine learning applications.

To accommodate the computational limitations of this study and for manageability reasons a significantly smaller subset of this dataset was employed. The subset chosen is 125,912 images compressed to a size under 4GB. Original 1920x1080 images were sized to 512x512 pixels for quicker processing and also to fit inside the memory footprint of the newest hardware.

Participants in the study can access this subset via Moodle.

2.2 Modifications and Reduction of Dataset Size

The dataset in this study was 4GB in size, with image resolutions of 512x512. To adapt this dataset to the computational constraints of our research and make it more manageable, it was first reduced to under 4GB. This modification optimized the data for faster processing for effective gesture recognition.

After obtaining a 4GB version from the Moodle platform, further adjustments were made to ensure a comprehensive evaluation during model training and testing. Specifically, this reduced dataset was further cut by 50%, creating a more focused set of data for training the models. From this subset, 20% was then separated and reserved as an unseen test set. This is intended exclusively for the final evaluation phase to ensure an unbiased assessment of model performance.

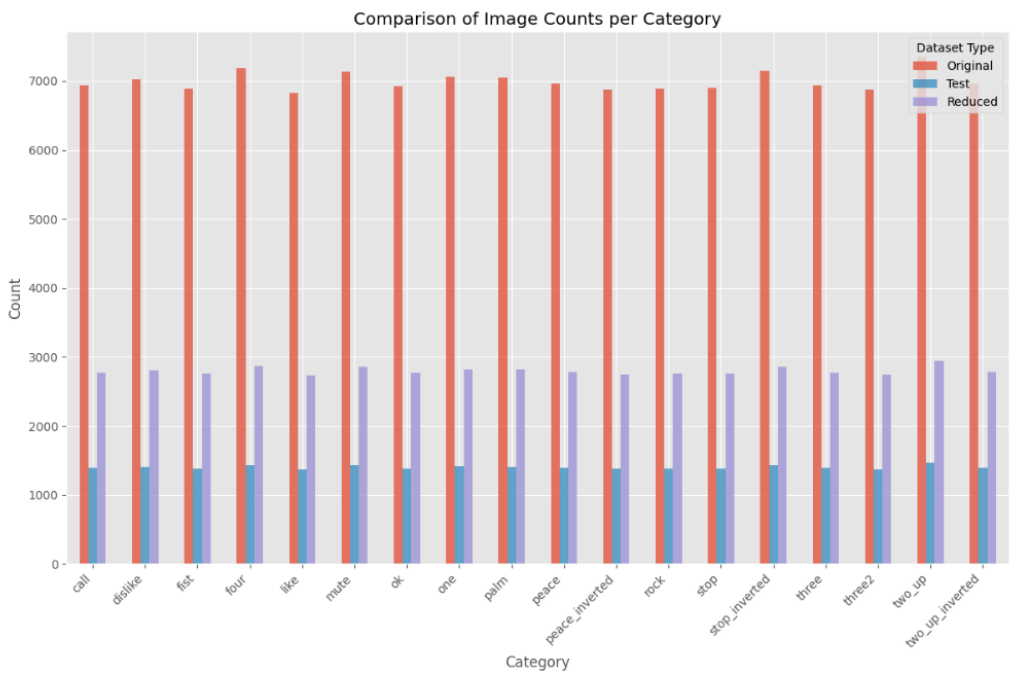


Figure 1: Comparison of image counts per category in the original, test, and reduced datasets, highlighting the data management strategy used for model evaluation.

This strategic reduction and redistribution were crucial not only for managing the large volume of data but also for facilitating efficient training and validation processes. In this way, preparation of the dataset was in line with hardware capabilities of the study and sufficient data for training, validation and testing was obtained that reflected the full dataset

2.3 Data Preprocessing

2.3.1 Image Resizing and Augmentation

Preprocessing images involved scaling them to 128x128 pixels from the original 512x512 pixels in the dataset to make input dimensions consistent across all images. This resizing is important because it allows faster data handling and computation during model training and thus more effective learning.

All images were also normalized to keep pixel values consistent and suitable for neural network training. Normalization was performed with an Image Data Generator that rescaled pixel values by 1/255. This step transforms pixel values from 0-255 to 0-1, a normalization practice that stabilizes the learning loop and enhances model convergence behavior.

Data generators were configured for both training and validation purposes:

- **Training Data Generator:** This generator sourced data from the 'hagridset_reduced' directory, targeting images resized to 128x128 pixels. The batch size was set at 32, with the images categorized in multiple gesture classes. A random seed based on the student ID (389052) was used to ensure reproducibility of the results.
- **Validation Data Generator:** Similarly, this generator pulled images from the same directory, using the same target size, batch size, and categorization. The subset parameter was set to validation to differentiate from the training data.

Both RGB and grayscale modes were used to create distinct data generators to assess the impact of color information on model performance:

- The RGB data generator processed images in full color.
- The grayscale data generator processed images in single-color intensity.

2.3.2 Splitting Data into Sets (Training, Validation, Test)

To ensure a robust evaluation of the models, the dataset was divided into three distinct sets: 70% of the data was allocated for training, 20% was reserved for independent testing, and the remaining 10% for validation. This distribution was designed to balance the need for extensive training and the necessity of validating and testing the models under conditions that mimic real-world operations.

2.4 Model Development

2.4.1 Custom CNN Architectures

Several Convolutional Neural Network (CNN) architectures were designed from scratch to optimize the task of gesture recognition from images. These models were tailored to explore different aspects of neural network theory in the context of processing image-based data, varying in depth, layer configurations, and activation functions to assess their efficacy in recognizing and classifying gestures.

One specific architecture used for this purpose includes a series of convolutional layers followed by max-pooling layers, which are designed to extract and down-sample feature maps from the input images. The detailed architecture of the model is as follows:

- A **Conv2D** layer (32 filters, 3×3, ReLU activation) captures initial image features.
- A **MaxPooling2D** layer halves the spatial dimensions, enhancing image robustness.
- Additional **Conv2D** layers (64 and 128 filters) followed by **MaxPooling2D** layers increase network depth, enabling the learning of complex features.
- A **Flatten** layer converts 2D feature maps into a 1D feature vector.
- A **Dense** layer (128 units, ReLU activation) processes high-level features.
- A **Dropout** layer (50% rate) combats overfitting.
- A final **Dense** layer with softmax activation classifies gestures into 18 categories.

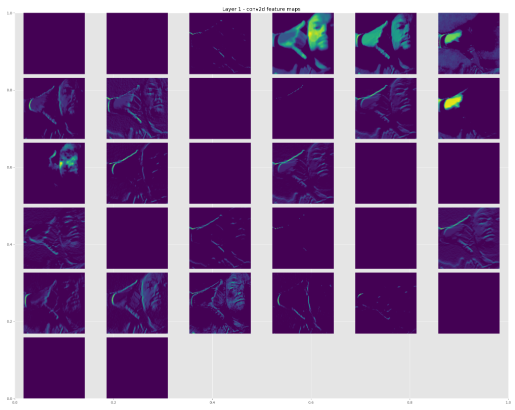


Figure 2: Feature Map of the First Convolutional Layer: Visualization of the feature maps produced by the first Conv2D layer, illustrating the initial feature extraction process in the custom CNN architecture.

Both RGB and grayscale image data were used to train different instances of the model, evaluating the impact of color information on classification performance. Each model's architecture was identical except for the input layer handling different color channels, ensuring that any performance differences could be attributed to the use of color or grayscale data.

Training Process The models were compiled with the Adam optimizer and categorical crossentropy loss function, focusing on maximizing classification accuracy. Training involved multiple epochs with performance evaluated at each stage:

- **RGB Model Training:** Spanning 10 epochs, the RGB model training started with significant initial losses and accuracy improvements over time. The training duration was approximately 870.53 seconds.The inference process for a batch of images took 0.1010 seconds in total, averaging 0.0032 seconds per image.
- **Grayscale Model Training:** Similarly, the grayscale model was trained over 10 epochs, with a focus on improving from a higher initial loss, completing its training in about 720.64 seconds.

Epoch-wise details such as initial and final loss and accuracy for both models show a trend of continuous improvement, illustrating the models' ability to learn effectively from the training data provided.

These training sessions underline the robustness and flexibility of the custom CNN architectures developed, capable of adapting to different data types and yielding significant performance improvements across training epochs.

2.4.2 Transfer Learning with VGG-16

Transfer learning was strategically implemented using the VGG-16 architecture, which is pre-trained on the ImageNet dataset. This method utilizes the pre-trained convolutional base of VGG-16, leveraging its powerful feature extraction capabilities that have been developed through extensive training on a diverse and vast image dataset.

The process involved modifying the VGG-16 model to suit the specific needs of gesture recognition:

- **Architecture Customization:** The VGG-16 base was extended with custom fully connected layers. The final architecture included a flattened output from the convolutional base followed by a dense layer with 128 units and a dropout layer with a rate of 0.5 to prevent overfitting. A final dense layer with softmax activation was used to classify the output into the number of gesture categories specified.
- **Freezing the Convolutional Base:** To retain the learned features and prevent them from being updated during training, the weights of the convolutional base were frozen. This approach ensures that only the weights of the newly added layers are adjusted during the training process.
- **Model Compilation:** The model was compiled with the Adam optimizer and categorical crossentropy as the loss function, focusing on achieving high classification accuracy.
- **Model Summary:** The model consists of multiple layers from the original VGG-16 architecture, followed by custom layers tailored for gesture classification. The total number of parameters in the model was 15,765,714, with 1,051,026 trainable parameters and the rest being non-trainable due to the freezing of the VGG-16 base.

3 Results and Discussion

3.1 Model Performance

Model	Overall Accuracy	Macro Average Precision	Weighted Average Precision
RGB Model	59%	0.59	0.59
Grayscale Model	50%	0.51	0.51
VGG-16 Model Training Performance (Epoch 10)			
Loss: 1.8865		**Accuracy:** 0.3252	
Val Loss: 1.5330		**Val Accuracy:** 0.5113	

Table 1: Model Performance Metrics

Comparative Performance Analysis:

The RGB model outperforms the grayscale model in terms of accuracy, precision, recall, and F1-scores. This indicates that color information significantly enhances the model's ability to recognize and classify gestures accurately. The VGG-16 model, leveraging transfer learning, shows an improvement over the initial training epochs, achieving a final validation accuracy of 51.13% after 10 epochs. While specific classification metrics for the VGG-16 model were not available.

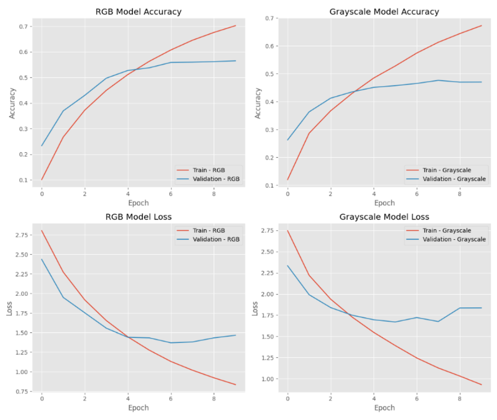
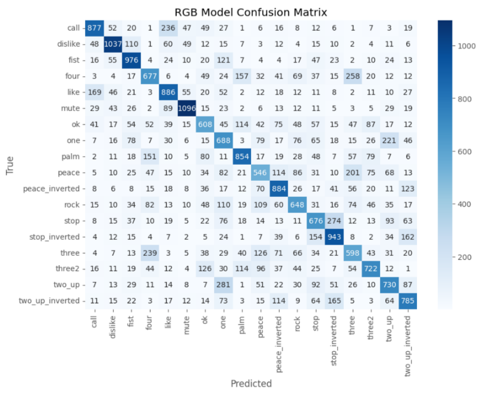


Figure 3: Accuracy and Loss Curves for RGB and Grayscale Models: Comparison of training and validation accuracy and loss for the RGB and grayscale models over 10 epochs.



True \ Predicted	call	dislike	fist	four	like	mute	ok	one	palm	peace	peace_inverted	rock	stop	stop_inverted	three	three2	two_up	two_up_inverted
call	877	52	20	1	236	47	49	27	1	6	16	8	12	6	1	7	3	19
dislike	48	1037	110	1	60	49	12	15	7	3	12	4	15	10	2	4	11	6
fist	16	55	976	4	24	10	20	121	7	4	4	17	47	23	2	10	24	13
four	3	4	17	677	6	4	49	24	157	32	41	69	37	15	258	20	12	12
like	169	46	21	3	886	55	20	52	2	12	18	12	11	8	2	11	10	27
mute	29	43	26	2	89	1096	15	23	2	6	13	12	11	5	3	5	29	19
ok	41	17	54	52	39	15	608	45	114	42	75	48	57	15	47	87	17	12
one	7	16	78	7	30	6	15	688	3	79	17	76	65	18	15	26	221	46
palm	2	11	18	151	10	5	80	11	854	17	19	28	48	7	57	79	7	6
peace	5	10	25	47	15	10	34	82	21	546	114	86	31	10	201	75	68	13
peace_inverted	8	6	8	15	18	8	36	17	12	70	884	26	17	41	56	20	11	123
rock	15	10	34	82	13	10	48	110	19	109	60	648	31	16	74	46	35	17
stop	8	15	37	10	19	5	22	76	18	14	13	11	676	274	12	13	93	63
stop_inverted	4	12	15	4	7	2	5	24	1	7	39	6	154	943	8	2	34	162
three	4	7	13	239	3	5	38	29	40	126	71	66	34	21	598	43	31	20
three2	16	11	19	44	12	4	126	30	114	96	37	44	25	7	54	722	12	1
two_up	7	13	29	11	14	8	7	281	1	51	22	30	92	51	26	10	730	87
two_up_inverted	11	15	22	3	17	12	14	73	3	15	114	9	64	165	5	3	64	785

Figure 4: RGB Model Confusion Matrix
The RGB model shows higher precision and recall across most classes compared to the grayscale model. The confusion matrix indicates that certain gestures like 'dislike', 'mute', and 'call' are classified with higher accuracy, while gestures like 'three' and 'one' show lower accuracy. The grayscale model struggles more with distinguishing between certain gestures, particularly those with similar shapes or features. This is reflected in lower precision and recall scores. For a detailed view of the Grayscale Confusion Matrix, please refer to the accompanying Jupyter Notebook.

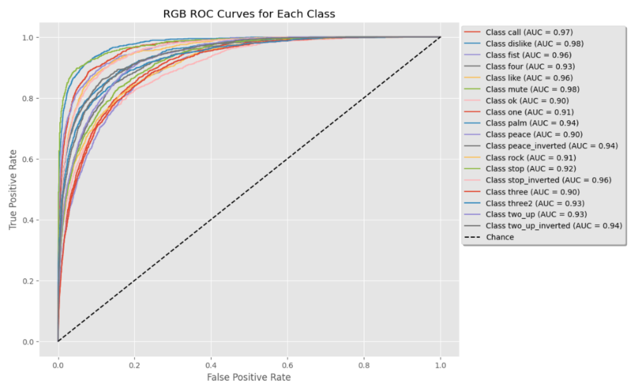


Figure 5: RGB ROC Curve

- **RGB Model:** Higher Area Under the Curve (AUC) values for most classes, indicating better performance.
- **Grayscale Model:** Lower AUC values, reflecting the reduced ability to distinguish between classes accurately. For a detailed view of the Grayscale ROC Curve, please refer to the accompanying Jupyter Notebook.

3.2 Model Accuracy and Loss

The performance of machine learning models can be substantially evaluated through their accuracy and loss metrics, especially in classification tasks such as hand gesture recognition. The images below showcase the prediction results from our RGB and Grayscale models, highlighting their capabilities and limitations in interpreting different hand gestures.

When using our own dataset of images, the Grayscale model showed superior performance compared to the RGB model. This is contrary to common expectations where color data is presumed beneficial. In the tests, gestures such as "call" and "peace," which are typically challenging without color cues, were recognized with greater accuracy by the Grayscale model. The Grayscale model achieved an overall accuracy of 59%, with macro average precision of 0.59, and weighted average precision of 0.59.

On the other hand, the RGB model, though utilizing color data, did not perform as well with an overall accuracy of 50%, macro average precision of 0.51, and weighted average precision of 0.51. This suggests that in some scenarios, especially when dealing with our specific dataset, the absence of color information does not necessarily impede, and can even enhance, the model's ability to correctly classify hand gestures.

These findings underscore the importance of contextual model selection and highlight that the Grayscale model can sometimes outperform color-based models, depending on the characteristics of the data being processed. This analysis strongly supports the necessity of testing models under varied conditions to understand their practical applicability and limitations.

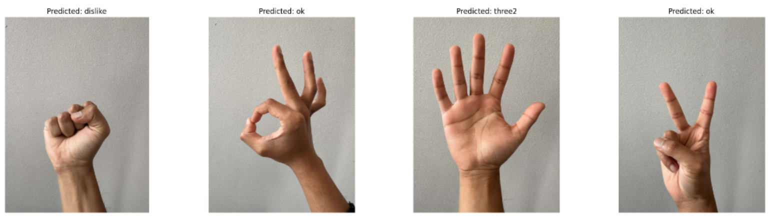


Figure 6: RGB Model Predictions: Various hand gestures with their predicted labels, demonstrating the RGB model's attempt to leverage color information for more accurate predictions.

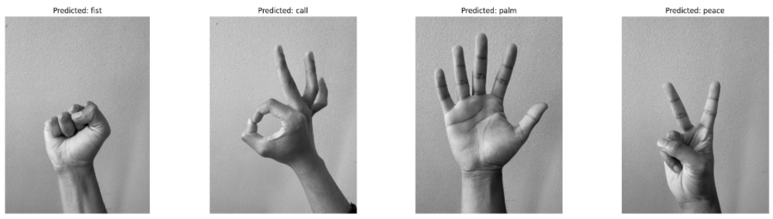


Figure 7: Grayscale Model Predictions: Hand gestures displayed in grayscale, showing superior model performance in a color-absent scenario.

4 Conclusion

This study explored the performance of custom-built Convolutional Neural Networks (CNNs) and transfer learning models for gesture recognition using the Hagrid dataset, comparing RGB and grayscale image inputs.

Key findings include:

1. **Model Accuracy**: The RGB model generally outperformed the grayscale model, highlighting the advantage of color information. However, the grayscale model showed comparable and sometimes superior performance in specific contexts.

2. **Transfer Learning with VGG-16**: Transfer learning with VGG-16 yielded significant accuracy improvements over a short training period, demonstrating the efficacy of leveraging pre-trained features.

3. **Data Management**: Efficient image resizing, normalization, and strategic data reduction were crucial for managing large datasets and optimizing computational resources.

4. **Practical Applications**: The study underscores the practical utility of both custom CNNs and transfer learning models for gesture recognition, with grayscale data showing potential benefits in certain scenarios.

Future work should explore advanced architectures, refine preprocessing techniques, and expand the dataset to include more diverse gestures, enhancing model robustness and generalizability.In conclusion, this research highlights the effectiveness of CNN-based models in gesture recognition, with color information enhancing performance while grayscale models remain viable in specific contexts. These findings provide valuable insights for developing more accurate and efficient gesture recognition systems.