

Lilies in the Valley

June 13, 2024

Emily Ramirez, Shreya Singh, Sneha Singh, Alima
Bellazer, Makayla Harden

The Lilies



Alima
Bellazer

Alima Bellazer currently attends Westover High school and 4C Academy in Albany, Ga. She would like to major in Nursing and her role in this project was presentation assistant designer.



Emily
Ramirez

Emily Ramirez attends Monroe High School in Albany, Ga. She would like to major in Civil or Aerospace Engineering and Mathematics. Her role in the project was coding assistant and data analyst.



Makayla
Harden

Makayla Harden attends Monroe High School as well. She would like to major in Environmental Science and her role was head presentation designer.



Sneha
Singh

Sneha Singh attends Houston County High School in Warner Robins, Ga. She aspires to major in Finance or any Mathematics related field and her role was coder and content researcher/writer.



Shreya
Singh

Shreya Singh attends Houston County High School in Warner Robins, Ga. She wants to major in science and was very well rounded, helping in multiple aspects and with things the team was struggling on as a whole.





Project Overview



Objective and Model Outputs:

Objective: Develop a model to classify hand gestures as rock, paper, or scissors.

Model Outputs: Outputs one of three classes: rock, paper, or scissors based on input images.



Data Set Description

Data: Color images (480x640x3) of hand gestures (rock, paper, scissors).

Classes: Scissors, Rock, Paper

Split: Separate training and testing sets for model evaluation

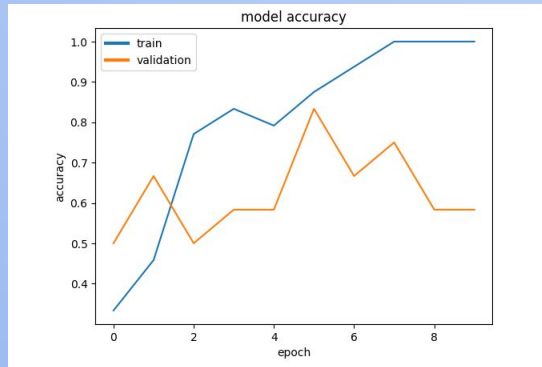


Highest Accuracy Achieved







Training Accuracy: 100%

Validation Accuracy: 58.3%

```
Training: accuracy = 1.000000 loss = 0.040201  
Validation: accuracy = 0.583333 loss = 0.699684  
Params count: 307519  
stop epoch = 9  
nb_epoch = 10
```



Methodology

-  **Data Handling:** Images loaded using Drive, organized into numpy arrays.
-  **Normalization:** Scale pixel values to $[0, 1]$.
-  **Encoding:** Class labels converted to one-hot vectors.
-  **Model Architecture:** CNN with convolutional and max-pooling layers followed by dense layers.
-  **Training:** Adam optimizer, categorical cross entropy loss.
-  **Evaluation:** Accuracy, loss metrics, confusion matrix for performance assessment.

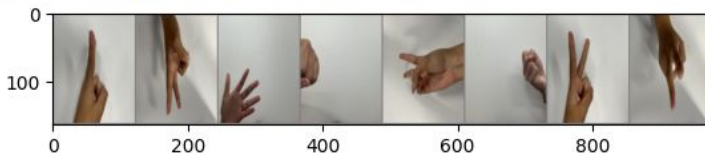


CNN Model



A CNN is like a smart detective that looks at pictures, finds important parts using special filters, shrinks down the picture to focus on the important stuff, and then figures out what's in the picture by piecing everything together like a puzzle. It can be explained by the different categories such as filters, convolution, dense layers, and pooling.

```
Accuracy of model_1 on the test images: 60 %  
Accuracy of model_2 on the test images: 53 %  
Accuracy for model_1 class: rock is 100.0 %  
Accuracy for model_1 class: paper is 0.0 %  
Accuracy for model_1 class: scissors is 80.0 %  
Accuracy for model_2 class: rock is 100.0 %  
Accuracy for model_2 class: paper is 0.0 %  
Accuracy for model_2 class: scissors is 60.0 %
```

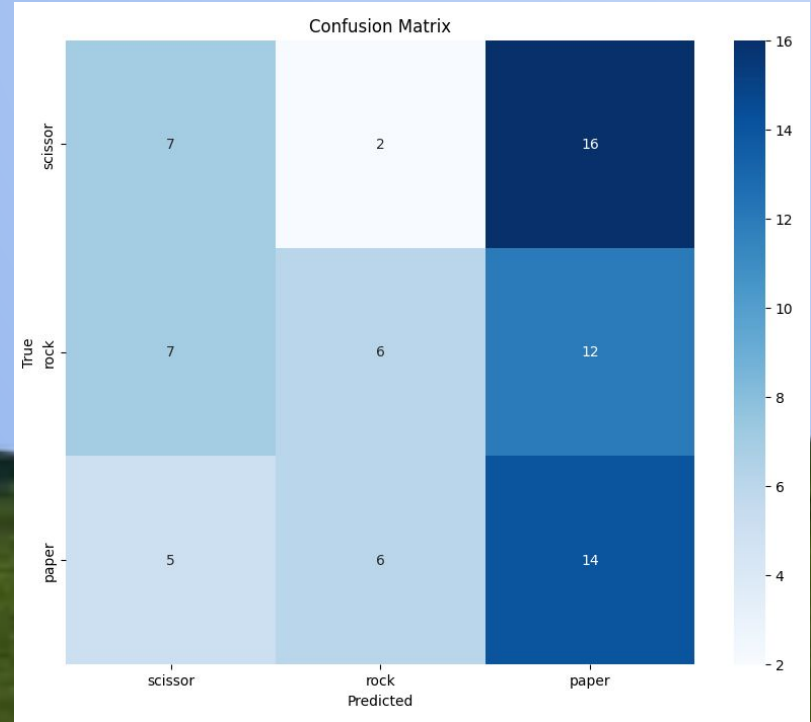




Confusion Matrix

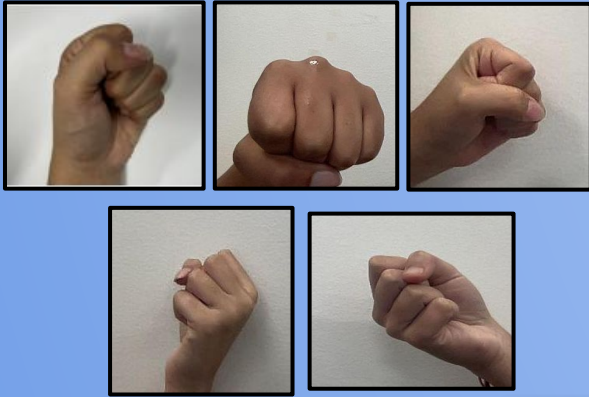


A confusion matrix helps us understand how well our model predicts different things. Imagine we have a game where we predict if someone shows a rock, paper, or scissors. In the matrix: These numbers show how our model is doing for each gesture. We use them to calculate important scores like Accuracy (how often the model is correct overall), Recall (how well it finds all instances of a gesture), Precision (how many of its predictions are correct), and more. By looking at this matrix, we can figure out ways to make our model even better, like using special methods to adjust how it learns and makes predictions.

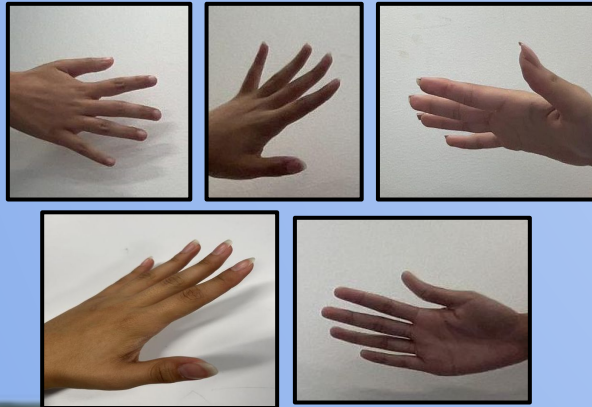


Sample Model Inputs

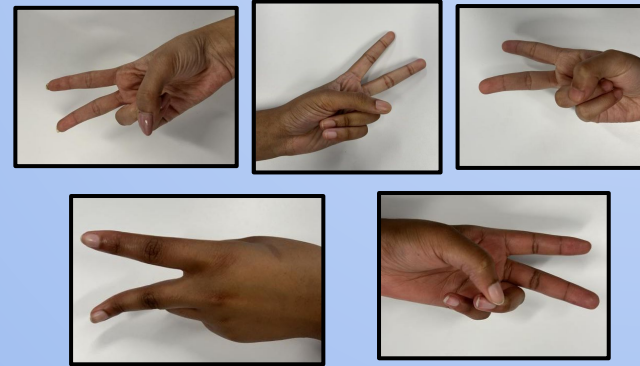
Rock



Paper



Scissors



Explanation: As observed in the images above, one of the shortcomings in our dataset was the variation in image positions, angles, and image quality. To address this, we standardized the orientation and quality of images during data collection and preprocessing.

Overfitting and Conclusion



Overfitting Analysis:

The model shows significant overfitting because the training accuracy is much higher than the validation accuracy (100% vs. 58.3%). This suggests that the model has learned to perfectly classify the training data but struggles to generalize to unseen validation data.



Methods to Improve Accuracy and Address Overfitting

Data Augmentation: Increase training data variability by applying transformations such as rotations, flips, and zooms. This exposes the model to more diverse examples and improves generalization.



Regularization Techniques:

Dropout: Randomly deactivate neurons during training to prevent over-reliance on specific features, thus improving model robustness.

Weight Regularization: Apply L1 or L2 penalties to network weights to constrain model complexity and prevent overfitting.



Transfer Learning:

Utilize pre-trained models (e.g., VGG, ResNet) trained on large datasets like ImageNet. Fine-tune these models on your rock-paper-scissors dataset to leverage learned feature representations, which can lead to improved accuracy.