|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model Performance Summary Table** | | | | |
| **Index** | **Model** | **Key Hyperparameters** | **Results** | **Conclusion & Explanation** |
| 1 | Conv2D + GRU  **Conv2D+GRU.keras** | Conv2D (16, 32, 64 filters), GRU (64 units), Dense (5 units), Dropout (40%) | Accuracy: Train 98.79%,  Val 94.0% | The model learns spatial and temporal features effectively. Conv2D layers capture spatial features, while GRU layers handle temporal dependencies. Regularization with dropout minimizes overfitting, enabling strong generalization. This model performs robustly due to the balance of spatial and temporal learning. |
| 2 | Conv2D + LSTM  **Conv2D+LSTM.keras** | Conv2D (16, 32, 64 filters), LSTM (32 units), Dense (5 units), Dropout (50%) | Accuracy: Train 92.01%,  Val 91.0% | Conv2D layers extract spatial patterns effectively, while the LSTM layer captures temporal dependencies. However, its slightly lower accuracy compared to Conv2D+GRU indicates GRUs might better capture the temporal nuances for gesture recognition tasks. Dropout helps reduce overfitting. |
| 3 | Conv3D Without Pretraining  **Conv3D-32-64-128.keras** | 3 Conv3D layers (filters: 32, 64, 128), GlobalAvgPooling, Dense (128 units, dropout 50%) | Accuracy: Train 93.82%,  Val 89.0% | The Conv3D model effectively learns spatial and temporal features. Its consistent performance shows that raw spatial-temporal features are well captured without needing a pretrained base. The dropout layer effectively regularizes training. Further optimization could focus on data augmentation. |
| 4 | GRU With MediaPipe Keypoints  **pretrained-mediapipe+gru.keras** | GRU (64 units), Flatten, Dense (5 units), Dropout (50%) | Accuracy: Train 95.5%,  Val 99.0% | MediaPipe keypoints simplify the input space, leading to a lightweight model with only 25k parameters. The GRU efficiently models the temporal dependencies in hand gestures, producing excellent generalization. This approach is computationally efficient, ideal for real-time applications, and robust due to keypoint-based input. |
| 5 | MobileNetV2 + GRU (Pretrained)  **pretrained-MobileNetV2+GRU.keras** | Pretrained MobileNetV2, GRU (32 units), Dense (5 units), Dropout (50%) | Accuracy: Train 92.01%,  Val 91.0% | The pretrained MobileNetV2 effectively extracts spatial features, while the GRU layer models temporal dependencies. However, freezing the pretrained base limits the model’s ability to adapt to specific gesture tasks. Fine-tuning the base layers could further improve performance. |
| 6 | MobileNetV3Small + GRU  **pretrained-MobileNetV3Small+GRU.keras** | Pretrained MobileNetV3Small, GRU (64 units), Dense (5 units), Dropout (50%) | Accuracy: Train 40%,  Val 45% | The frozen MobileNetV3Small base limits the model’s performance, resulting in underfitting. Temporal modeling via GRU is insufficient to compensate for the lack of fine-tuning. This model requires significant improvements through fine-tuning, data augmentation, or more robust temporal modeling. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Best Models Summary** | | | | |
| **Category** | **Model** | **Key Hyperparameters** | **Results** | **Why This Model Stands Out** |
| **Non-Pretrained** | Conv3D  **Conv3D-32-64-128.keras** | 3 Conv3D layers (filters: 32, 64, 128), GlobalAvgPooling, Dense (128 units, dropout 50%) | Accuracy: Train 93.82%,  Val 89.0% | Conv3D achieves consistent performance with stable loss and accuracy trends. Its ability to learn spatial and temporal features simultaneously, without relying on pretrained weights, makes it a robust option. The stability suggests potential for improvement with additional training. |
| **Pretrained** | GRU with MediaPipe Keypoints  **pretrained-mediapipe+gru.keras** | GRU (64 units), Flatten, Dense (5 units), Dropout (50%), MediaPipe Hand Keypoints | Accuracy: Train 95.5%,  Val 99.0% | Leveraging MediaPipe’s pretrained keypoint extractor reduces input complexity, allowing the lightweight GRU-based model to achieve exceptional accuracy and efficiency. Its simplicity and computational efficiency make it ideal for real-time applications while maintaining robust generalization. |