Subham Srivastava

**Development and Evaluation of Gesture Recognition Models for Smart TV Control**

A Comprehensive Analysis of Deep Learning Architectures and Transfer Learning Techniques for Enhancing User Interaction

Contents

[**Problem Statement:** 2](#_Toc170830671)

[**Understanding the Dataset:** 2](#_Toc170830672)

[**Objective:** 2](#_Toc170830673)

[**Model Descriptions:** 2](#_Toc170830674)

[**Data Pre-processing:** 2](#_Toc170830675)

[**Data Generator:** 2](#_Toc170830676)

[**Model Training:** 3](#_Toc170830677)

[**Conclusion:** 3](#_Toc170830678)

[**Summary of Model Performance:** 4](#_Toc170830679)

**Problem Statement:**

You are a data scientist at a home electronics company developing a smart TV feature to recognize five user gestures, allowing remote-free control:

* **Thumbs up**: Increase volume
* **Thumbs down**: Decrease volume
* **Left swipe**: Rewind 10 seconds
* **Right swipe**: Fast forward 10 seconds
* **Stop**: Pause the movie

**Understanding the Dataset:**

The dataset includes hundreds of short videos (2-3 seconds, 30 frames each) of people performing the gestures. The data is divided into 'train' and 'val' folders, each with corresponding CSV files.

**Objective:**

Train models on the 'train' dataset to accurately predict gestures and evaluate performance on the 'val' dataset. The final model will be tested on a withheld 'test' set.

**Model Descriptions:**

1. **CNN + RNN Architecture**:
   * **CNN**: Extracts feature vectors from each frame.
   * **RNN**: Processes the sequence of feature vectors.
   * **Output**: Softmax layer for classification.
2. **3D Convolutional Neural Networks (Conv3D)**:
   * Processes video as a 4D tensor (e.g., 100 x 100 x 30 x 3).
   * Uses 3D filters to convolve across the x, y, and z dimensions.

**Data Pre-processing:**

* **Resizing**: Standardizes frame dimensions.
* **Normalization**: Mitigates lighting and shadow distortions.

**Data Generator:**

Pre-processes images to handle different dimensions (70x70, 100x100, 120x120) and creates batches of video frames. Ensures consistent cropping, resizing, and normalization.

**Model Training:**

* **Experiments**: Tested various model configurations, hyper-parameters, batch sizes, image dimensions, filter sizes, padding, and stride lengths.
* **Optimization**: Used SGD optimizer for better accuracy, adjusting parameters like decay rate and learning rate.
* **Regularization**: Implemented batch normalization, pooling, and dropout layers to combat overfitting.
* **Early Stopping**: Halted training when validation loss plateaued.

**Conclusion:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Model Number** | **Architecture** | **Epochs** | **Batch Size** | **Shape** | **Frames** | **Training Accuracy** | **Validation Accuracy** | **Model Analysis** |
| 1 | Conv3D | 10 | 32 | (120,120) | 10 | 52% | 42% | Low accuracy; not overfitted. Increase batch size and epochs to improve accuracy. |
| 2 | Conv3D | 10 | 50 | (120,120) | 10 | 50% | 28% | Increased batch size reduced runtime but not accuracy. |
| 3 | Conv3D | 10 | 50 | (100,100) | 10 | 50% | 29% | Decreased image size slightly overfitted. |
| 4 | Conv3D | 10 | 50 | (70,70) | 10 | 60% | 32% | Improved training accuracy but signs of overfitting. |
| 5 | Conv3D | 50 | 50 | (70,70) | 10 | 94% | 58% | Higher epochs improved accuracy but some evidence of overfitting. |
| 6 | CNN + RNN (GRU) | 50 | 50 | (70,70) | 10 | 90% | 75% | Good accuracy |
| 7 | CNN + RNN (GRU) | 50 | 50 | (120,120) | 10 | 88% | 64% | Shows some evidence of overfitting. |
| 8 | Conv2D + Dense | 50 | 50 | (70,70) | 10 | 96% | 71% | High accuracy, better performance. |
| 9 | TimeDistributed + ConvLSTM2D | 50 | 50 | (70,70) | 10 | 63% | 45% | Poor accuracy. |
| 10 | TimeDistributed + ConvLSTM2D | 50 | 50 | (120,120) | 10 | 65% | 52% | Poor accuracy. |
| 11 | Transfer Learning (VGG16) | 50 | 50 | (120,120) | 10 | 99.50% | 85% | Best model with high accuracy and no clear sign of overfitting |
| 12 | Transfer Learning (ResNet50) | 50 | 50 | (120,120) | 10 | 94% | 43% | High training accuracy, but clear overfitting. |

**Summary of Model Performance:**

The project involved experimenting with various models and configurations to achieve optimal gesture recognition for smart TV controls. Key observations include:

* **Conv3D Models**:
  + Early models (Models 1-5) struggled with low accuracy and signs of overfitting. Increasing epochs and adjusting image size helped, but improvements were limited.
* **CNN + RNN Models**:
  + Model 6, using a TimeDistributed CNN2D with GRU, achieved high accuracy (90% training, 75% validation) with no overfitting, indicating strong performance.
  + Model 7 showed some overfitting with slightly lower validation accuracy.
* **Conv2D + Dense Model**:
  + Model 8 performed well with 96% training and 71% validation accuracy, showing it to be a robust configuration.
* **Conv2D + LSTM Models**:
  + Models 9 and 10 showed poor accuracy, indicating this architecture was not effective for this task.
* **Transfer Learning Models**:
  + Model 11 (VGG16) achieved the best results with 99.5% training and 85% validation accuracy, making it the top-performing model.
  + Model 12 (ResNet50) showed high training accuracy but significant overfitting with low validation accuracy.

Overall, transfer learning using VGG16 provided the best performance, while CNN + RNN and Conv2D + Dense architectures also showed promising results. Models using Conv2D + LSTM were the least effective.