**“Work Execution”**

Please follow the following steps for running the model for Video Analysis:

1. Make sure you have the requirements.txt file in your current working directory.
2. Install the Required Modules Open your terminal or command prompt and execute the following command to install the necessary modules:
   * pip install -r requirements.txt
3. Download the UCF101 Dataset:
   * Visit the [link](https://www.kaggle.com/datasets/pevogam/ucf101/download?datasetVersionNumber=1) (UCF101 dataset) and download the dataset onto your local machine.
4. Extract the downloaded dataset files and create a new directory named "input". Place all the extracted files inside the "input" directory.
5. Launch your preferred Python environment, such as Jupyter Notebook or Anaconda, and open the Python notebook file related to the video analysis model. Make sure the notebook is in the same project directory.

That's it! You are now ready to run the video analysis model using the UCF101 dataset. Just make sure to execute each step carefully, ensuring that the required modules are installed, the dataset is downloaded and organised correctly, and the Python notebook is executed in an appropriate environment.

Following 2 models are implemented for the Video Analysis:

1. CNN + LSTM (ConvLSTM2D)
2. EfficientNet

The following steps are followed in the code while execution of both models:

1. Import Necessary Libraries: At the beginning of the code, all the required libraries and modules are imported. This includes libraries for data manipulation, deep learning frameworks, image processing, evaluation metrics, and visualisation.
2. Load the Dataset: The dataset is loaded from the "input" directory that contains the video files and their corresponding labels. Depending on the specific implementation, the dataset may be in various formats such as frames extracted from videos or video files themselves.
3. Split the Dataset into Train and Test: The loaded dataset is then divided into two subsets: a training set and a test set. The training set is used to train the model, while the test set is used to evaluate its performance on unseen data.
4. Configure the Model for Training: The model architecture and its hyperparameters are defined or loaded from a pre-trained model. This includes specifying the number of layers, type of layers (e.g., convolutional, recurrent), activation functions, optimizer, and loss function. The model is set up and compiled before training.
5. Train and Validate the Model: The model is trained using the training set and validated using a validation set. The training process typically involves iterating over multiple epochs (complete passes through the training set) and updating the model's parameters to minimise the defined loss function. The validation set helps monitor the model's performance and prevent overfitting.
6. Plot Train and Validation Accuracy: During or after the training process, the train and validation accuracy metrics are computed for each epoch. These metrics indicate how well the model is learning the patterns in the data. These accuracy values are often visualised using line plots to analyse the model's performance over time and identify any overfitting or underfitting issues.
7. Test Accuracy and Confusion Matrix: Once the model is trained, it is evaluated on the test set to determine its final accuracy on unseen data. The test accuracy is calculated, and a confusion matrix is often generated to analyze the model's performance on different classes. The confusion matrix provides insights into the model's ability to correctly classify videos into their respective categories.

CNN + LSTM (ConvLSTM2D) Model Summary:

The model is a sequential model comprising multiple layers for video analysis. It starts with Batch Normalisation to normalise the input data. Then, it incorporates multiple ConvLSTM2D layers, each followed by MaxPooling3D, Dropout, and Batch Normalisation layers. These layers capture spatial-temporal features, downsample the data, and prevent overfitting. The model concludes with Flatten, Dense (4096 units), and Dense (5 units with softmax activation) layers for classification.

EfficientNet Model Summary:

The efficient\_net\_model is based on the EfficientNetB0 architecture, a pre-trained convolutional neural network. The model's top layer is excluded using "include\_top=False" to enable transfer learning. The EfficientNetB0 layers are wrapped within TimeDistributed, which applies the same network to each time step of the input. A Dense layer with 5 units is added, followed by GlobalAveragePooling3D to aggregate spatial information. The model is compiled using the Adam optimizer, sparse categorical cross-entropy loss function, and accuracy as the evaluation metric.

Summary:

The video analysis model, combining machine learning with computer vision techniques, demonstrates the power of CNN + LSTM (ConvLSTM2D) and EfficientNet architectures. By leveraging deep learning algorithms, this model achieves exceptional accuracy in video classification tasks. Its integration of advanced machine learning and computer vision methodologies makes it a robust and practical solution for real-world video analysis applications.