|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Experiment No | Model | No of Images | Image Size | No of epoch | Batch size | No of params | Result | Decision + Explanation |
| 1 | Conv3D | 30 | 160x160 | 30 | 40 | 9902149 | **train\_accuracy 99% and validation accuracy 47%** | The model heavily overfitted, as seen in the large gap between training (99%) and validation (47%) accuracies. The high parameter count (~9.9M) combined with limited data likely caused overfitting. The architecture uses a simple TimeDistributed CNN followed by an LSTM, which provides a good baseline but lacks advanced regularization techniques like BatchNormalization or dropout. It is suitable for initial experiments but limited by shallow feature extraction and high memory requirements. For initial experiments, this approach is viable, but deeper feature extraction and regularization are required. |
| 2 | Conv3D | 18 | 120x120 | 15 | 25 | 79941 | **train\_accuracy of 45 % and val\_accuracy of 51 %.** | Reducing input size and number of frames drastically reduced parameters (~79k), leading to better generalization but causing underfitting due to insufficient feature extraction. Validation accuracy (51%) improved slightly compared to Experiment 1. This improved baseline adds extra convolution layers but still lacks regularization layers for stable training. Future iterations should increase the number of epochs and batch size to prevent premature convergence. |
| 3 | Conv3D | 18 | 120x120 | 20 | 30 | 79941 | **train\_accuracy of 89 % and val\_accuracy of 76 % .** | Increasing training epochs improved learning, raising validation accuracy to 76%. However, the architecture’s limited feature extraction capabilities caused a plateau in performance. The input resolution (120x120) provides a balance between computational efficiency and spatial granularity but may need dropout or augmentation to address overfitting. |
| 4 | Conv3D | 18 | 160x160 | 50 | 30 | 79941 | **train\_accuracy of 93 % and val\_accuracy of 77 %.** | Increasing the input resolution and parameters improved feature extraction but led to marginal gains in validation accuracy (77%). The increased computational cost outweighs the benefits, suggesting a need to explore hybrid architectures (e.g., CNN+RNN) for better performance. This model is well-suited for datasets requiring higher spatial detail. |
| 5 | CNN + LSTM | 20 | 120x120 | 30 | 30 | 14922949 | **train\_accuracy of 99 % and val\_accuracy of 60 %.** | The hybrid architecture improved temporal modeling, but validation accuracy (60%) indicated underfitting due to limited training epochs. Incorporates BatchNormalization and pooling layers, improving stability, but lacks dropout, leading to potential overfitting on larger datasets. Suggested increasing epochs and exploring data augmentation for further improvements. This approach is suitable for initial experiments requiring sequence modeling. |
| 6 | CNN + LSTM | 20 | 120x120 | 50 | 30 | 26391749 | **train\_accuracy of 99 % and val\_accuracy of 67 %.** | Increasing epochs improved validation accuracy to 67%, but the model was computationally expensive (~26M parameters). Explored reducing complexity by tuning resolution and experimenting with alternative hybrid architectures. A well-rounded model for balanced performance with BatchNormalization, L2 regularization, and appropriate pooling layers. The resolution (120x120) might limit finer detail extraction in complex datasets. |
| 7 | CNN-LSTM Model | 20 | 160x160 | 20 | 20 | 5080677 | **train\_accuracy of 99% and val\_accuracy of 65 %.** | Reduced parameter count by optimizing architecture, decreasing computational load but maintaining performance. Validation accuracy remained at 65%, showing diminishing returns. Decided to try GRU for reduced complexity and faster convergence. Suitable for memory-efficient tasks with temporal dependencies. Incorporates BatchNormalization and dropout for regularization. |
| 8 | CNN-GRU Model | 20 | 160x160 | 20 | 30 | 409285 | **train\_accuracy of 99% and val\_accuracy of 48 %.** | GRU-based architecture was lightweight (~409k params), but performance was poor, with validation accuracy dropping to 48%. Decided to leverage transfer learning with MobileNet for efficient feature extraction. Adds more Conv2D layers and dropout, making it robust to overfitting while still being computationally efficient. |
| 9 | MobileNet Transfer Learning Model - without updating weights + LSTM | 20 | 160x160 | 20 | 30 | 5413317 | **train\_accuracy of 99% and val\_accuracy of 76 %.** | Transfer learning significantly boosted performance, with validation accuracy reaching 76%. However, the lack of weight updates limited the model’s ability to fine-tune to the dataset. Enabled weight updates for further improvements. Good for leveraging pre-trained models in a computationally efficient manner. Uses MobileNet with frozen weights for feature extraction but limited fine-tuning capacity. |
| 10 | MobileNet Transfer Learning Model - with updating weights + GRU | 18 | 120x120 | 20 | 20 | 3693253 | **train\_accuracy of 99% and val\_accuracy of 89 %.** | Updating weights allowed fine-tuning, enabling the model to learn dataset-specific features. Achieved the highest validation accuracy (89%) with a relatively lightweight architecture (~3.7M params). This was chosen as the final model. Builds on Model 9 but allows training of MobileNet layers. This flexibility provides better feature learning and customization for specific datasets, making it ideal for complex and diverse datasets. |